

AlderLake Intel® Firmware Support Package (FSP) Integration Guide

Fri Nov 3 2023 00:18:09

By using this document, in addition to any agreements you have with Intel, you accept the terms set forth below. You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: <http://www.intel.com/design/literature.htm>

Any software source code reprinted in this document is furnished for informational purposes only and may only be used or copied and no license, express or implied, by estoppel or otherwise, to any of the reprinted source code is granted by this document.

[When the doc contains software source code for a special or limited purpose (such as informational purposes only), use the conditionalized Software Disclaimer tag. Otherwise, use the generic software source code disclaimer from the Legal page and include a copy of the software license or a hyperlink to its permanent location.]

This document contains information on products in the design phase of development. Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. Go to: http://www.intel.com/products/processor_number/

Code Names are only for use by Intel to identify products, platforms, programs, services, etc. ("products") in development by Intel that have not been made commercially available to the public, i.e., announced, launched or shipped. They are never to be used as "commercial" names for products. Also, they are not intended to function as trademarks.

Intel, Intel Atom, [include any Intel trademarks which are used in this document] and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

Copyright ©Intel Corporation. All rights reserved.

1 Introduction	1
2 Overview	3
3 FSP Integration	5
4 FSP Dispatch Mode	9
4.1 Dispatch Mode Policy Init	10
4.2 Config Blocks	12
4.3 FSP Error Information	17
4.4 Dispatch Mode Integration	18
5 FSP API Mode	21
5.1 FSP APIs	21
5.2 Reset Return Codes	25
5.3 UPD Porting Guide	26
6 Porting Recommendations	27
7 FSP Output	29
8 POST Codes	33
9 Deprecated List	41
10 Todo List	43
11 Class Index	45
11.1 Class List	45
12 File Index	55
12.1 File List	55
13 Class Documentation	59
13.1 _CONFIG_BLOCK Struct Reference	59
13.1.1 Detailed Description	60
13.2 _CONFIG_BLOCK_HEADER Struct Reference	60
13.2.1 Detailed Description	61
13.3 _CONFIG_BLOCK_TABLE_STRUCT Struct Reference	61
13.3.1 Detailed Description	62
13.4 _EFI_MM_RESERVED_MMRAM_REGION Struct Reference	62
13.4.1 Detailed Description	62
13.4.2 Member Data Documentation	62
13.4.2.1 MmramReservedSize	62
13.4.2.2 MmramReservedStart	63
13.5 _EFI_PEI_MP_SERVICES_PPI Struct Reference	63
13.5.1 Detailed Description	63

13.6 _ITBT_GENERIC_CONFIG Struct Reference	63
13.6.1 Detailed Description	64
13.6.2 Member Data Documentation	64
13.6.2.1 ITbtForcePowerOnTimeoutInMs	64
13.7 _ITBT_ROOTPORT_CONFIG Struct Reference	64
13.7.1 Detailed Description	65
13.8 _LIST_ENTRY Struct Reference	65
13.8.1 Detailed Description	65
13.9 _PEI_ITBT_CONFIG Struct Reference	66
13.9.1 Detailed Description	66
13.10 _PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI Struct Reference	67
13.10.1 Detailed Description	67
13.11 _PEI_SI_DEFAULT_POLICY_INIT_PPI Struct Reference	67
13.11.1 Detailed Description	67
13.12 _PPM_CUSTOM_CTDp_TABLE Struct Reference	68
13.12.1 Detailed Description	68
13.13 _SI_POLICY_STRUCT Struct Reference	68
13.13.1 Detailed Description	69
13.14 _SI_PREMEM_POLICY_STRUCT Struct Reference	70
13.14.1 Detailed Description	70
13.15 ADR_CONFIG Struct Reference	71
13.15.1 Detailed Description	72
13.16 ADR_SOURCE_ENABLE Union Reference	72
13.16.1 Detailed Description	72
13.17 AMT_DXE_CONFIG Struct Reference	73
13.17.1 Detailed Description	74
13.17.2 Member Data Documentation	74
13.17.2.1 AmtbxHotkeyPressed	74
13.17.2.2 AmtbxSelectionScreen	74
13.17.2.3 CiraTimeout	75
13.17.2.4 CpuReplacementTimeout	75
13.17.2.5 HideUnConfigureMeConfirm	75
13.17.2.6 MebxDebugMsg	75
13.17.2.7 MebxGraphicsMode	76
13.17.2.8 MebxNonUiTextMode	76
13.17.2.9 MebxUiTextMode	76
13.17.2.10 OemResolutionSettingsRsvd	76
13.18 AMT_PEI_CONFIG Struct Reference	77
13.18.1 Detailed Description	78
13.18.2 Member Data Documentation	78
13.18.2.1 AmtEnabled	78
13.18.2.2 AmtKvmEnabled	78

13.18.2.3 ManageabilityMode	78
13.18.2.4 RemoteAssistance	79
13.18.2.5 WatchDogEnabled	79
13.18.2.6 WatchDogTimerBios	79
13.18.2.7 WatchDogTimerOs	79
13.19 BIOS_GUARD_CONFIG Struct Reference	80
13.19.1 Detailed Description	81
13.20 CNVI_CONFIG Struct Reference	81
13.20.1 Detailed Description	82
13.20.2 Member Data Documentation	82
13.20.2.1 BtAudioOffload	82
13.20.2.2 Mode	83
13.21 CNVI_PIN_MUX Struct Reference	83
13.21.1 Detailed Description	83
13.22 CNVI_PREMEM_CONFIG Struct Reference	84
13.22.1 Detailed Description	84
13.23 CPU_CONFIG Struct Reference	85
13.23.1 Detailed Description	86
13.23.2 Member Data Documentation	86
13.23.2.1 AcSplitLock	86
13.23.2.2 AesEnable	87
13.23.2.3 Avx3Disable	87
13.23.2.4 AvxDisable	87
13.23.2.5 PpinSupport	88
13.23.2.6 SmbiosType4MaxSpeedOverride	88
13.23.2.7 TxtEnable	88
13.23.2.8 X2ApicSupport	88
13.24 CPU_CONFIG_LIB_PREMEM_CONFIG Struct Reference	89
13.24.1 Detailed Description	90
13.24.2 Member Data Documentation	91
13.24.2.1 ActiveCoreCount	91
13.24.2.2 ActiveCoreCount1	91
13.24.2.3 ActiveSmallCoreCount	92
13.24.2.4 ActiveSocNorthAtomCoreCount	92
13.24.2.5 BclkSource	92
13.24.2.6 BootFrequency	93
13.24.2.7 ComputeDieSscEnable	93
13.24.2.8 ComputeDieSscValue	93
13.24.2.9 CpuRatio	94
13.24.2.10 CrashLogEnable	94
13.24.2.11 CrashLogGprs	94
13.24.2.12 FClkFrequency	94

13.24.2.13 InfDfxPwrEnable	95
13.24.2.14 PeciC10Reset	95
13.24.2.15 PeciSxReset	95
13.24.2.16 SocDieSscEnable	96
13.24.2.17 SocDieSscValue	96
13.24.2.18 TmeEnable	96
13.24.2.19 VmxEnable	97
13.25 CPU_DMI_EQ_PARAM Struct Reference	97
13.25.1 Detailed Description	97
13.26 CPU_DMI_PREMEM_CONFIG Struct Reference	98
13.26.1 Detailed Description	99
13.26.2 Member Data Documentation	100
13.26.2.1 DmiGen3EqPh2Enable	100
13.26.2.2 DmiGen3EqPh3Method	100
13.26.2.3 DmiGen3ProgramStaticEq	100
13.26.2.4 DmiGen3RxCtlePeaking	101
13.26.2.5 DmiMaxLinkSpeed	101
13.27 CPU_PCIE_CONFIG Struct Reference	101
13.27.1 Detailed Description	102
13.27.2 Member Data Documentation	102
13.27.2.1 EqPh3LaneParam	103
13.27.2.2 PcieDeviceOverrideTablePtr	103
13.27.2.3 PegGen3ProgramStaticEq	103
13.27.2.4 PegGen4ProgramStaticEq	103
13.27.2.5 SetSecuredRegisterLock	104
13.28 CPU_PCIE_EQ_LANE_PARAM Struct Reference	104
13.28.1 Detailed Description	104
13.29 CPU_PCIE_GPIO_INFO Struct Reference	105
13.29.1 Detailed Description	105
13.30 CPU_PCIE_ROOT_PORT_CONFIG Struct Reference	105
13.30.1 Detailed Description	106
13.30.2 Member Data Documentation	106
13.30.2.1 Gen4EqPh3Method	106
13.31 CPU_PCIE_RP_PREMEM_CONFIG Struct Reference	107
13.31.1 Detailed Description	108
13.31.2 Member Data Documentation	108
13.31.2.1 ClkReqMsgEnable	108
13.31.2.2 ClkReqMsgEnableRp	108
13.31.2.3 LinkDownGpios	109
13.31.2.4 PcieSpeed	109
13.31.2.5 RpEnabledMask	109
13.31.2.6 SlotImplemented	109

13.32 CPU_PCIE_RTD3_GPIO Struct Reference	110
13.32.1 Detailed Description	110
13.33 CPU_POWER_MGMT_BASIC_CONFIG Struct Reference	111
13.33.1 Detailed Description	113
13.33.2 Member Data Documentation	114
13.33.2.1 AtomTurboRatioLimitNumCore	114
13.33.2.2 AtomTurboRatioLimitRatio	114
13.33.2.3 BootFrequency	115
13.33.2.4 DualTauBoost	115
13.33.2.5 EightCoreRatioLimit	115
13.33.2.6 EnableEpbPeciOverride	116
13.33.2.7 EnableFastMsRhwpReq	116
13.33.2.8 EnableHwpAutoEppGrouping	116
13.33.2.9 EnableHwpAutoPerCorePstate	116
13.33.2.10 EnableHwpScalabilityTracking	117
13.33.2.11 EnableIltbm	117
13.33.2.12 EnableIltbmDriver	117
13.33.2.13 EnablePerCorePState	117
13.33.2.14 FiveCoreRatioLimit	118
13.33.2.15 FourCoreRatioLimit	118
13.33.2.16 HdcControl	118
13.33.2.17 Hwp	119
13.33.2.18 OneCoreRatioLimit	119
13.33.2.19 PowerLimit1	119
13.33.2.20 PowerLimit1Time	120
13.33.2.21 PowerLimit2Power	120
13.33.2.22 PowerLimit3	120
13.33.2.23 PowerLimit4	120
13.33.2.24 SevenCoreRatioLimit	121
13.33.2.25 SixCoreRatioLimit	121
13.33.2.26 TccActivationOffset	121
13.33.2.27 TccOffsetClamp	122
13.33.2.28 TccOffsetTimeWindowForRatI	122
13.33.2.29 ThreeCoreRatioLimit	122
13.33.2.30 TurboRatioLimitNumCore	122
13.33.2.31 TurboRatioLimitRatio	123
13.33.2.32 TwoCoreRatioLimit	123
13.34 CPU_POWER_MGMT_CUSTOM_CONFIG Struct Reference	123
13.34.1 Detailed Description	124
13.35 CPU_POWER_MGMT_PSYS_CONFIG Struct Reference	124
13.35.1 Detailed Description	126
13.36 CPU_POWER_MGMT_TEST_CONFIG Struct Reference	126

13.36.1 Detailed Description	128
13.36.2 Member Data Documentation	129
13.36.2.1 C3AutoDemotion	129
13.36.2.2 C3UnDemotion	129
13.36.2.3 ConfigTdpLevel	129
13.36.2.4 CstateLatencyControl0TimeUnit	130
13.36.2.5 CustomPowerUnit	130
13.36.2.6 PpmIrmSetting	130
13.37 CPU_POWER_MGMT_VR_CONFIG Struct Reference	131
13.37.1 Detailed Description	134
13.37.2 Member Data Documentation	135
13.37.2.1 EnableFastVmode	135
13.37.2.2 FivrRfiFrequency	135
13.37.2.3 FivrSpreadSpectrum	136
13.37.2.4 IccLimit	136
13.37.2.5 TdcTimeWindow	136
13.37.2.6 VccInDemotionCapacitanceInUf	136
13.37.2.7 VccInDemotionEnable	137
13.37.2.8 VccInDemotionQuiescentPowerInMw	137
13.37.2.9 VrPowerDeliveryDesign	137
13.37.2.10 VsysAssertionDeglitchMantissa	137
13.37.2.11 VsysFullScale	138
13.38 CPU_SECURITY_PREMEM_CONFIG Struct Reference	138
13.38.1 Detailed Description	139
13.38.2 Member Data Documentation	139
13.38.2.1 BiosGuard	140
13.38.2.2 EnableC6Dram	140
13.38.2.3 EnableSgx	140
13.38.2.4 Txt	141
13.39 CPU_TEST_CONFIG Struct Reference	141
13.39.1 Detailed Description	142
13.39.2 Member Data Documentation	142
13.39.2.1 ProcessorTraceMemSize	142
13.40 CPU_TRACE_HUB_PREMEM_CONFIG Struct Reference	143
13.40.1 Detailed Description	143
13.41 CPU_TXT_PREMEM_CONFIG Struct Reference	144
13.41.1 Detailed Description	145
13.42 DDI_CONFIGURATION Struct Reference	145
13.42.1 Detailed Description	146
13.43 DMI_HW_WIDTH_CONTROL Struct Reference	146
13.43.1 Detailed Description	147
13.44 EFI_HOB_CPU Struct Reference	147

13.44.1 Detailed Description	148
13.44.2 Member Data Documentation	148
13.44.2.1 Header	148
13.45 EFI_HOB_FIRMWARE_VOLUME Struct Reference	148
13.45.1 Detailed Description	149
13.45.2 Member Data Documentation	149
13.45.2.1 Header	149
13.46 EFI_HOB_FIRMWARE_VOLUME2 Struct Reference	149
13.46.1 Detailed Description	150
13.46.2 Member Data Documentation	150
13.46.2.1 Header	150
13.47 EFI_HOB_FIRMWARE_VOLUME3 Struct Reference	151
13.47.1 Detailed Description	151
13.47.2 Member Data Documentation	151
13.47.2.1 ExtractedFv	152
13.47.2.2 FileName	152
13.47.2.3 FvName	152
13.47.2.4 Header	152
13.48 EFI_HOB_GENERIC_HEADER Struct Reference	153
13.48.1 Detailed Description	153
13.49 EFI_HOB_GUID_TYPE Struct Reference	153
13.49.1 Detailed Description	154
13.49.2 Member Data Documentation	154
13.49.2.1 Header	154
13.50 EFI_HOB_HANDOFF_INFO_TABLE Struct Reference	154
13.50.1 Detailed Description	155
13.50.2 Member Data Documentation	155
13.50.2.1 EfiMemoryTop	155
13.50.2.2 Header	155
13.50.2.3 Version	156
13.51 EFI_HOB_MEMORY_ALLOCATION Struct Reference	156
13.51.1 Detailed Description	157
13.51.2 Member Data Documentation	157
13.51.2.1 Header	157
13.52 EFI_HOB_MEMORY_ALLOCATION_BSP_STORE Struct Reference	157
13.52.1 Detailed Description	158
13.52.2 Member Data Documentation	158
13.52.2.1 Header	158
13.53 EFI_HOB_MEMORY_ALLOCATION_HEADER Struct Reference	158
13.53.1 Detailed Description	159
13.53.2 Member Data Documentation	159
13.53.2.1 MemoryBaseAddress	159

13.53.2.2 MemoryType	159
13.53.2.3 Name	160
13.54 EFI_HOB_MEMORY_ALLOCATION_MODULE Struct Reference	160
13.54.1 Detailed Description	161
13.54.2 Member Data Documentation	161
13.54.2.1 Header	161
13.55 EFI_HOB_MEMORY_ALLOCATION_STACK Struct Reference	161
13.55.1 Detailed Description	162
13.55.2 Member Data Documentation	162
13.55.2.1 Header	162
13.56 EFI_HOB_MEMORY_POOL Struct Reference	162
13.56.1 Detailed Description	163
13.56.2 Member Data Documentation	163
13.56.2.1 Header	163
13.57 EFI_HOB_RESOURCE_DESCRIPTOR Struct Reference	163
13.57.1 Detailed Description	164
13.57.2 Member Data Documentation	164
13.57.2.1 Header	164
13.57.2.2 Owner	164
13.58 EFI_HOB_UEFI_CAPSULE Struct Reference	165
13.58.1 Detailed Description	165
13.58.2 Member Data Documentation	165
13.58.2.1 BaseAddress	166
13.59 EFI_IP_ADDRESS Union Reference	166
13.59.1 Detailed Description	166
13.60 EFI_MAC_ADDRESS Struct Reference	167
13.60.1 Detailed Description	167
13.61 EFI_MMRAM_DESCRIPTOR Struct Reference	167
13.61.1 Detailed Description	167
13.61.2 Member Data Documentation	167
13.61.2.1 CpuStart	168
13.61.2.2 PhysicalStart	168
13.61.2.3 RegionState	168
13.62 EFI_PEI_HOB_POINTERS Union Reference	169
13.62.1 Detailed Description	169
13.63 EFI_TIME Struct Reference	169
13.63.1 Detailed Description	170
13.64 FIRMWARE_VERSION Struct Reference	170
13.64.1 Detailed Description	170
13.65 FIRMWARE_VERSION_INFO Struct Reference	170
13.65.1 Detailed Description	171
13.66 FIRMWARE_VERSION_INFO_HOB Struct Reference	171

13.66.1 Detailed Description	172
13.67 FIVR_EXT_RAIL_CONFIG Struct Reference	172
13.67.1 Detailed Description	172
13.67.2 Member Data Documentation	172
13.67.2.1 IccMax	173
13.67.2.2 SupportedVoltageStates	173
13.67.2.3 Voltage	173
13.68 FIVR_VCCIN_AUX_CONFIG Struct Reference	173
13.68.1 Detailed Description	174
13.68.2 Member Data Documentation	174
13.68.2.1 LowToHighCurModeVolTranTime	174
13.68.2.2 OffToHighCurModeVolTranTime	174
13.68.2.3 RetToHighCurModeVolTranTime	175
13.68.2.4 RetToLowCurModeVolTranTime	175
13.69 FSP_ERROR_INFO_HOB Struct Reference	175
13.69.1 Detailed Description	176
13.70 FSPM_ARCH_CONFIG_PPI Struct Reference	176
13.70.1 Detailed Description	176
13.71 GBE_CONFIG Struct Reference	177
13.71.1 Detailed Description	177
13.71.2 Member Data Documentation	178
13.71.2.1 Enable	178
13.72 GNA_CONFIG Struct Reference	178
13.72.1 Detailed Description	179
13.72.2 Member Data Documentation	179
13.72.2.1 GnaEnable	179
13.73 GPIO_CONFIG Struct Reference	179
13.73.1 Detailed Description	180
13.73.2 Member Data Documentation	180
13.73.2.1 Direction	180
13.73.2.2 ElectricalConfig	181
13.73.2.3 HostSoftPadOwn	181
13.73.2.4 InterruptConfig	181
13.73.2.5 LockConfig	181
13.73.2.6 OutputState	182
13.73.2.7 PadMode	182
13.73.2.8 PowerConfig	182
13.74 GRAPHICS_DXE_CONFIG Struct Reference	183
13.74.1 Detailed Description	184
13.74.2 Member Data Documentation	184
13.74.2.1 IgdDvmtMemSize	184
13.75 GRAPHICS_PEI_CONFIG Struct Reference	185

13.75.1 Detailed Description	186
13.75.2 Member Data Documentation	187
13.75.2.1 CdClock	187
13.76 GRAPHICS_PEI_PREMEM_CONFIG Struct Reference	187
13.76.1 Detailed Description	189
13.76.2 Member Data Documentation	189
13.76.2.1 InternalGraphics	189
13.77 GUID Struct Reference	190
13.77.1 Detailed Description	190
13.78 HDA_LINK_DMIC Struct Reference	190
13.78.1 Detailed Description	190
13.79 HDA_LINK_HDA Struct Reference	191
13.79.1 Detailed Description	191
13.80 HDA_LINK_SNDW Struct Reference	191
13.80.1 Detailed Description	191
13.81 HDA_LINK_SSP Struct Reference	192
13.81.1 Detailed Description	192
13.82 HDA_VERB_TABLE_HEADER Struct Reference	192
13.82.1 Detailed Description	192
13.83 HDAUDIO_CONFIG Struct Reference	193
13.83.1 Detailed Description	194
13.83.2 Member Data Documentation	194
13.83.2.1 VerbTableEntryNum	194
13.83.2.2 VerbTablePtr	194
13.84 HDAUDIO_DXE_CONFIG Struct Reference	195
13.84.1 Detailed Description	195
13.85 HDAUDIO_PREMEM_CONFIG Struct Reference	196
13.85.1 Detailed Description	197
13.85.2 Member Data Documentation	197
13.85.2.1 AudioLinkDmic	197
13.85.2.2 AudioLinkHda	198
13.85.2.3 AudioLinkSndw	198
13.85.2.4 AudioLinkSsp	198
13.86 HSIO_PARAMETERS Struct Reference	198
13.86.1 Detailed Description	200
13.86.2 Member Data Documentation	200
13.86.2.1 HsioCtrlAdaptOffsetCfg	200
13.87 HYBRID_GRAPHICS_CONFIG Struct Reference	200
13.87.1 Detailed Description	201
13.88 HYBRID_STORAGE_CONFIG Struct Reference	202
13.88.1 Detailed Description	202
13.89 I2C_PIN_MUX Struct Reference	203

13.89.1 Detailed Description	203
13.90 IEH_CONFIG Struct Reference	203
13.90.1 Detailed Description	204
13.91 IOM_AUX_ORI_PAD_CONFIG Struct Reference	204
13.91.1 Detailed Description	204
13.92 IOM_INTERFACE_CONFIG Struct Reference	205
13.92.1 Detailed Description	205
13.93 IPU_PREMEM_CONFIG Struct Reference	205
13.93.1 Detailed Description	206
13.93.2 Member Data Documentation	207
13.93.2.1 ImguCkOutEn	207
13.93.2.2 IpuEnable	207
13.94 IPv4_ADDRESS Struct Reference	207
13.94.1 Detailed Description	207
13.95 IPv6_ADDRESS Struct Reference	208
13.95.1 Detailed Description	208
13.96 ISH_CONFIG Struct Reference	208
13.96.1 Detailed Description	209
13.97 ISH_GP Struct Reference	209
13.97.1 Detailed Description	209
13.98 ISH_GPIO_CONFIG Struct Reference	210
13.98.1 Detailed Description	210
13.98.2 Member Data Documentation	210
13.98.2.1 PadTermination	210
13.98.2.2 PinMux	210
13.99 ISH_I2C Struct Reference	211
13.99.1 Detailed Description	211
13.100 ISH_I2C_PIN_CONFIG Struct Reference	212
13.100.1 Detailed Description	212
13.101 ISH_I3C_CONFIG Struct Reference	213
13.101.1 Detailed Description	213
13.102 ISH_PREMEM_CONFIG Struct Reference	214
13.102.1 Detailed Description	214
13.102.2 Member Data Documentation	214
13.102.2.1 Enable	215
13.103 ISH_SPI Struct Reference	215
13.103.1 Detailed Description	216
13.104 ISH_SPI_PIN_CONFIG Struct Reference	216
13.104.1 Detailed Description	217
13.105 ISH_UART Struct Reference	217
13.105.1 Detailed Description	218
13.106 ISH_UART_PIN_CONFIG Struct Reference	218

13.106.1 Detailed Description	219
13.107 ME_PEI_CONFIG Struct Reference	219
13.107.1 Detailed Description	220
13.107.2 Member Data Documentation	220
13.107.2.1 Heci3Enabled	220
13.107.2.2 MeUnconfigOnRtcClear	221
13.108 ME_PEI_PREMEM_CONFIG Struct Reference	221
13.108.1 Detailed Description	222
13.108.2 Member Data Documentation	223
13.108.2.1 SkipMbpHob	223
13.109 MEMORY_CONFIG_NO_CRC Struct Reference	223
13.109.1 Detailed Description	224
13.109.2 Member Data Documentation	224
13.109.2.1 SerialDebugLevel	225
13.110 MEMORY_CONFIGURATION Struct Reference	225
13.110.1 Detailed Description	236
13.110.2 Member Data Documentation	236
13.110.2.1 ChHashInterleaveBit	236
13.110.2.2 DCC	236
13.110.2.3 ECT	236
13.110.2.4 FirstDimmBitMask	237
13.110.2.5 FirstDimmBitMaskEcc	237
13.110.2.6 FreqLimitMixedConfig_1R1R_16GB	237
13.110.2.7 FreqLimitMixedConfig_1R1R_8GB	237
13.110.2.8 FreqLimitMixedConfig_1R1R_8GB_16GB	238
13.110.2.9 FreqLimitMixedConfig_2R2R	238
13.110.2.10 WRDSUDT	238
13.111 OVERCLOCKING_PREMEM_CONFIG Struct Reference	239
13.111.1 Detailed Description	243
13.111.2 Member Data Documentation	244
13.111.2.1 AtomClusterRatio	244
13.111.2.2 AtomL2MaxOcRatio	244
13.111.2.3 AtomL2VoltageAdaptive	245
13.111.2.4 AtomL2VoltageMode	245
13.111.2.5 AtomL2VoltageOffset	245
13.111.2.6 AtomL2VoltageOverride	245
13.111.2.7 Avx2VoltageScaleFactor	246
13.111.2.8 Avx512VoltageScaleFactor	246
13.111.2.9 CoreMaxOcRatio	246
13.111.2.10 CoreVfConfigScope	247
13.111.2.11 CoreVfPointOffset	247
13.111.2.12 CoreVfPointOffsetMode	247

13.111.2.13 CoreVoltageAdaptive	247
13.111.2.14 CoreVoltageMode	248
13.111.2.15 CoreVoltageOffset	248
13.111.2.16 CoreVoltageOverride	248
13.111.2.17 DisablePerAtomMask	248
13.111.2.18 DisablePerCoreMask	249
13.111.2.19 FastThrottleThreshold	249
13.111.2.20 lalccMax	249
13.111.2.21 OcSupport	249
13.111.2.22 PerCoreHtDisable	250
13.111.2.23 PerCoreVoltageOffset	250
13.111.2.24 PllMaxBandingRatio	250
13.111.2.25 PvdMode	250
13.111.2.26 PvdRatioThreshold	251
13.111.2.27 RingDownBin	251
13.111.2.28 RingMaxOcRatio	251
13.111.2.29 RingVfPointOffset	251
13.111.2.30 RingVfPointOffsetMode	252
13.111.2.31 RingVoltageAdaptive	252
13.111.2.32 RingVoltageMode	252
13.111.2.33 RingVoltageOffset	252
13.111.2.34 RingVoltageOverride	253
13.111.2.35 SaExtraTurboVoltage	253
13.111.2.36 SaVoltageMode	253
13.111.2.37 SaVoltageOverride	253
13.111.2.38 TjMaxOffset	254
13.111.2.39 TvbDownBinsTempThreshold0	254
13.111.2.40 TvbDownBinsTempThreshold1	254
13.111.2.41 TvbRatioClipping	254
13.111.2.42 TvbTempThreshold0	255
13.111.2.43 TvbTempThreshold1	255
13.111.2.44 TvbVoltageOptimization	255
13.112 PCH_DCI_PREMEM_CONFIG Struct Reference	256
13.112.1 Detailed Description	257
13.112.2 Member Data Documentation	257
13.112.2.1 DciClkEnable	257
13.112.2.2 DciDbcMode	257
13.112.2.3 DciEn	258
13.112.2.4 DciUsb3TypecUfpDbg	258
13.112.2.5 KeepEarlyTrace	258
13.113 PCH_DEVICE_INTERRUPT_CONFIG Struct Reference	258
13.113.1 Detailed Description	259

13.114 PCH_DMI_CONFIG Struct Reference	259
13.114.1 Detailed Description	260
13.114.2 Member Data Documentation	260
13.114.2.1 DmiPowerReduction	260
13.115 PCH_DMI_EQ_PARAM Struct Reference	261
13.115.1 Detailed Description	261
13.116 PCH_ESPI_CONFIG Struct Reference	261
13.116.1 Detailed Description	262
13.116.2 Member Data Documentation	262
13.116.2.1 BmeMasterSlaveEnabled	262
13.116.2.2 LgmrEnable	263
13.117 PCH_FIVR_CONFIG Struct Reference	263
13.117.1 Detailed Description	264
13.117.2 Member Data Documentation	264
13.117.2.1 ExtVnnRailSx	264
13.118 PCH_GENERAL_CONFIG Struct Reference	264
13.118.1 Detailed Description	265
13.118.2 Member Data Documentation	265
13.118.2.1 AcpiL6dPmeHandling	265
13.118.2.2 Crid	265
13.118.2.3 LegacyIoLowLatency	266
13.119 PCH_GENERAL_PREMEM_CONFIG Struct Reference	266
13.119.1 Detailed Description	267
13.119.2 Member Data Documentation	267
13.119.2.1 DisableDynamicTccoldHandshake	267
13.119.2.2 GpioOverride	268
13.119.2.3 IoeDebugEn	268
13.119.2.4 PmodeClkEn	268
13.120 PCH_INTERRUPT_CONFIG Struct Reference	269
13.120.1 Detailed Description	270
13.121 PCH_IOAPIC_CONFIG Struct Reference	270
13.121.1 Detailed Description	271
13.121.2 Member Data Documentation	271
13.121.2.1 Enable8254ClockGating	271
13.121.2.2 Enable8254ClockGatingOnS3	272
13.122 PCH_LOCK_DOWN_CONFIG Struct Reference	272
13.122.1 Detailed Description	273
13.122.2 Member Data Documentation	273
13.122.2.1 BiosInterface	273
13.122.2.2 BiosLock	274
13.122.2.3 GlobalSmi	274
13.122.2.4 UnlockGpioPads	274

13.123 PCH_MEMORY_THROTTLING Struct Reference	275
13.123.1 Detailed Description	275
13.123.2 Member Data Documentation	275
13.123.2.1 Enable	275
13.123.2.2 TsGpioPinSetting	276
13.124 PCH_P2SB_CONFIG Struct Reference	276
13.124.1 Detailed Description	277
13.124.2 Member Data Documentation	277
13.124.2.1 SbAccessUnlock	277
13.125 PCH_PCIE_CLOCK Struct Reference	277
13.125.1 Detailed Description	278
13.126 PCH_PCIE_CONFIG Struct Reference	278
13.126.1 Detailed Description	279
13.127 PCH_PCIE_ROOT_PORT_CONFIG Struct Reference	279
13.127.1 Detailed Description	279
13.127.2 Member Data Documentation	279
13.127.2.1 ExtSync	280
13.127.2.2 MvcEnabled	280
13.127.2.3 VppPort	280
13.128 PCH_PCIE_RP_PREMEM_CONFIG Struct Reference	281
13.128.1 Detailed Description	281
13.128.2 Member Data Documentation	282
13.128.2.1 RpEnabledMask	282
13.129 PCH_PM_CONFIG Struct Reference	282
13.129.1 Detailed Description	284
13.129.2 Member Data Documentation	284
13.129.2.1 C10DynamicThresholdAdjustment	284
13.129.2.2 CpuC10GatePinEnable	284
13.129.2.3 DisableDsxAcPresentPulldown	285
13.129.2.4 DisableEnergyReport	285
13.129.2.5 DisableNativePowerButton	285
13.129.2.6 GlobalResetMasksOverride	285
13.129.2.7 LatchEventsC10Exit	286
13.129.2.8 LpmS0ixSubStateEnable	286
13.129.2.9 ModPhySusPgEnable	286
13.129.2.10 OsIdleEnable	286
13.129.2.11 PchPwrCycDur	287
13.129.2.12 PciePIIScc	287
13.129.2.13 PmcDbgMsgEn	287
13.129.2.14 PsOnEnable	288
13.129.2.15 PwrBtnOverridePeriod	288
13.129.2.16 S0ixAutoDemotion	288

13.129.2.17 SlpLanLowDc	288
13.129.2.18 SlpStrchSusUp	289
13.129.2.19 ThermTimerDelay	289
13.129.2.20 Usb2PhySusPgEnable	289
13.130 PCH_SATA_PORT_CONFIG Struct Reference	289
13.130.1 Detailed Description	290
13.130.2 Member Data Documentation	290
13.130.2.1 Enable	291
13.130.2.2 ZpOdd	291
13.131 PCH_SMBUS_PREMEM_CONFIG Struct Reference	291
13.131.1 Detailed Description	292
13.131.2 Member Data Documentation	292
13.131.2.1 Enable	292
13.131.2.2 Header	293
13.131.2.3 SpdWriteDisable	293
13.132 PCH_TRACE_HUB_PREMEM_CONFIG Struct Reference	293
13.132.1 Detailed Description	294
13.133 PCH_WAKE_CONFIG Struct Reference	294
13.133.1 Detailed Description	294
13.133.2 Member Data Documentation	295
13.133.2.1 PmeB0S5Dis	295
13.134 PCH_WDT_PREMEM_CONFIG Struct Reference	295
13.134.1 Detailed Description	296
13.135 PCIE_COMMON_CONFIG Struct Reference	296
13.135.1 Detailed Description	297
13.135.2 Member Data Documentation	297
13.135.2.1 ComplianceTestMode	297
13.135.2.2 EnablePort8xhDecode	297
13.135.2.3 RpFunctionSwap	297
13.136 PCIE_DEVICE_OVERRIDE Struct Reference	298
13.136.1 Detailed Description	298
13.136.2 Member Data Documentation	299
13.136.2.1 ForceLtrOverride	299
13.136.2.2 L1sCommonModeRestoreTime	299
13.136.2.3 L1sTpowerOnScale	299
13.136.2.4 L1sTpowerOnValue	300
13.136.2.5 L1SubstatesCapMask	300
13.136.2.6 L1SubstatesCapOffset	300
13.136.2.7 NonSnoopLatency	300
13.136.2.8 SnoopLatency	301
13.137 PCIE_EQ_PARAM Struct Reference	301
13.137.1 Detailed Description	301

13.138 PCIE_IMR_CONFIG Struct Reference	301
13.138.1 Detailed Description	302
13.139 PCIE_LINK_EQ_PLATFORM_SETTINGS Struct Reference	302
13.139.1 Detailed Description	303
13.139.2 Member Data Documentation	303
13.139.2.1 EqPh23Bypass	303
13.139.2.2 EqPh3Bypass	303
13.139.2.3 LocalTransmitterOverrideEnable	304
13.139.2.4 Ph2LocalTransmitterOverridePreset	304
13.139.2.5 Ph3NumberOfPresetsOrCoefficients	304
13.140 PCIE_PEI_PREMEM_CONFIG Struct Reference	305
13.140.1 Detailed Description	306
13.140.2 Member Data Documentation	306
13.140.2.1 DmiGen3EqPh2Enable	306
13.140.2.2 DmiGen3EqPh3Method	307
13.140.2.3 DmiGen3ProgramStaticEq	307
13.140.2.4 DmiGen3RxCtlePeaking	307
13.140.2.5 DmiMaxLinkSpeed	308
13.140.2.6 InitPcieAspmAfterOprom	308
13.141 PCIE_PREMEM_CONFIG Struct Reference	308
13.141.1 Detailed Description	309
13.142 PCIE_RP_DXE_CONFIG Struct Reference	309
13.142.1 Detailed Description	310
13.142.2 Member Data Documentation	310
13.142.2.1 PcieDeviceOverrideTablePtr	311
13.143 PMC_GLOBAL_RESET_MASK Union Reference	311
13.143.1 Detailed Description	311
13.144 PMC_INTERFACE_CONFIG Struct Reference	311
13.144.1 Detailed Description	312
13.145 PMC_LPM_S0IX_SUB_STATE_EN Union Reference	312
13.145.1 Detailed Description	312
13.145.2 Member Data Documentation	312
13.145.2.1 S0i2p2En	312
13.145.2.2 S0i3p3En	313
13.145.2.3 S0i3p4En	313
13.146 PPM_CUSTOM_RATIO_TABLE Struct Reference	313
13.146.1 Detailed Description	314
13.146.2 Member Data Documentation	314
13.146.2.1 StateRatioMax16	314
13.147 PSF_CONFIG Struct Reference	314
13.147.1 Detailed Description	315
13.147.2 Member Data Documentation	315

13.147.2.1 TccEnable	316
13.148 RST_CONFIG Struct Reference	316
13.148.1 Detailed Description	317
13.149 RST_HARDWARE_REMAPPED_STORAGE_CONFIG Struct Reference	318
13.149.1 Detailed Description	318
13.149.2 Member Data Documentation	318
13.149.2.1 DeviceResetDelay	318
13.149.2.2 Enable	319
13.150 RTC_CONFIG Struct Reference	319
13.150.1 Detailed Description	320
13.150.2 Member Data Documentation	320
13.150.2.1 BiosInterfaceLock	320
13.150.2.2 MemoryLock	320
13.151 SA_ADDRESS_DECODE Struct Reference	321
13.151.1 Detailed Description	321
13.152 SA_FUNCTION_CALLS Struct Reference	321
13.152.1 Detailed Description	323
13.153 SA_MEMORY_DQDQS_MAPPING Struct Reference	324
13.153.1 Detailed Description	324
13.154 SA_MEMORY_FUNCTIONS Struct Reference	324
13.154.1 Detailed Description	325
13.155 SA_MEMORY_RCOMP Struct Reference	325
13.155.1 Detailed Description	325
13.156 SA_MISC_PEI_CONFIG Struct Reference	326
13.156.1 Detailed Description	326
13.157 SA_MISC_PEI_PREMEM_CONFIG Struct Reference	327
13.157.1 Detailed Description	329
13.157.2 Member Data Documentation	329
13.157.2.1 FirstDimmBitMask	329
13.157.2.2 FirstDimmBitMaskEcc	329
13.157.2.3 IcdSize	330
13.157.2.4 ScanExtGfxForLegacyOpRom	330
13.157.2.5 SpdAddressTable	330
13.157.2.6 TsegSize	331
13.158 SA_XDCI_IRQ_INT_CONFIG Struct Reference	331
13.158.1 Detailed Description	331
13.159 SATA_CONFIG Struct Reference	332
13.159.1 Detailed Description	333
13.159.2 Member Data Documentation	333
13.159.2.1 Enable	333
13.159.2.2 EsataSpeedLimit	333
13.159.2.3 RaidDeviceId	333

13.159.2.4 SataMode	334
13.159.2.5 SpeedLimit	334
13.159.2.6 ThermalThrottling	334
13.160 SATA_THERMAL_THROTTLING Struct Reference	334
13.160.1 Detailed Description	335
13.161 SERIAL_IO_CONFIG Struct Reference	336
13.161.1 Detailed Description	336
13.162 SERIAL_IO_I2C_CONFIG Struct Reference	337
13.162.1 Detailed Description	337
13.162.2 Member Data Documentation	337
13.162.2.1 PadTermination	337
13.163 SERIAL_IO_SPI_CONFIG Struct Reference	338
13.163.1 Detailed Description	338
13.164 SERIAL_IO_UART_ATTRIBUTES Struct Reference	339
13.164.1 Detailed Description	339
13.165 SERIAL_IO_UART_CONFIG Struct Reference	339
13.165.1 Detailed Description	340
13.166 SI_CONFIG Struct Reference	340
13.166.1 Detailed Description	342
13.166.2 Member Data Documentation	342
13.166.2.1 CsmFlag	342
13.166.2.2 CustomizedSsid	342
13.166.2.3 CustomizedSvid	342
13.166.2.4 NumberOfSsidTableEntry	343
13.166.2.5 SkipBiosDoneWhenFwUpdate	343
13.166.2.6 SkipSsidProgramming	343
13.166.2.7 SsidTablePtr	343
13.167 SI_PREMEM_CONFIG Struct Reference	344
13.167.1 Detailed Description	345
13.167.2 Member Data Documentation	345
13.167.2.1 PlatformDebugConsent	345
13.167.2.2 SkipOverrideBootModeWhenFwUpdate	345
13.168 SMBIOS_STRUCTURE Struct Reference	346
13.168.1 Detailed Description	346
13.169 SPD_DATA_BUFFER Struct Reference	346
13.169.1 Detailed Description	346
13.170 SPD_OFFSET_TABLE Struct Reference	346
13.170.1 Detailed Description	347
13.171 SPI_PIN_MUX Struct Reference	347
13.171.1 Detailed Description	347
13.172 SVID_SID_VALUE Struct Reference	348
13.172.1 Detailed Description	348

13.173 TCSS_DEVEN_PEI_PREMEM_CONFIG Union Reference	348
13.173.1 Detailed Description	348
13.174 TCSS_IOM_ORI_OVERRIDE Struct Reference	348
13.174.1 Detailed Description	349
13.175 TCSS_IOM_PEI_CONFIG Struct Reference	349
13.175.1 Detailed Description	350
13.176 TCSS_MISC_PEI_CONFIG Struct Reference	350
13.176.1 Detailed Description	351
13.177 TCSS_MISC_PEI_PREMEM_CONFIG Struct Reference	351
13.177.1 Detailed Description	351
13.177.2 Member Data Documentation	351
13.177.2.1 PcieMultipleSegmentEnabled	351
13.177.2.2 Rsvd	352
13.178 TCSS_PCIE_PEI_POLICY Struct Reference	352
13.178.1 Detailed Description	352
13.179 TCSS_PCIE_PORT_POLICY Struct Reference	353
13.179.1 Detailed Description	354
13.180 TCSS_PEI_CONFIG Struct Reference	354
13.180.1 Detailed Description	355
13.181 TCSS_PEI_PREMEM_CONFIG Struct Reference	355
13.181.1 Detailed Description	356
13.182 TCSS_USBTC_PEI_PERMEM_CONFIG Struct Reference	356
13.182.1 Detailed Description	356
13.183 TELEMETRY_PEI_CONFIG Struct Reference	357
13.183.1 Detailed Description	357
13.184 TELEMETRY_PEI_PREMEM_CONFIG Struct Reference	358
13.184.1 Detailed Description	358
13.185 THC_CONFIG Struct Reference	359
13.185.1 Detailed Description	359
13.186 THC_HID_OVER_SPI Struct Reference	360
13.186.1 Detailed Description	361
13.187 THC_PORT Struct Reference	361
13.187.1 Detailed Description	362
13.188 THERMAL_CONFIG Struct Reference	362
13.188.1 Detailed Description	363
13.188.2 Member Data Documentation	363
13.188.2.1 DmiHaAWC	363
13.188.2.2 PchHotLevel	363
13.188.2.3 TTLevels	364
13.189 THERMAL_THROTTLE_LEVELS Struct Reference	364
13.189.1 Detailed Description	364
13.189.2 Member Data Documentation	365

13.189.2.1 PchCrossThrottling	365
13.189.2.2 TTLock	365
13.189.2.3 TTState13Enable	365
13.190 TRACE_HUB_CONFIG Struct Reference	365
13.190.1 Detailed Description	366
13.190.2 Member Data Documentation	366
13.190.2.1 AetEnabled	366
13.190.2.2 EnableMode	366
13.190.2.3 MemReg0Size	367
13.191 TS_GPIO_PIN_SETTING Struct Reference	367
13.191.1 Detailed Description	367
13.191.2 Member Data Documentation	367
13.191.2.1 PmsyncEnable	368
13.192 TSN_CONFIG Struct Reference	368
13.192.1 Detailed Description	369
13.193 UART_PIN_MUX Struct Reference	369
13.193.1 Detailed Description	370
13.194 USB2_PHY_CONFIG Struct Reference	370
13.194.1 Detailed Description	371
13.194.2 Member Data Documentation	371
13.194.2.1 Port	371
13.195 USB2_PHY_PARAMETERS Struct Reference	371
13.195.1 Detailed Description	372
13.196 USB2_PORT_CONFIG Struct Reference	372
13.196.1 Detailed Description	372
13.196.2 Member Data Documentation	372
13.196.2.1 OverCurrentPin	372
13.197 USB3_HSIO_CONFIG Struct Reference	373
13.197.1 Detailed Description	373
13.198 USB3_PORT_CONFIG Struct Reference	374
13.198.1 Detailed Description	374
13.198.2 Member Data Documentation	374
13.198.2.1 OverCurrentPin	374
13.199 USB_CONFIG Struct Reference	375
13.199.1 Detailed Description	376
13.199.2 Member Data Documentation	376
13.199.2.1 LtrOverrideEnable	376
13.199.2.2 OverCurrentEnable	376
13.199.2.3 PdoProgramming	377
13.199.2.4 UaolEnable	377
13.199.2.5 XhciOcLock	377
13.200 VMD_PEI_CONFIG Struct Reference	378

13.200.1 Detailed Description	379
13.200.2 Member Data Documentation	379
13.200.2.1 VmdCfgBarSize	379
13.201 VTD_CONFIG Struct Reference	380
13.201.1 Detailed Description	381
13.201.2 Member Data Documentation	381
13.201.2.1 EnableDmaBuffer	381
13.201.2.2 PreBootDmaMask	381
13.201.2.3 VtdDisable	382
13.202 VTD_DXE_CONFIG Struct Reference	382
13.202.1 Detailed Description	383
13.203 XDCI_CONFIG Struct Reference	383
13.203.1 Detailed Description	383
13.203.2 Member Data Documentation	383
13.203.2.1 Enable	383
14 File Documentation	385
14.1 AdrConfig.h File Reference	385
14.1.1 Detailed Description	386
14.2 AmtConfig.h File Reference	386
14.2.1 Detailed Description	387
14.2.2 Typedef Documentation	387
14.2.2.1 AMT_REPORT_ERROR	388
14.3 Base.h File Reference	388
14.3.1 Detailed Description	392
14.3.2 Macro Definition Documentation	393
14.3.2.1 _BASE_INT_SIZE_OF	393
14.3.2.2 _INT_SIZE_OF	393
14.3.2.3 ABS	393
14.3.2.4 ALIGN_POINTER	394
14.3.2.5 ALIGN_VALUE	394
14.3.2.6 ALIGN_VARIABLE	395
14.3.2.7 ANALYZER_NORETURN	395
14.3.2.8 ANALYZER_UNREACHABLE	395
14.3.2.9 ARRAY_SIZE	396
14.3.2.10 BASE_ARG	396
14.3.2.11 BASE_CR	397
14.3.2.12 ENCODE_ERROR	397
14.3.2.13 ENCODE_WARNING	398
14.3.2.14 FALSE	398
14.3.2.15 MAX	398
14.3.2.16 MIN	399

14.3.2.17 NORETURN	399
14.3.2.18 OFFSET_OF	399
14.3.2.19 RETURN_ADDRESS	400
14.3.2.20 RETURN_BUFFER_TOO_SMALL	400
14.3.2.21 RETURN_ERROR	400
14.3.2.22 RETURNS_TWICE	401
14.3.2.23 SIGNATURE_16	401
14.3.2.24 SIGNATURE_32	402
14.3.2.25 SIGNATURE_64	402
14.3.2.26 STATIC_ASSERT	403
14.3.2.27 TRUE	403
14.3.2.28 UNREACHABLE	403
14.3.2.29 VA_ARG	404
14.3.2.30 VA_COPY	405
14.3.2.31 VA_END	405
14.3.2.32 VA_START	406
14.3.3 Typedef Documentation	406
14.3.3.1 BASE_LIST	406
14.3.3.2 VA_LIST	406
14.4 BiosGuardConfig.h File Reference	407
14.4.1 Detailed Description	408
14.4.2 Typedef Documentation	408
14.4.2.1 PLATFORM_SEND_EC_COMMAND	408
14.5 CnviConfig.h File Reference	409
14.5.1 Detailed Description	409
14.6 ConfigBlock.h File Reference	410
14.6.1 Detailed Description	411
14.7 ConfigBlockLib.h File Reference	411
14.7.1 Detailed Description	412
14.7.2 Function Documentation	412
14.7.2.1 AddConfigBlock()	412
14.7.2.2 CreateConfigBlockTable()	412
14.7.2.3 GetConfigBlock()	413
14.8 CpuConfig.h File Reference	413
14.8.1 Detailed Description	414
14.9 CpuConfigLibPreMemConfig.h File Reference	414
14.9.1 Detailed Description	414
14.10 CpuDmiPreMemConfig.h File Reference	415
14.10.1 Detailed Description	415
14.11 CpuPcieConfig.h File Reference	416
14.11.1 Detailed Description	417
14.11.2 Macro Definition Documentation	418

14.11.2.1 CPU_PCIE_CONFIG_REVISION	418
14.11.3 Enumeration Type Documentation	418
14.11.3.1 CPU_PCIE_EQ_METHOD	418
14.12 CpuPowerMgmtBasicConfig.h File Reference	419
14.12.1 Detailed Description	419
14.13 CpuPowerMgmtCustomConfig.h File Reference	419
14.13.1 Detailed Description	420
14.13.2 Macro Definition Documentation	421
14.13.2.1 MAX_CUSTOM_CTDTP_ENTRIES	421
14.14 CpuPowerMgmtPsysConfig.h File Reference	421
14.14.1 Detailed Description	421
14.15 CpuPowerMgmtTestConfig.h File Reference	422
14.15.1 Detailed Description	422
14.15.2 Enumeration Type Documentation	423
14.15.2.1 CUSTOM_POWER_UNIT	423
14.16 CpuPowerMgmtVrConfig.h File Reference	423
14.16.1 Detailed Description	423
14.16.2 Macro Definition Documentation	424
14.16.2.1 MAX_NUM_VRS	424
14.17 CpuSecurityPreMemConfig.h File Reference	424
14.17.1 Detailed Description	425
14.18 CpuTestConfig.h File Reference	425
14.18.1 Detailed Description	425
14.19 CpuTxtConfig.h File Reference	426
14.19.1 Detailed Description	426
14.20 DciConfig.h File Reference	427
14.20.1 Detailed Description	427
14.21 EspiConfig.h File Reference	427
14.21.1 Detailed Description	428
14.22 FirmwareVersionInfoHob.h File Reference	428
14.22.1 Detailed Description	429
14.23 FivrConfig.h File Reference	429
14.23.1 Detailed Description	430
14.23.2 Macro Definition Documentation	430
14.23.2.1 PCH_FIVR_CONFIG_REVISION	430
14.24 FspErrorInfo.h File Reference	431
14.24.1 Detailed Description	431
14.25 FspFixedPcds.h File Reference	431
14.25.1 Detailed Description	432
14.26 FspInfoHob.h File Reference	432
14.26.1 Detailed Description	432
14.27 FspmArchConfigPpi.h File Reference	433

14.27.1 Detailed Description	433
14.28 GbeConfig.h File Reference	434
14.28.1 Detailed Description	434
14.29 GnaConfig.h File Reference	434
14.29.1 Detailed Description	435
14.30 GpioConfig.h File Reference	435
14.30.1 Detailed Description	436
14.30.2 Enumeration Type Documentation	437
14.30.2.1 GPIO_DIRECTION	437
14.30.2.2 GPIO_ELECTRICAL_CONFIG	437
14.30.2.3 GPIO_HARDWARE_DEFAULT	438
14.30.2.4 GPIO_HOSTSW_OWN	438
14.30.2.5 GPIO_INT_CONFIG	438
14.30.2.6 GPIO_LOCK_CONFIG	439
14.30.2.7 GPIO_OTHER_CONFIG	439
14.30.2.8 GPIO_OUTPUT_STATE	440
14.30.2.9 GPIO_PAD_MODE	440
14.30.2.10 GPIO_RESET_CONFIG	441
14.31 GpioSampleDef.h File Reference	442
14.31.1 Detailed Description	443
14.32 GraphicsConfig.h File Reference	443
14.32.1 Detailed Description	444
14.33 HdAudioConfig.h File Reference	444
14.33.1 Detailed Description	446
14.34 HybridGraphicsConfig.h File Reference	446
14.34.1 Detailed Description	447
14.35 HybridStorageConfig.h File Reference	447
14.35.1 Detailed Description	448
14.36 IehConfig.h File Reference	448
14.36.1 Detailed Description	449
14.37 InterruptConfig.h File Reference	450
14.37.1 Detailed Description	450
14.37.2 Enumeration Type Documentation	451
14.37.2.1 PCH_INT_PIN	451
14.38 IoApicConfig.h File Reference	451
14.38.1 Detailed Description	451
14.39 IpuPreMemConfig.h File Reference	452
14.39.1 Detailed Description	452
14.40 IshConfig.h File Reference	453
14.40.1 Detailed Description	453
14.41 LockDownConfig.h File Reference	454
14.41.1 Detailed Description	454

14.42 MemoryConfig.h File Reference	455
14.42.1 Detailed Description	458
14.42.2 Enumeration Type Documentation	459
14.42.2.1 SA_SPD	459
14.42.2.2 SPD_BOOT_MODE	459
14.43 MePeiConfig.h File Reference	459
14.43.1 Detailed Description	460
14.44 MpServices.h File Reference	460
14.44.1 Detailed Description	461
14.44.2 Typedef Documentation	461
14.44.2.1 EFI_PEI_MP_SERVICES_ENABLEDISABLEAP	462
14.44.2.2 EFI_PEI_MP_SERVICES_GET_NUMBER_OF_PROCESSORS	463
14.44.2.3 EFI_PEI_MP_SERVICES_GET_PROCESSOR_INFO	464
14.44.2.4 EFI_PEI_MP_SERVICES_STARTUP_ALL_APS	464
14.44.2.5 EFI_PEI_MP_SERVICES_STARTUP_THIS_AP	465
14.44.2.6 EFI_PEI_MP_SERVICES_SWITCH_BSP	466
14.44.2.7 EFI_PEI_MP_SERVICES_WHOAMI	466
14.45 OverclockingConfig.h File Reference	467
14.45.1 Detailed Description	467
14.46 P2sbConfig.h File Reference	468
14.46.1 Detailed Description	468
14.47 PchDmiConfig.h File Reference	469
14.47.1 Detailed Description	469
14.48 PchGeneralConfig.h File Reference	469
14.48.1 Detailed Description	470
14.48.2 Enumeration Type Documentation	470
14.48.2.1 PCH_RESERVED_PAGE_ROUTE	470
14.49 PchPcieRpConfig.h File Reference	471
14.49.1 Detailed Description	472
14.49.2 Enumeration Type Documentation	472
14.49.2.1 PCH_PCIE_CLOCK_USAGE	472
14.50 PcieConfig.h File Reference	473
14.50.1 Detailed Description	474
14.50.2 Enumeration Type Documentation	474
14.50.2.1 PCIE_FORM_FACTOR	475
14.50.2.2 PCIE_LINK_EQ_METHOD	475
14.50.2.3 PCIE_LINK_EQ_MODE	475
14.51 PciePreMemConfig.h File Reference	475
14.51.1 Detailed Description	476
14.52 PeiIbtConfig.h File Reference	476
14.52.1 Detailed Description	477
14.52.2 Typedef Documentation	477

14.52.2.1 PEI_ITBT_CONFIG	478
14.53 PeiTbtGenericStructure.h File Reference	478
14.53.1 Detailed Description	479
14.54 PeiPreMemSiDefaultPolicy.h File Reference	479
14.54.1 Detailed Description	480
14.55 PeiSiDefaultPolicy.h File Reference	480
14.55.1 Detailed Description	481
14.56 PiHob.h File Reference	481
14.56.1 Detailed Description	482
14.57 PiMultiPhase.h File Reference	483
14.57.1 Detailed Description	484
14.57.2 Macro Definition Documentation	484
14.57.2.1 DXE_ERROR	484
14.57.2.2 EFI_AUTH_STATUS_PLATFORM_OVERRIDE	485
14.57.2.3 PI_ENCODE_ERROR	485
14.57.2.4 PI_ENCODE_WARNING	485
14.57.3 Typedef Documentation	485
14.57.3.1 EFI_AP_PROCEDURE	485
14.57.3.2 EFI_AP_PROCEDURE2	486
14.58 PmConfig.h File Reference	486
14.58.1 Detailed Description	487
14.58.2 Enumeration Type Documentation	488
14.58.2.1 PCH_SLP_S4_MIN_ASSERT	488
14.59 PsfConfig.h File Reference	488
14.59.1 Detailed Description	489
14.60 RstConfig.h File Reference	490
14.60.1 Detailed Description	490
14.61 RtcConfig.h File Reference	491
14.61.1 Detailed Description	491
14.62 SaMiscPeiConfig.h File Reference	492
14.62.1 Detailed Description	492
14.63 SaMiscPeiPreMemConfig.h File Reference	492
14.63.1 Detailed Description	493
14.64 SataConfig.h File Reference	493
14.64.1 Detailed Description	494
14.65 SerialIoConfig.h File Reference	494
14.65.1 Detailed Description	495
14.66 SerialIoDevices.h File Reference	496
14.66.1 Detailed Description	497
14.66.2 Enumeration Type Documentation	497
14.66.2.1 SERIAL_IO_I2C_MODE	498
14.66.2.2 SERIAL_IO_SPI_MODE	498

14.66.2.3 SERIAL_IO_UART_MODE	499
14.66.2.4 SERIAL_IO_UART_PG	500
14.67 SiConfig.h File Reference	500
14.67.1 Detailed Description	500
14.68 SiPolicyStruct.h File Reference	501
14.68.1 Detailed Description	502
14.68.2 Macro Definition Documentation	502
14.68.2.1 SI_POLICY_REVISION	503
14.68.2.2 SI_PREMEM_POLICY_REVISION	503
14.69 SiPreMemConfig.h File Reference	503
14.69.1 Detailed Description	504
14.70 SmbusConfig.h File Reference	504
14.70.1 Detailed Description	505
14.71 TcssPeiConfig.h File Reference	505
14.71.1 Detailed Description	506
14.72 TcssPeiPreMemConfig.h File Reference	507
14.72.1 Detailed Description	507
14.73 TelemetryPeiConfig.h File Reference	508
14.73.1 Detailed Description	509
14.74 ThcConfig.h File Reference	509
14.74.1 Detailed Description	510
14.74.2 Enumeration Type Documentation	510
14.74.2.1 THC_MODE	510
14.74.2.2 THC_PORT_ASSIGNMENT	511
14.75 ThermalConfig.h File Reference	511
14.75.1 Detailed Description	511
14.76 TraceHubConfig.h File Reference	512
14.76.1 Detailed Description	513
14.77 TsnConfig.h File Reference	513
14.77.1 Detailed Description	513
14.78 UefiBaseType.h File Reference	514
14.78.1 Detailed Description	516
14.78.2 Macro Definition Documentation	516
14.78.2.1 EFI_PAGES_TO_SIZE	516
14.78.2.2 EFI_SIZE_TO_PAGES	517
14.78.3 Typedef Documentation	517
14.78.3.1 EFI_IPv4_ADDRESS	517
14.78.3.2 EFI_IPv6_ADDRESS	517
14.79 Usb2PhyConfig.h File Reference	518
14.79.1 Detailed Description	518
14.80 Usb3HsioConfig.h File Reference	519
14.80.1 Detailed Description	519

14.81 UsbConfig.h File Reference	520
14.81.1 Detailed Description	520
14.82 VmdPeiConfig.h File Reference	521
14.82.1 Detailed Description	522
14.83 VtdConfig.h File Reference	522
14.83.1 Detailed Description	523
14.83.2 Typedef Documentation	523
14.83.2.1 VTD_ENABLE_DMA_BUFFER	523
14.84 WatchDogConfig.h File Reference	524
14.84.1 Detailed Description	524
Index	525

Chapter 1

Introduction

1.1 Purpose

The purpose of this document is to describe the steps required to integrate the Intel® Firmware Support Package (FSP) into a boot loader solution. It supports **AlderLake** platforms with **AlderLake** processor and **AlderLake** Platform Controller Hub (PCH).

1.2 Intended Audience

This document is targeted at all platform and system developers who need to consume FSP binaries in their boot loader solutions. This includes, but is not limited to: system BIOS developers, boot loader developers, system integrators, as well as end users.

1.3 Related Documents

- *Platform Initialization (PI) Specification v1.7* located at <http://www.uefi.org/specifications>
- *Intel® Firmware Support Package: External Architecture Specification (EAS) v2.1* located at <https://cdrdv2.intel.com/v1/dl/getContent/611786>
- *Boot Setting File Specification (BSF) v1.0* https://firmware.intel.com/sites/default/files/BSF_1_0.pdf
- *Binary Configuration Tool for Intel® Firmware Support Package* available at <http://www.intel.com/fsp>

1.4 Acronyms and Terminology

Acronym	Definition
BCT	Binary Configuration Tool
BDSM	Base Data Of Stolen Memory
BSF	Boot Setting File
BSP	Boot Strap Processor

Acronym	Definition
BWG	BIOS Writer's Guide
CAR	Cache As Ram
CRB	Customer Reference Board
DPR	DMA Protected Range
FIT	Firmware Interface Table
FSP	Firmware Support Package
FSP API	Firmware Support Package Interface
FW	Firmware
GTT	Graphics Translation Table
IED	Intel Enhanced Debug
IFWI	Integrated Firmware Image
IOT	Internal Observation Trace
MRC	Memory Reference Code (Memory Init code encapsulated by FSP-M)
MOT	Memory Observation Trace
PCH	Platform Controller Hub
PMC	Power Management Controller
PMRR	Protected Memory Range Reporting
REMAP	Remapped Memory Area
RVP	Reference and Validation Platform
SBSP	System BSP
SMI	System Management Interrupt
SMM	System Management Mode
SMRAM	System Management Mode RAM
SPI	Serial Peripheral Interface
TOLUD	Top of Low Usable Memory
TOUUD	Top of Upper Usable Memory
TSEG	Memory Reserved at the Top of Memory to be used as SMRAM
UPD	Updatable Product Data

Chapter 2

Overview

2.1 Technical Overview

The *Intel® Firmware Support Package (FSP)* provides chipset and processor initialization in a format that can easily be incorporated into many existing boot loaders.

The FSP will perform the necessary initialization steps as documented in the BWG including initialization of the CPU, memory controller, chipset and certain bus interfaces, if necessary.

FSP is not a stand-alone boot loader; therefore it needs to be integrated into a host boot loader to carry out other boot loader functions, such as: initializing non-Intel components, conducting bus enumeration, and discovering devices in the system and all industry standard initialization.

The FSP binary can be integrated easily into many different boot loaders, such as Coreboot, EDKII MinPlatform, Intel® Slim Bootloader, etc. and also into the embedded OS directly.

Below are some required steps for the integration:

- **Customizing** The static FSP configuration parameters are part of the FSP binary and can be customized by tools provided by Intel. This step is optional as configuration data may also be provided at runtime.
- **Rebasing** The FSP is not Position Independent Code (PIC) and the whole FSP has to be rebased if it is placed at a location which is different from the preferred address during build process.
- **Placing** Once the FSP binary is ready for integration, the boot loader build process needs to be modified to place this FSP binary at the specific rebasing location identified above.
- **Interfacing** The boot loader needs to add code to setup the operating environment for the FSP, call the FSP with correct parameters and parse the FSP output to retrieve the necessary information returned by the FSP.

2.2 FSP Distribution Package

- The FSP distribution package contains the following:
 - FSP Binary
 - FSP Integration Guide
 - BSF Configuration File
 - Data Structure Header Files
- The FSP configuration utility called BCT is available as a separate package. It can be downloaded from link mentioned in Section 1.3.

2.2.1 Package Layout

- **Docs (Auto generated)**
 - FSP_Integration_Guide.pdf
 - FSP_Integration_Guide.chm
- **Include**
 - FsptUpd.h, FspmUpd.h and FspsUpd.h (FSP UPD structure and related definitions)
 - [GpioSampleDef.h](#) (Sample enum definitions for GPIO table)
- FspBinPkg.dec (EDKII declaration file for package)
- Fsp.bsf (BSF file for configuring the data using BCT tool)
- Fsp.fd (FSP Binary)

Chapter 3

FSP Integration

3.1 Assumptions Used in this Document

The FSP for this platform is built with a preferred base address given by [PcdFspAreaBaseAddress](#) and so the reference code provided in the document assumes that the FSP is placed at this base address during the final boot loader build. Users may rebase the FSP binary at a different location with Intel's Binary Configuration Tool (BCT) or SplitFspBin.py before integrating to the boot loader.

For other assumptions and conventions, please refer to Chapter 8 and 9 of the FSP External Architecture Specification version 2.2.

3.2 Boot Flow

Please refer Chapter 7 in the FSP External Architecture Specification version 2.2 for Boot flow chart. The FSP for this platform supports dispatch mode, see Chapter 7 and 9 of the FSP External Architecture Specification version 2.2 for a description of dispatch mode.

3.3 FSP INFO Header

The FSP has an Information Header that provides critical information that is required by the bootloader to successfully interface with the FSP. The structure of the FSP Information Header is documented in the FSP External Architecture Specification version 2.2 with a HeaderRevision of 5.

3.4 FSP Image ID and Revision

FSP information header contains an Image ID field and an Image Revision field that provide the identification and revision information of the FSP binary. It is important to verify these fields while integrating the FSP as API parameters could change over different FSP IDs and revisions. All the FSP FV segments(FSP-T, FSP-M and FSP-P-S) must have same FSP Image ID and revision number, using FV segments with different revision numbers in a single FSP image is not valid. The FSP API parameters documented in this integration guide are applicable for the Image ID and Revision specified as below.

The FSP ImageId string in the FSP information header is given by [PcdFspImageIdString](#) and the ImageRevision field is given by [SiliconInitVersionMajor|Minor|FspVersionRevision|FspVersionBuild](#) (Ex:0x0A001460).

3.5 FSP Global Data

FSP uses some amount of TempRam area to store FSP global data which contains some critical data like pointers to FSP information headers and UPD configuration regions, FSP/Bootloader stack pointers required for stack switching etc. HPET Timer register(2) [PcdGlobalDataPointerAddress](#) is reserved to store address of this global data, and hence boot loader should not use this register for any other purpose. If TempRAM initialization is done by boot loader, then HPET has to be initialized to the base so that access to the register will work fine.

3.6 Memory Map

Below diagram represents the memory map allocated by FSP including the FSP specific regions.

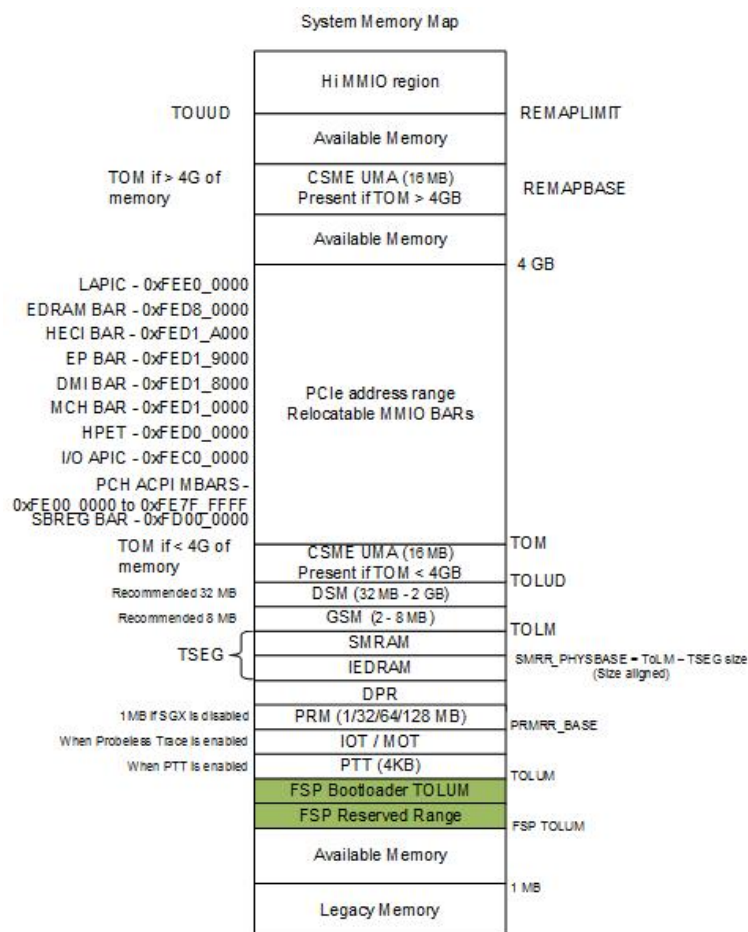


Figure 3.1 System Memory Map

3.7 Optional modules

For products using FSP in API mode, having SPI size constraints, the following modules can be safely removed to help save additional space. These modules are included in FSP binary primarily when FSP is used in dispatch mode:

- PchInitDxeFsp

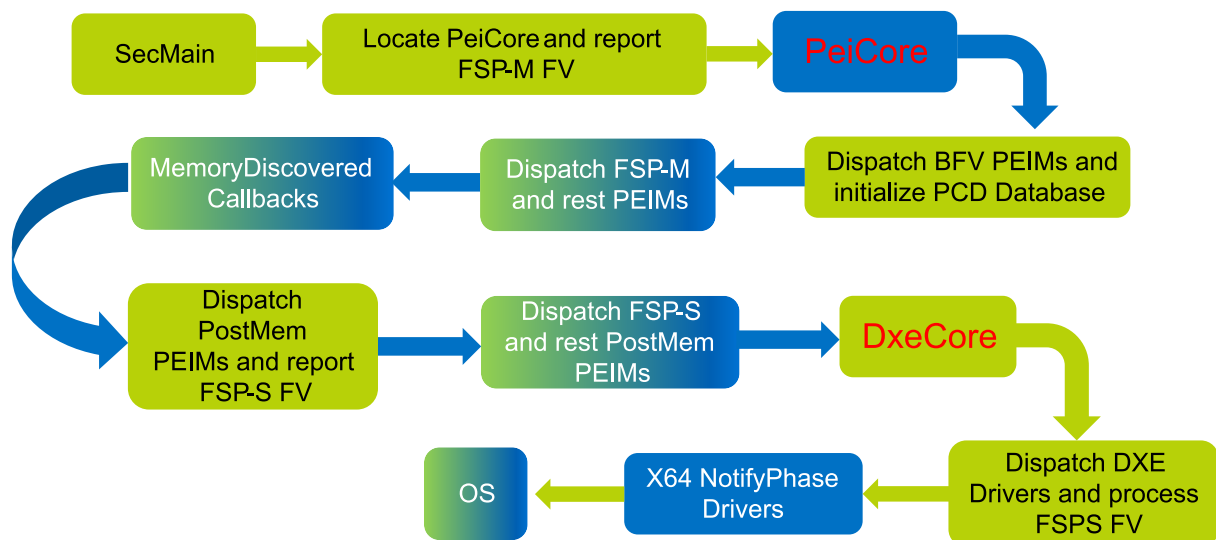
- HecInitDxeFsp
- SalnitDxeFsp
- FspS3Notify
- FspEndOfPei2Peim
- CpuMpPei Please notice that bootloader has to provide CpuMpPpi to FSP via FSP-S UPD if CpuMpPei is removed from FSP.

Chapter 4

FSP Dispatch Mode

Overview

The FSP for this platform supports dispatch mode. Support for dispatch mode can be detected by checking if `FSP_INFO_HEADER->ImageAttribute[BIT1] == 1`. Dispatch mode is intended to enable FSP to integrate well in to UEFI bootloader implementations. Dispatch mode implements a boot flow that is as close to a standard UEFI boot flow as possible. In dispatch mode, the FSP is exposed as standard Firmware Volumes (FVs) directly to the bootloader. The PEIMs in these FVs are executed in the same PEI environment as the boot loader. In dispatch mode, the PPI database, PCD database, and HOB list are shared between the boot loader and the FSP.



Blue blocks are from the FSP binary and green blocks are from the bootloader. Blocks with mixed colors indicate that both bootloader and FSP modules are dispatched during that phase of the boot flow. In dispatch mode, the `NotifyPhase()` API is not used. Instead, FSP-S contains DXE drivers that implement the native callbacks on equivalent events for each of the `NotifyPhase()` invocations.

In dispatch mode, the the PPI database and PCD database are used for providing policy data from the bootloader to FSP. Because these mechanisms provide a great deal of flexibility, dispatch mode does not constrain the method for passing policy data as strongly as API mode. The following sections describe the dispatch mode policy initialization flow used specifically for this platform.

Note

The function names and PEIMs used by the UEFI PI boot loader are implementation dependent and may vary from those shown here. The function names and PEIMs used by MinPlatform are provided as an example.

Execution of FSP-M begins after `FspmWrapperInit()` in `IntelFsp2WrapperPkg/FspmWrapperPeim/FspmWrapperPeim.c` installs a `EFI_PEI_FIRMWARE_VOLUME_INFO_PPI` instance for FSP-M. This causes the PEI foundation to become aware of the FV(s) in FSP-M, which schedules all PEIMs in FSP-M for dispatch by PEI. This results in the `SilnitPreMem` PEIM in FSP-M being executed.

One of the actions performed by the `SilnitPreMem` entry-point is calling the `SilnitPrePolicy()` function, which installs the `PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI`. The DEPEX for `MinPlatformPkg/PlatformInit/SiliconPolicyPei/SiliconPolicyPeiPreMem.inf` requires `PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI` to exist in the PPI database before `SiliconPolicyPeiPreMem` can run. While `SiliconPolicyPeiPreMem.inf` does not directly add `PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI` to the DEPEX, `SiliconPolicyPeiPreMem.inf` does statically link against `AlderlakeSiliconPkg/Library/PeiSiliconPolicyInitLib/PeiPreMemSiliconPolicyInitLib.inf`, which adds `PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI` to the DEPEX. After the `SilnitPreMem` entry-point completes the dependency will be satisfied, making `SiliconPolicyPeiPreMem` eligible for dispatch.

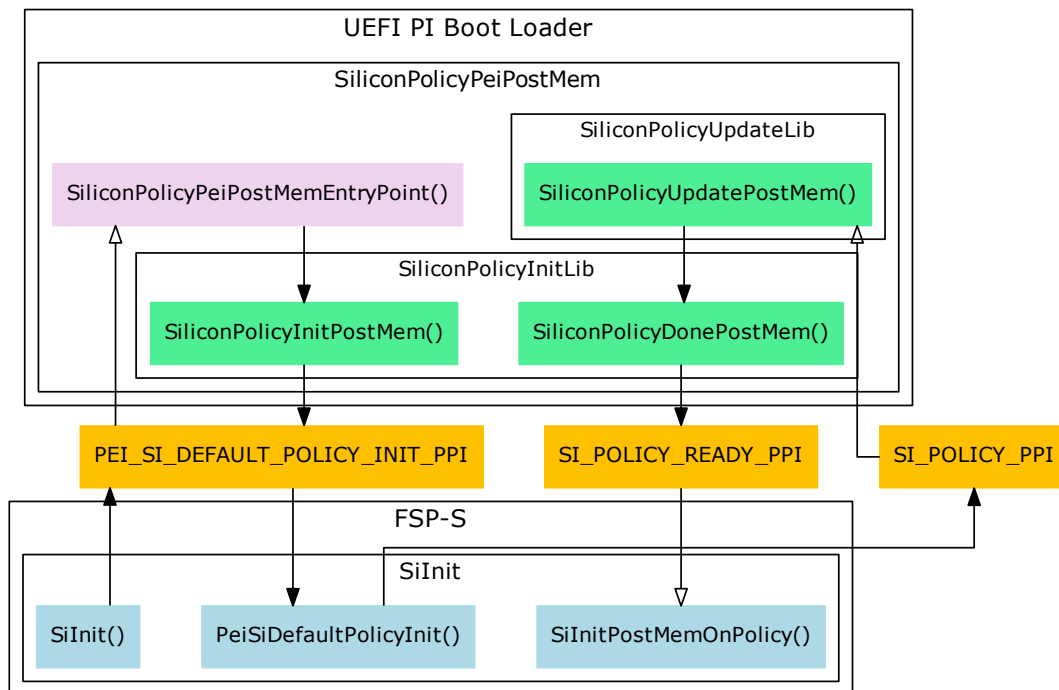
Next `SiliconPolicyPeiPreMem` PEIM will run. `SiliconPolicyPeiPreMemEntryPoint()` in `MinPlatformPkg/PlatformInit/SiliconPolicyPei/SiliconPolicyPeiPreMem.c` will call `SiliconPolicyInitPreMem()` in `AlderlakeSiliconPkg/Library/PeiSiliconPolicyInitLib/PeiPolicyInitPreMem.c`. This function will locate the instance of `PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI` previously created by `SilnitPreMem` and use it to call `PeiPreMemSiDefaultPolicyInit()` in `SilnitPreMem`. `PeiPreMemSiDefaultPolicyInit()` will construct `SI_PREMEM_POLICY_PPI`, which contains all the policy data needed by FSP-M. `SiliconPolicyInitPreMem()` will then locate `SI_PREMEM_POLICY_PPI` and return it to `SiliconPolicyPeiPreMemEntryPoint()`.

Next, `SiliconPolicyPeiPreMemEntryPoint()` takes the pointer to the policy data returned by `SiliconPolicyInitPreMem()` and passes it to `SiliconPolicyUpdatePreMem()`. `SiliconPolicyUpdatePreMem()` is part of `SiliconPolicyUpdateLib`, which is statically linked to `SiliconPolicyPeiPreMem`. `SiliconPolicyUpdateLib` is special since it is only piece of motherboard specific code in this flow. An example of `SiliconPolicyUpdatePreMem()` is seen in `AlderlakeOpenBoardPkg/AlderlakeMRvp/Policy/Library/PeiSiliconPolicyUpdateLib/PeiSiliconPolicyUpdateLib.c`. `SiliconPolicyUpdatePreMem()` uses a read-modify-write operation to apply any motherboard specific settings to the policy data while leaving the default values intact when appropriate.

After `SiliconPolicyUpdatePreMem()` applies any motherboard specific updates, `SiliconPolicyPeiPreMemEntryPoint()` calls `SiliconPolicyDonePreMem()` in `AlderlakeSiliconPkg/Library/PeiSiliconPolicyInitLib/PeiPolicyInitPreMem.c`. `SiliconPolicyDonePreMem()` dumps the policy data to the debug log (on a DEBUG build of MinPlatform only) and installs the `SI_PREMEM_POLICY_READY_PPI`. `SI_PREMEM_POLICY_READY_PPI` tells FSP-M that everything is ready and that it can run the SOC init code now. Installing `SI_PREMEM_POLICY_READY_PPI` causes the PEI Foundation to signal a `PeiServices->NotifyPpi()` callback previously set up by `SilnitPreMem`. This callback invokes `SilnitPreMemOnPolicyReady()` in `SilnitPreMem`, which runs MRC and all the other SOC init code in FSP-M.

4.1.1 FSP-S Policy Initialization Flow

Policy Initialization for FSP-S follows essentially the same flow as FSP-M. The primary difference is function and PPI names do not include the "PreMem" qualifier.



4.2 Config Blocks

Overview

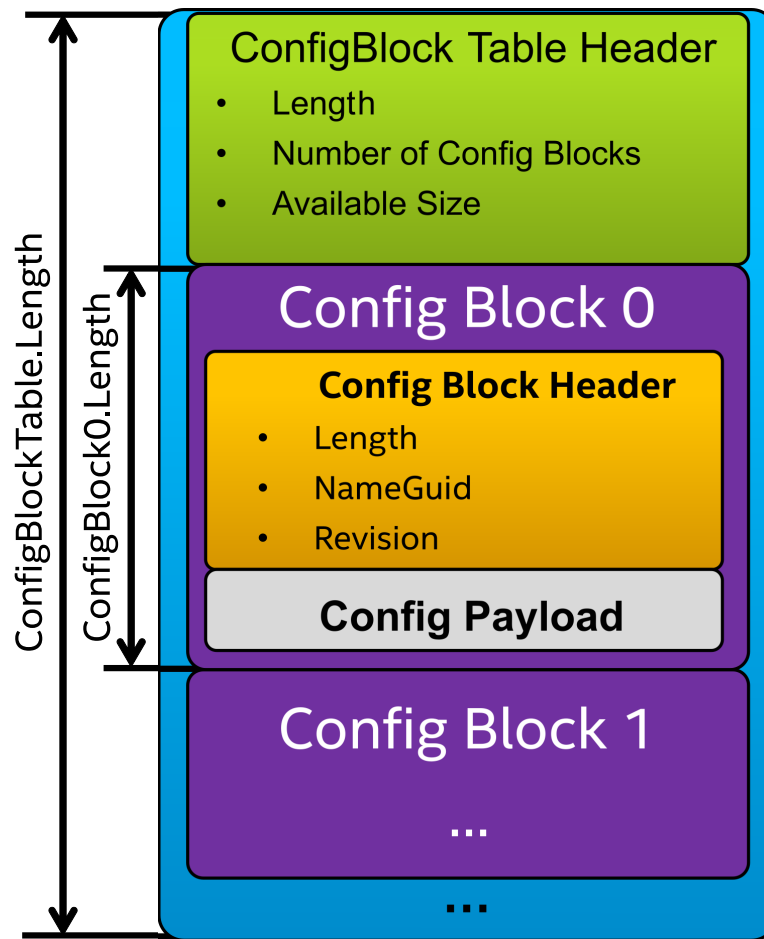
Config Blocks are a binary data serialization format with several unique properties that are useful for firmware implementations.

The serialization format is simple to implement in the C language, minimizing the code needed to implement it; this minimizes the size of the FSP binary. Second, the format is binary backwards compatible; enabling the bootloader to use an older version of the data structure at compile time and retain compatibility with both new and old FSP binaries. Third, the format is mutable in memory without requiring the data to be re-serialized. Finally, the format is position independent with no need for a base address or a rebase operation. This allows the data structure to be moved from pre-memory to post-memory without applying any fixups.

Data in [SI_PREMEM_POLICY_PPI](#) and [SI_POLICY_PPI](#) are stored using the Config Block format.

4.2.1 Config Block Data Format

In essence, Config Blocks are multiple C structures that are stored in a single continuous memory range using run length encoding.



The overall structure is termed a Config Block Table. The Config Block Table starts with a [CONFIG_BLOCK_TABLE_HEADER](#) structure. This header indicates the number of Config Blocks in the Config Block Table, the total size of the table, and the amount of free space remaining in the table. The table may be over-provisioned, which allows additional Config Blocks to be added after the table's initial creation.

The size of all Config Blocks must be an even multiple of 32-bits as all Config Blocks are DWORD aligned. When enumerating a Config Block Table one may assume that all Config Blocks are packed back-to-back, hence the enumeration algorithm can use the length of each config block to find the next config block in the table.

Each Config Block has a [GUID](#) that identifies which C structure the Config Block contains. All Config Blocks start with a standardized [CONFIG_BLOCK_HEADER](#), which allows all the structures in the table to be enumerated even if the format of the data payload inside each Config Block is unknown.

```
typedef struct _CONFIG_BLOCK_HEADER {
    EFI_HOB_GUID_TYPE GuidHob;           ///< Offset 0-23 GUID extension HOB header
    UINT8      Revision;                 ///< Offset 24 Revision of this config block
    UINT8      Attributes;               ///< Offset 25 The main revision for config block
    UINT8      Reserved[2];              ///< Offset 26-27 Reserved for future use
} CONFIG_BLOCK_HEADER;
```

The [CONFIG_BLOCK_HEADER](#) uses the header of a [GUID](#) HOB to store its length and [GUID](#). This makes it easy to copy Config Blocks to the HOB list. Copying Config Blocks to the HOB list is a common use case. Often the ACPI code uses policies that were initially supplied by a Config Block. Copying the Config Block to the HOB list makes it easy for DXE phase to copy any needed policies to the ACPI global NVS memory later.

The Revision field assists with binary backward compatibility. Whenever a new field is added to a Config Block, the new field is added to the bottom of the Config Block and the Revision is incremented by 1. Thus if a bootloader wishes to retain compatibility with new and old versions of the FSP binary, it may do so by either by only using fields that are present in Revision 1 of the Config Block or by checking the Revision number before modifying fields added since Revision 1.

The `CONFIG_BLOCK_TABLE_HEADER` builds upon the `CONFIG_BLOCK_HEADER`:

```
typedef struct _CONFIG_BLOCK_TABLE_STRUCT {
    CONFIG_BLOCK_HEADER    Header;          ///< Offset 0-27  GUID number for main entry of config
    block
    UINT8                  Rsvd0[2];        ///< Offset 28-29 Reserved for future use
    UINT16                  NumberOfBlocks;  ///< Offset 30-31 Number of config blocks (N)
    UINT32                  AvailableSize;   ///< Offset 32-35 Current config block table size
}
///
/// Individual Config Block Structures are added starting here
///
} CONFIG_BLOCK_TABLE_HEADER;
```

4.2.2 Config Block Library APIs

To assist in the creation and parsing of Config Blocks, Intel has provided an open source Config Block library in `IntelSiliconPkg/Library/BaseConfigBlockLib`. This library provides the following functions:

- `GetConfigBlock()`
- `AddConfigBlock()`
- `CreateConfigBlockTable()`

In most cases, bootloaders will only need to call `GetConfigBlock()`.

4.2.3 Config Blocks used by FSP-M

On **AlderLake** platforms `SI_PREMEM_POLICY_PPI` contains a Config Block Table. `PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI` will construct this Config Block Table with all the below Config Blocks pre-populated. Additionally, all the Config Blocks will be initialized with default policy values as described in section 4.1.

`SI_PREMEM_POLICY_PPI` contains the following Config Blocks:

Todo This list is from TigerLake, needs to be updated for AlderLake

- `SI_PREMEM_CONFIG`
- `HOST_BRIDGE_PREMEM_CONFIG`
- `CPU_CONFIG_LIB_PREMEM_CONFIG`
- `CPU_DMI_PREMEM_CONFIG`
- `CPU_PCIE_RP_PREMEM_CONFIG`
- `CPU_SECURITY_PREMEM_CONFIG`
- `CPU_TRACE_HUB_PREMEM_CONFIG`
- `CPU_TXT_PREMEM_CONFIG`
- `GRAPHICS_PEI_PREMEM_CONFIG`
- `HDAUDIO_PREMEM_CONFIG`
- `HYBRID_GRAPHICS_CONFIG`
- `IPU_PREMEM_CONFIG`

- [ISH_PREMEM_CONFIG](#)
- [ME_PEI_PREMEM_CONFIG](#)
- [MEMORY_CONFIG_NO_CRC](#)
- [MEMORY_CONFIGURATION](#)
- [OVERCLOCKING_PREMEM_CONFIG](#)
- [PCH_DCI_PREMEM_CONFIG](#)
- [PCH_GENERAL_PREMEM_CONFIG](#)
- [PCH_HSIO_PCIE_PREMEM_CONFIG](#)
- [PCH_HSIO_SATA_PREMEM_CONFIG](#)
- [PCH_LPC_PREMEM_CONFIG](#)
- [PCH_PCIE_RP_PREMEM_CONFIG](#)
- [PCH_SMBUS_PREMEM_CONFIG](#)
- [PCH_WDT_PREMEM_CONFIG](#)
- [PCIE_PEI_PREMEM_CONFIG](#)
- [PCIE_PREMEM_CONFIG](#)
- [PRAM_PREMEM_CONFIG](#)
- [SA_MISC_PEI_PREMEM_CONFIG](#)
- [TCSS_PEI_PREMEM_CONFIG](#)
- [TELEMETRY_PEI_PREMEM_CONFIG](#)
- [TWOLM_PREMEM_CONFIG](#)
- [VTD_CONFIG](#)

4.2.4 Config Blocks used by FSP-S

On **AlderLake** platforms [SI_POLICY_PPI](#) contains a Config Block Table. [PEI_SI_DEFAULT_POLICY_INIT_PPI](#) will construct this Config Block Table with all the below Config Blocks pre-populated. Additionally, all the Config Blocks will be initialized with default policy values as described in section 4.1.

[SI_POLICY_PPI](#) contains the following Config Blocks:

Todo This list is from TigerLake, needs to be updated for AlderLake

- [SI_CONFIG](#)
- [HOST_BRIDGE_PEI_CONFIG](#)
- [ADR_CONFIG](#)
- [AMT_PEI_CONFIG](#)
- [BIOS_GUARD_CONFIG](#)
- [CNVI_CONFIG](#)

- CPU_CONFIG
- CPU_PCIE_CONFIG
- CPU_PID_TEST_CONFIG
- CPU_POWER_MGMT_BASIC_CONFIG
- CPU_POWER_MGMT_CUSTOM_CONFIG
- CPU_POWER_MGMT_PSYS_CONFIG
- CPU_POWER_MGMT_TEST_CONFIG
- CPU_POWER_MGMT_VR_CONFIG
- CPU_TEST_CONFIG
- GBE_CONFIG
- GNA_CONFIG
- GRAPHICS_PEI_CONFIG
- HDAUDIO_CONFIG
- HYBRID_STORAGE_CONFIG
- IEH_CONFIG
- ISH_CONFIG
- ME_PEI_CONFIG
- PCH_DMI_CONFIG
- PCH_ESPI_CONFIG
- PCH_FIVR_CONFIG
- PCH_FLASH_PROTECTION_CONFIG
- PCH_GENERAL_CONFIG
- PCH_HSIO_CONFIG
- PCH_INTERRUPT_CONFIG
- PCH_IOAPIC_CONFIG
- PCH_LOCK_DOWN_CONFIG
- PCH_P2SB_CONFIG
- PCH_PCIE_CONFIG
- PCH_PM_CONFIG
- PEI_ITBT_CONFIG
- PSF_CONFIG
- RST_CONFIG
- RTC_CONFIG
- SA_MISC_PEI_CONFIG
- SATA_CONFIG
- SERIAL_IO_CONFIG

- [TCSS_PEI_CONFIG](#)
- [TELEMETRY_PEI_CONFIG](#)
- [THC_CONFIG](#)
- [THERMAL_CONFIG](#)
- [TSN_CONFIG](#)
- [USB_CONFIG](#)
- [USB2_PHY_CONFIG](#)
- [USB3_HSIO_CONFIG](#)
- [VMD_PEI_CONFIG](#)

4.3 FSP Error Information

In the case of a fatal error occurring during the execution of the FSP, it may not be possible for the FSP to continue.

If a fatal error that prevents the successful completion of the FSP occurs, the FSP may use `FSP_ERROR_INFO` to report this error to the bootloader. During PEI phase, `(*PeiServices)->ReportStatusCode()` shall be used to transmit this error notification to the bootloader. During DXE phase, `EFI_STATUS_CODE_PROTOCOL.ReportStatusCode()` shall be used to transmit this error notification to the bootloader. The bootloader must ensure that `ReportStatusCode()` services are available before FSP-M begins execution.

```
typedef struct {
    EFI_STATUS_CODE_DATA    DataHeader;
    EFI_GUID                ErrorType;
    EFI_STATUS              Status;
} FSP_ERROR_INFO;
```

`FSP_ERROR_INFO` is provided as the optional `EFI_STATUS_CODE_DATA` parameter to `ReportStatusCode()`. `EFI_STATUS_CODE_DATA` provides a `CallerId` [GUID](#), this `CallerId` combined with the `ErrorType` [GUID](#) describes the error to the bootloader. The FSP for this platform implements the following `CallerId` GUIDs:

CallerId	Description
{0x1f4dc7e9, 0x26ca, 0x4336, {0x8c, 0xe3, 0x39, 0x31, 0x03, 0xb5, 0xf3, 0xd7}}	ME
{0x98230916, 0xe632, 0x49ff, {0x81, 0x81, 0x55, 0xce, 0xe5, 0x10, 0x36, 0x89}}	System Agent
{0x5a47c211, 0x642f, 0x4f92, {0x9c, 0xb3, 0x7f, 0xeb, 0x93, 0xda, 0xdd, 0xba}}	MRC

If the `CallerId` parameter is not a [GUID](#) in the table above, then it should be the [GUID](#) that identifies the PEIM or DXE driver which was executing at the time of the error.

The following `ErrorType` GUIDs are implemented:

ErrorType	Description
{0x948585c4, 0x76a4, 0x45bb, {0xbe, 0x6c, 0x39, 0x61, 0xc3, 0xab, 0xde, 0x15}}	ME EOP failure
{0x8106a5cc, 0x30ba, 0x41cf, {0xa1, 0x78, 0x63, 0x38, 0x91, 0x11, 0xae, 0xb2}}	SA PEI GOP Init failure
{0x348cc7fe, 0x1e9a, 0x4c7a, {0x86, 0x28, 0xae, 0x48, 0x5b, 0x42, 0x10, 0xf0}}	SA PEI GOP GetMode failure
{0x5de1c071, 0x2c9c, 0x4a53, {0x80, 0x21, 0x4e, 0x80, 0xd2, 0x5d, 0x44, 0xa8}}	MRC training failure

When the FSP calls `ReportStatusCode()`, the `Type` parameter's `EFI_STATUS_CODE_TYPE_MASK` must be `EFI_ERROR_CODE` with the `EFI_STATUS_CODE_SEVERITY_MASK >= EFI_ERROR_UNRECOVERED`. The `Value` and `Instance` parameters must be 0.

The bootloader must register a listener for this status code. This listener should check if `DataHeader.Type == STATUS_CODE_DATA_TYPE_FSP_ERROR_GUID` to detect an `FSP_ERROR_INFO` notification. If an `FSP_ERROR_INFO` notification is encountered, the bootloader should assume that normal operation is no longer possible. In debug scenarios, this notification should be considered an `ASSERT`.

4.4 Dispatch Mode Integration

Dispatch Mode Integration Notes:

1. The FSP for this platform contains PEIMs compiled for the IA32 architecture. The boot loader therefore must utilize a PEI Foundation compiled for the IA32 architecture.
2. Since the FSP binary can be integrated into flash at any address, the boot loader has to report FSP FVs to the PEI and DXE dispatcher using PI specification defined mechanisms so PEIMs and DXE drivers inside the FSP Binary can be dispatched. `FspmWrapperPeim` and `FspWrapperPeim` from `IntelFsp2WrapperPkg` can aid in implementing this.
3. For this platform, FSP-T, FSP-M, and FSP-S contain 1 FV each.
4. The FSP distribution package will include a DSC file which contains all DynamicEx PCDs consumed by the FSP binary. The boot loader should include the DSC during its build process so that any PCDs defined by this DSC file are included in the boot loader's PCD database. This enables the boot loader and FSP to share a single PCD database.

- A NULL library (`FspPcdListLibNull.inf`) is included in the FSP distribution package. This library should be included in one of the boot loader's PEIMs. This ensures all DynamicEx PCDs used by the FSP are included in the boot loader's PCD database. One can fulfill this requirement by including the following code snippet in `*BoardPkg.dsc`:

```
IntelFsp2WrapperPkg/FspmWrapperPeim/FspmWrapperPeim.inf {
  <LibraryClasses>
  !if gIntelFsp2WrapperTokenSpaceGuid.PcdFspModeSelection == 0
    #
    # In FSP Dispatch mode below dummy library should be linked to bootloader PEIM
    # to build all DynamicEx PCDs that FSP consumes into bootloader PCD database.
    #
    NULL|$(PLATFORM_FSP_BIN_PACKAGE)/Library/FspPcdListLib/FspPcdListLibNull.inf
  !endif
}
```

5. The boot loader must provide at minimum 284KB of stack and 128KB of HOB heap to execute FSP on this platform.
6. In dispatch mode, the boot loader should not use FSP API calls described in chapter 5 of this document or chapter 8 of the FSP External Architecture Specification version 2.2. The `TempRamInit` API is the only exception, it is supported in both API mode and dispatch mode. All other APIs (`MemoryInit`, `SiliconInit`, etc.) should not be invoked.
7. For dispatch mode, FSP contains x64 DXE drivers to replace the `NotifyPhase` API. This eliminates thunking from 64bit to 32bit when using FSP dispatch mode. The boot loader should remove `S3EndOfPeiNotify` and `FspWrapperNotifyDxe` since they are no longer used in dispatch mode.
8. `EFI_PEI_CORE_FV_LOCATION_PPI` should be installed by the boot loader's SEC phase. `EFI_PEI_CORE_FV_LOCATION_PPI` should point to the first Firmware Volume (FV) in FSP-M so the `PeiCore` inside FSP will be invoked. If `EFI_PEI_CORE_FV_LOCATION_PPI` is not installed or `PeiCore` cannot be found at the address specified by `EFI_PEI_CORE_FV_LOCATION_PPI`, the `PeiCore` from the Boot Firmware Volume (BFV) will be invoked instead.

9. FSP-S requires multi-threaded code to complete silicon initialization on this platform. FSP-S includes the `UefiCpuPkg/CpuMpPei/CpuMpPei.inf` PEIM to provide multiprocessing. The bootloader can choose to either use the `MP_SERVICES` provided by the FSP for all PEIMs or the bootloader may provide an alternative implementation. In either case, only one `MP_SERVICES` implementation can be active at once. If the bootloader wishes to not use the FSP provided `MP_SERVICES`, then the bootloader must install the `MP_SERVICES_PPI` before installing the `FV_INFO_PPI` for FSP-S. If `MP_SERVICES_PPI` already exists, then FSP-S will use it and not execute its own implementation.
10. If you are using the Intel Reference BIOS, some EDK2 overrides may be required for Dispatch Mode, please refer to override folders in reference code or the override EDK2 repo for more details. For open source MinPlatform based firmware, no EDK2 overrides are required.
11. `FSPM_ARCH_CONFIG_PPI->NvsBufferPtr` is now a universal policy option (FSP Dispatch Mode and EDK2 native.) To enable the fast MRC training flow, the boot loader or platform code must to install this PPI to restore the previous MRC training data (`SA_MISC_PEI_PREMEM_CONFIG->S3DataPtr` is obsolete).
12. Policy initialization Flow Changes:
 - PEIMs from FSP-M/FSP-S should be dispatched early in PEI to produce the *DefaultPolicyInit* PPIs. -> Bootloader consumes the *DefaultPolicyInit* PPIs produced by the FSP binary to create the policy PPIs with default settings. -> Bootloader then locates and updates the policy PPIs as needed. -> Bootloader installs the *PolicyReadyPpi* after policy updates are completed. This signals to the FSP that silicon initialization may proceed.
 - Bootloader shall consume two PPIs produced by FSP binary to create policy PPIs with default settings. These PPIs are:
 - `PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI`
 - `PEI_SI_DEFAULT_POLICY_INIT_PPI`
 - The bootloader shall call the two functions below after the bootloader has completed any needed policy updates:
 - `SiPreMemInstallPolicyReadyPpi()`
 - `SiInstallPolicyReadyPpi()`
13. Debug message handling in dispatch mode:
 - Before the `ReportStatusCode` service is ready, a debug built FSP will send debug messages using the FSP-T UPD configuration (passed as FSP-T API input parameter). FSP-T is recommended to be used regardless FSP API mode or Dispatch mode.
 - Once the `ReportStatusCode` service is ready, a debug built FSP will send debug messages using the `ReportStatusCode` service.
 - It is recommended that bootloader register a `StatusCode` listener immediately after the `ReportStatus->Code` service is ready. It is important to register this listener as soon as possible so that all debug messages sent by the FSP are captured.
 - Please refer to section 9.4.7 in the Intel(R) Firmware Support Package External Architecture Specification v2.2 for details about the `ReportStatusCode` debug message format.
14. Enable or Disable FSP `S3BootScript` functionality: Since `edk2-stable201911` release the `PiDxeS3Boot->ScriptLib` supports enabling or disabling `S3BootScript` functionality by PCD, and FSP consumes this PCD as `DynamicEx` type to enable or disable internal `S3BootScript` functionality. Bootloader must configure below PCD before entering DXE phase or before executing FSP DXE drivers.
 - `gEfiMdeModulePkgTokenSpaceGuid.PcdAcpiS3Enable = TRUE` if `S3BootScript` should be enabled.
 - `gEfiMdeModulePkgTokenSpaceGuid.PcdAcpiS3Enable = FALSE` if `S3BootScript` should be disabled.

Chapter 5

FSP API Mode

Overview

This release of the FSP supports the all APIs required by the FSP External Architecture Specification version 2.2. These APIs are only used when running the FSP in API mode. In Dispatch mode, these APIs are not used (with the exception of TempRamInit.) The FSP information header contains the address offset for these APIs. Register usage is described in the FSP External Architecture Specification version 2.2. Any usage not described by the specification is described in the individual sections below.

The sections below will highlight any changes that are specific to this FSP release.

5.1 FSP APIs

5.1.1 TempRamInit API

Please refer Chapter 8.6 in the FSP External Architecture Specification version 2.2 for complete details including the prototype, parameters and return value details for this API.

TempRamInit does basic early initialization primarily setting up temporary RAM using cache. It returns ECX pointing to beginning of temporary memory and EDX pointing to end of temporary memory + 1. The total temporary ram currently available is given by [PcdTemporaryRamSize](#) starting from the base address of [PcdTemporaryRamBase](#). Out of the total temporary memory available, the last [PcdFspReservedBufferSize](#) bytes of space are reserved by the FSP for TempRamInit if temporary RAM initialization is done by the FSP. Any remaining space from **TemporaryRamBase**(ECX) to **TemporaryRamBase+TemporaryRamSize-FspReservedBufferSize** (EDX) is available for both bootloader and FSP use.

TempRamInit** also sets up the code caching of the region passed in through CodeCacheBase and CodeCacheLength, which are input parameters to TempRamInitApi. if 0 is passed in for CodeCacheBase, the base used will be (4 GB - 1 - CodeCacheLength).

Note

: When programming MTRRs CodeCacheLength will be reduced, if the LLC size on the current processor is smaller than the requested size.

It is a requirement for Firmware to have a Firmware Interface Table (FIT). The FIT contains pointers to each microcode update. The microcode update is loaded for all logical processors before executing the reset vector. If more than one microcode update for the CPU is present, the microcode update with the latest revision is loaded.

FSPT_UPD.MicrocodeRegionBase** and **FSPT_UPD.MicrocodeRegionLength** are input parameters to TempRamInit API. If these values are 0, FSP will not attempt to update microcode. If MicrocodeRegionBase != 0 and MicrocodeRegionLength != 0, then FSP will search that region for microcode updates. If a newer microcode update revision is found in the region, FSP will load it.

TempRamInit** will program MTRRs values to provide the following memory map:

Memory range	Cache Attribute
0xFE000000 - 0x00040000	Write back
CodeCacheBase - CodeCacheLength	Write protect

5.1.2 FspMemoryInit API

Please refer to Chapter 8.7 in the FSP External Architecture Specification version 2.2 for the prototype, parameters and return value details for this API.

The variable **FspmUpdPtr** is a pointer to the **FSPM_UPD** structure which is described in the header file **FspmUpd.h**.

The bootloader must pass valid a CAR region for FSP through **FSPM_UPD.FspmArchUpd.StackBase** and **FSPM_UPD.FspmArchUpd.StackSize** UPDs.

The FSP for this platform will run **FspMemoryInit** top of the stack provided by the bootloader instead of establishing a separate stack as described by the FSP External Architecture Specification version 2.1/2.2. The memory region provided by the **FSPM_UPD.FspmArchUpd.StackBase** and **FSPM_UPD.FspmArchUpd.StackSize** UPDs is used to establish a HOB heap. The names **StackBase** and **StackSize** can be confusing since they are **NOT** used for stack. These names were retained for backwards compatibility with FSP v2.0.

Below are the heap and stack requirements for FSP on this platform:

HOB Heap requirement:

HOB Heap	UPD	Setting
Base	FSPM_UPD.FspmArchUpd.StackBase	Any non-conflict CAR region (0xFE17F00 as default)
Size	FSPM_UPD.FspmArchUpd.StackSize	at least 128KB

Stack requirement: FSP's stack usage starts from the current stack pointer. The minimum stack size requirement for FSP-M is 284KB.

The bootloader must ensure that sufficient stack space is available to fulfill the FSP-M minimum stack size requirement at the point in execution where **FspMemoryInit()** is called. The stack allocated by the bootloader must be large enough for both FSP-M as well as any other parent function calls that are still on the stack at the point when **FspMemoryInit()** is called.

After **FspMemoryInit()** is completed, permanent memory is available. After this point, the memory pressure experienced early in boot is eliminated. Accordingly, right before **FspMemoryInit()** exits, any data that needs to be retained for later use by **FspSiliconInit()** will be copied to permanent memory. **FspSiliconInit()** will then execute on a second stack.

The base address of HECI device (Bus 0, Device 22, Function 0) is required to be initialized prior to calling **FspMemoryInit()**. The default address is programmed to 0xFED1A000.

FspMemoryInit() will calculate the memory map by taking into account the size of several memory regions: TSEG, IED, GTT, BDSM, ME stolen, Uncore PMRR, IOT, MOT, DPR, REMAP, TOLUD, TOUTD. These memory regions may not be initialized by **FspMemoryInit()**, but space will be reserved for them.

5.1.3 TempRamExit API

Please refer to Chapter 8.8 in the FSP external Architecture Specification version 2.2 for the prototype, parameters and return value details for this API.

If Boot Loader initializes the Temporary RAM (CAR) and skip calling **TempRamInit API**, it is expected that boot-loader must skip calling this API and bootloader will tear down the temporary memory area setup in the cache and bring the cache to normal mode of operation.

The FSP for this platform doesn't have any input parameters for this API. The value of *TempRamExitParamPtr* should be NULL.

At the end of *TempRamExit* the original code and data cache are disabled. FSP will reconfigure all MTRRs as described in the table below. These MTRR values optimize performance in most scenarios. If the boot loader wishes to configure the MTRRs differently, they can be reprogrammed immediately after this API call.

Memory range	Cache Attribute
0xFF000000 - 0xFFFFFFFF (Flash region)	Write protect
0x00000000 - 0x0009FFFF	Write back
0x000C0000 - Top of Low Memory	Write back
xxxx - xxxx	x *Note1
0x100000000 - Top of High Memory	Write back *Note2

Stack requirement: 4KB of free stack space should be provided to execute **TempRamExit**.

Note1: Certain silicon features require specific cache types for specific memory ranges. These ranges will be configured by FSP when such features are enabled.

Note2: In some cases MTRRs might not be enough to cover all desired regions, in this case memory regions need to be adjusted for better alignment (e.g., adjust MmioSize or MmioSizeAdjustment UPD) Covering flash region and above 4GB memory is another case which may consume more MTRRs, when there is not enough MTRR available FSP will only cover above 4GB memory partially. In this case the boot loader can optimize MTRRs to remove flash from the cached regions after all needed data is loaded from flash and before booting the OS.

5.1.4 FspSiliconInit API

Please refer to Chapter 8.9 in the FSP external Architecture Specification version 2.2 for the prototype, parameters and return value details for this API.

The variable *FspUpdPtr* is a pointer to the **FSPS_UPD** structure which is described in the header file *FspUpd.h*.

It is expected that the boot loader will adjust MTRRs for SBSP if needed after **TempRamExit** but before entering **FspSiliconInit**. If the MTRRs are not programmed properly, boot performance can be impacted.

The region of 0x5_8000 - 0x5_8FFF is used by *FspSiliconInit* for starting APs. If this data is important to bootloader, then bootloader needs to preserve it before calling *FspSiliconInit*.

FspSiliconInit requires multi-threaded code to complete silicon initialization on this platform. FSP includes the *UefiCpuPkg/CpuMpPei/CpuMpPei.inf* PEIM to provide multiprocessing. Accordingly, the boot loader must expect FSP to perform an INIT-SIPI-SIPI during **FspSiliconInit** and **NotifyPhase** which will take control of all APs in the system and restart them. This will kill any background threads that were running on the APs. In some cases, this may not be desirable. As an alternative, the boot loader may provide an instance of the [EFI_PEI_MP_SERVICES_PPI](#) through the *FspUpd->FspConfig.CpuMpPpi* UPD for the FSP to use instead of the built-in implementation. **This is entirely optional**, if callbacks from FSP to the boot loader are not desired, one may set *FspUpd->FspConfig.CpuMpPpi == NULL*.

In summary, there are two methods that FSP can use for performing multiprocessing:

1. Using the multiprocessing services built-in to the FSP.

2. Using the boot loader's multiprocessing services.

Using method #1, the boot loader will set `FspUpd->FspConfig.CpuMpPpi == NULL`. If this UPD is NULL, then FSP will use its built-in MP services implementation for multiprocessing. Accordingly, the boot loader must expect FSP to perform an INIT-SIPI-SIPI during **FspSiliconInit** and **NotifyPhase** which will take control of all APs in the system and restart them. This will kill any background threads that were running on the APs.

Note: If bootloader needs to take control of APs back, a full AP re-initialization is required after FSP-S completed and control has been transferred back to bootloader.

Using method #2, the boot loader will set `FspUpd->FspConfig.CpuMpPpi != NULL`. If this UPD is not NULL, it must point to an instance of [EFI_PEI_MP_SERVICES_PPI](#). When the FSP needs to perform multiprocessing, it will use the [EFI_PEI_MP_SERVICES_PPI](#) instance provided by the boot loader to do so. Accordingly, the boot loader must expect the function pointers in [EFI_PEI_MP_SERVICES_PPI](#) to be invoked in the middle of the execution of **FspSiliconInit** and **NotifyPhase**.

It is a requirement for the bootloader to have a Firmware Interface Table (FIT), which contains pointers to each microcode. The microcode is loaded for all cores before reset vector. If more than one microcode update for the CPU is present, the latest revision is loaded.

MicrocodeRegionBase and MicrocodeRegionLength are both input parameters to TempRamInit and UPD for SiliconInit API. UPD has priority and will be searched for a later revision than TempRamInit. If MicrocodeRegionBase and MicrocodeRegionLength values are 0, FSP will not attempt to update the microcode. If a microcode region is passed, FSP will search that region for microcode updates. If a newer microcode update revision is found in the region, FSP will load it.

FspSiliconInit initializes PCH audio. This includes selecting the HD Audio verb table and initializing the audio CODEC.

FspSiliconInit initializes the following PCH features: HECI, USB, HSIO, Integrated Sensor Hub, Camera, PCI Express, DMI, Vt-d.

FspSiliconInit initializes the following CPU features: XD, VMX, AES, IED, HDC, x(2)APIC, Intel® Processor Trace, Three Strike Counter, Machine Check, Cache Pre-fetchers, Core PMRR, and Power Management.

FspSiliconInit initializes Internal Graphics. During this initialization, FSP publishes the [EFI_PEI_GRAPHICS_INFO_HOB](#) and the [EFI_PEI_GRAPHICS_DEVICE_INFO_HOB](#). These HOBs will not be published during S3 resume as FSP will not initialize the internal graphics frame buffer during S3 resume.

FspSiliconInit programs SA BARs: MchBar, DmiBar, EpBar, GdxcBar, EDRAM (if supported). Please refer to section 2.8 (MemoryMap) for the corresponding Bar values. GttMadr (0xDF000000) and GmAdr(0xC0000000) are temporarily programmed and cleared after use in FSP.

Stack requirement: 4KB of free stack space should be provided to execute *FspSiliconInit*.

5.1.5 FspMultiPhaseSilnit API

Please refer Chapter 8.10 in the FSP External Architecture Specification version 2.2 for the prototype, parameters and return value details for this API.

If the FspMultiPhaseSilnit API is enabled (by setting `FSPS_ARCH_UPD->EnableMultiPhaseSiliconInit` UPD to 1 before the FspSiliconInit API is invoked), the bootloader should call this API with the MultiPhaseAction parameter set to 0 to get the NumberOfPhases supported by this platform. FSP will return control back to the bootloader NumberOfPhases times and then the bootloader should call FspMultiPhaseSilnit API with the MultiPhaseAction parameter set to 1 by NumberOfPhases times.

Note

: Disable `FSPS_ARCH_UPD->EnableMultiPhaseSiliconInit` when we disable `FSP_M_CONFIG->TcssXhciEn`.

This platform supports the following MultiPhaseSilnit phase(s):

Phase	FSP return point	Purpose
1	After TCSS initialization completed	for TCSS board specific initialization from bootloader side

5.1.6 NotifyPhase API

Please refer Chapter 8.11 in the FSP External Architecture Specification version 2.2 for the prototype, parameters and return value details for this API.

Stack requirement: 4KB of free stack space should be provided to execute *NotifyPhase*.

5.1.6.1 PostPciEnumeration Notification

This phase *EnumInitPhaseAfterPciEnumeration* is to be called after PCI enumeration but before execution of third party code such as option ROMs. It includes powering down unused PCH SATA ports, locking registers in PMC, DMI, TCO, and SPI Flash (DLOCK + WRSDIS + FLOCKDN). It also includes enabling bus mastering for eSPI connected devices.

5.1.6.2 ReadyToBoot Notification

This phase *EnumInitPhaseReadyToBoot* is to be called before giving control to the operating system. It includes some final initialization steps recommended by the BWG, including power management settings, and sending ME Message EOP (End of Post).

5.1.6.3 EndOfFirmware Notification

This phase *EnumInitEndOfFirmware* is to be called before the firmware/preboot environment transfers management of all system resources to the OS or next level execution environment. It includes final locking of chipset registers.

5.1.7 FSP Events API

Please refer Chapter 8.5 in the FSP External Architecture Specification version 2.2 for the prototype, parameters and return value details for these APIs.

This platform supports FSP Events which re-direct debug messages to event handlers provided by bootloader. To enable this function the bootloader must provide event handler function pointers via UPDs.

FSP phase	UPD for bootloader to pass handler function pointer
FSP-T	FSPT_ARCH_UPD -> FspDebugHandler
FSP-M	FSPM_ARCH_UPD -> FspEventHandler
FSP-S	FSPS_ARCH_UPD -> FspEventHandler

5.2 Reset Return Codes

As per FSP External Architecture Specification version 2.0/2.1/2.2, any reset required in the FSP flow will be reported by returning one of the FSP_STATUS_RESET_REQUIRED* return codes.

It is the boot loader's responsibility to reset the system according to the reset type requested.

Below table specifies the return status returned by FSP API and the requested reset type.

FSP_STATUS_RESET_REQUIRED Code	Reset Type requested
0x40000001	Cold Reset
0x40000002	Warm Reset
0x40000003	Global Reset - Puts the system through a Global Reset through HECI or a Full Reset through PCH
0x40000004	Reserved
0x40000005	Reserved
0x40000006	Reserved
0x40000007	Reserved
0x40000008	Reserved

5.3 UPD Porting Guide

Recommended values for UPDs:

UPD	Dependency	Description	Value
CstateLatencyControl1Irtl	Server platform	Server platform should has different setting	0x6B
PchPcieHsioRxSetCtleEnable	Board design	Different board requires different value	tune
PchPcieHsioRxSetCtle	Board design	Different board requires different value	tune
PchSataHsioRxGen3EqBoostMag↔ Enable	Board design	Different board requires different value	tune
PchSataHsioRxGen3EqBoostMag	Board design	Different board requires different value	tune
PchSataHsioTxGen1DownscaleAmp↔ Enable	Board design	Different board requires different value	tune
PchSataHsioTxGen1DownscaleAmp	Board design	Different board requires different value	tune
PchSataHsioTxGen2DownscaleAmp↔ Enable	Board design	Different board requires different value	tune
PchSataHsioTxGen2DownscaleAmp	Board design	Different board requires different value	tune
PchNumRsvdSmbusAddresses	Board design	Different board requires different value	tune
RsvdSmbusAddressTablePtr	Board design	Different board requires different value	tune
BiosSize	Board design	Different board requires different value	tune

Chapter 6

Porting Recommendations

Here are some notes and recommendations for adapting an existing boot loader to FSP.

6.1 Locking PAM register

FSP 2.0 introduced the EndOfFirmware Notify phase callback which is the recommended place for locking PAM registers. Accordingly, by default FSP locks the PAM registers during the EndOfFirmware Notify phase. If the EndOfFirmware Notify phase is too early to lock PAM registers, then the PAM locking code inside the FSP can be disabled by setting the UPD -> FSP_S_TEST_CONFIG -> SkipPamLock or SA policy -> [_SI_PREMEM_POLICY_STRUCT](#) -> [SA_MISC_PEL_CONFIG](#) -> SkipPamLock. If the PAM locking code inside the FSP is skipped, then the boot loader must lock the PAM registers before booting the OS. by programming one PCI config space register as below.

The PAM registers must be locked in all boot paths including S3 resume. To lock the PAM registers, program the following lock bit:

```
MmioOr32 (B0: D0: F0: Register 0x80, BIT0)
```

6.2 Locking SMRAM register

It is recommended that SMRAM be locked before any third party code (e.g. OpROM) execution. The point in execution where third party OpROMs begin executing can vary depending on the boot loader implementation. To provide flexibility, the FSP by default will not lock it. The boot loader should lock SMRAM by programming the following lock bit before any third party OpROM execution. Additionally, SMRAM should be locked before the **EnumInitPhaseReadyToBoot** notify phase is called. During S3 resume, the lock bit should be set right before the OS wake vector.

```
PciOr8 (B0: D0: F0: Register 0x88, BIT4);
```

Note: This register must be programmed using the legacy CF8/CFC PCI access mechanism. (MMIO access will not work)

6.3 Locking SMI register

It is recommended that the global SMI bit is locked before any third party code (e.g. OpROM) execution. SMM initialization flows may vary depending on boot loader implementation details. Accordingly, FSP will not lock it by default. The boot loader is responsible for locking the following registers after SMM configuration is complete. Set AcpiBase + 0x30[0] to 1b to enable global SMI. Set PMC PCI offset A0h[4] = 1b to lock SMI.

6.4 Verify below settings are correct for your platforms

PMC PCI configuration space is not PCI specification compliant. The FSP will hide the PMC controller to avoid external software or OS from corrupting the BAR addresses. FSP will program the PMC controller IO and MIO BAR's with below addresses. Please use these addresses in the boot loader code instead of reading BAR addresses from the PMC controller.

Register	Values
ABASE	0x1800
PWRMBASE	0xFE000000
PCIEXBAR_BASE_ADDRESS	0xE0000000

Note

:

- Boot Loader can use a different value for PCIEXBAR_BASE_ADDRESS either by modifying the UPD (under FSP-T) or by overriding the PCIEXBAR (B0:D0:F0:R60h) before calling FspMemoryInit.
- Boot Loader should avoid using conflicting address when reprogramming PCIEXBAR_BASE_ADDRESS than the recommended one.

Chapter 7

FSP Output

The FSP builds a series of data structures called Hand-Off-Blocks (HOBs) as it progresses through initializing the silicon.

Please refer to the Platform Initialization (PI) Specification - Volume 3: Shared Architectural Elements specification for PI Architectural HOBs. Please refer Chapter 10 in the FSP External Architecture Specification version 2.2 for details about FSP Architectural HOBs.

The sections below describe the HOBs not covered in the above two specifications.

7.1 SMRAM Resource Descriptor HOB

The FSP will report the system SMRAM T-SEG range through a generic resource HOB if T-SEG is enabled. The owner field of the HOB identifies the owner as T-SEG.

```
#define FSP_HOB_RESOURCE_OWNER_TSEG_GUID \
{ 0xd038747c, 0xd00c, 0x4980, { 0xb3, 0x19, 0x49, 0x01, 0x99, 0xa4, 0x7d, 0x55 } }
```

7.2 SMBIOS INFO HOB

The FSP will report the SMBIOS through a HOB with below [GUID](#). This information can be consumed by the bootloader to produce the SMBIOS tables. These structures are included as part of MemInfoHob.h, SmbiosCacheInfoHob.h, SmbiosProcessorInfoHob.h, & [FirmwareVersionInfoHob.h](#)

```
#define SI_MEMORY_INFO_DATA_HOB_GUID \
{ 0x9b2071d4, 0xb054, 0x4e0c, { 0x8d, 0x09, 0x11, 0xcf, 0x8b, 0x9f, 0x03, 0x23 } };

typedef struct {
    MrcDimmStatus Status;                ///< See MrcDimmStatus for the definition of this field.
    UINT8 DimmId;
    UINT32 DimmCapacity;                ///< DIMM size in MBytes.
    UINT16 MfgId;
    UINT8 ModulePartNum[20];            ///< Module part number for DDR3 is 18 bytes however for DDR4 20
    bytes as per JEDEC Spec, so reserving 20 bytes
    UINT8 RankInDimm;                  ///< The number of ranks in this DIMM.
    UINT8 SpdDramDeviceType;            ///< Save SPD DramDeviceType information needed for SMBIOS
    structure creation.
    UINT8 SpdModuleType;                ///< Save SPD ModuleType information needed for SMBIOS structure
    creation.
    UINT8 SpdModuleMemoryBusWidth;      ///< Save SPD ModuleMemoryBusWidth information needed for SMBIOS
    structure creation.
    UINT8 SpdSave[MAX_SPD_SAVE_DATA];  ///< Save SPD Manufacturing information needed for SMBIOS
    structure creation.
} DIMM_INFO;

typedef struct {
    UINT8 Status;                      ///< Indicates whether this channel should be used.
    UINT8 ChannelId;
    UINT8 DimmCount;                  ///< Number of valid DIMMs that exist in the channel.
```

```

MRC_CH_TIMING Timing[MAX_PROFILE];          ///< The channel timing values.
DIMM_INFO Dimm[MAX_DIMM];                   ///< Save the DIMM output characteristics.
} CHANNEL_INFO;
typedef struct {
    UINT8      Status;                      ///< Indicates whether this controller should be used.
    UINT16     DeviceId;                    ///< The PCI device id of this memory controller.
    UINT8      RevisionId;                  ///< The PCI revision id of this memory controller.
    UINT8      ChannelCount;                ///< Number of valid channels that exist on the controller.
    CHANNEL_INFO Channel[MAX_CH];           ///< The following are channel level definitions.
} CONTROLLER_INFO;
typedef struct {
    EFI_HOB_GUID_TYPE EfiHobGuidType;
    UINT8      Revision;
    UINT16     DataWidth;
    ///< As defined in SMBIOS 3.0 spec
    ///< Section 7.18.2 and Table 75
    UINT8      DdrType;                     ///< DDR type: DDR3, DDR4, or LPDDR3
    UINT32     Frequency;                   ///< The system's common memory controller frequency in MT/s.
    ///< As defined in SMBIOS 3.0 spec
    ///< Section 7.17.3 and Table 72
    UINT8      ErrorCorrectionType;
    SiMrcVersion Version;
    UINT32     FreqMax;
    BOOLEAN    EccSupport;
    UINT8      MemoryProfile;
    UINT32     TotalPhysicalMemorySize;
    UINT8      XmpProfileEnable;
    UINT8      Ratio;
    UINT8      RefClk;
    UINT32     VddVoltage[MAX_PROFILE];
    CONTROLLER_INFO Controller[MAX_NODE];
} MEMORY_INFO_DATA_HOB;
#define SI_MEMORY_PLATFORM_DATA_HOB \
    { 0x6210d62f, 0x418d, 0x4999, { 0xa2, 0x45, 0x22, 0x10, 0x0a, 0x5d, 0xea, 0x44 } }
typedef struct {
    UINT8      Revision;
    UINT8      Reserved[3];
    UINT32     BootMode;
    UINT32     TsegSize;
    UINT32     TsegBase;
    UINT32     PrmrrSize;
    UINT32     PrmrrBase;
    UINT32     GttBase;
    UINT32     MmioSize;
    UINT32     PciEBaseAddress;
} MEMORY_PLATFORM_DATA;
typedef struct {
    EFI_HOB_GUID_TYPE EfiHobGuidType;
    MEMORY_PLATFORM_DATA Data;
    UINT8      *Buffer;
} MEMORY_PLATFORM_DATA_HOB;
#define SMBIOS_CACHE_INFO_HOB_GUID \
    { 0xd805b74e, 0x1460, 0x4755, {0xbb, 0x36, 0x1e, 0x8c, 0x8a, 0xd6, 0x78, 0xd7} }

///
/// SMBIOS Cache Info HOB Structure
///
typedef struct {
    UINT16     ProcessorSocketNumber;
    UINT16     NumberOfCacheLevels;          ///< Based on Number of Cache Types L1/L2/L3
    UINT8      SocketDesignationStrIndex;    ///< String Index in the string Buffer. Example "L1-CACHE"
    UINT16     CacheConfiguration;           ///< Format defined in SMBIOS Spec v3.0 Section 7.8 Table 36
    UINT16     MaxCacheSize;                 ///< Format defined in SMBIOS Spec v3.0 Section 7.8.1
    UINT16     InstalledSize;                 ///< Format defined in SMBIOS Spec v3.0 Section 7.8.1
    UINT16     SupportedSramType;             ///< Format defined in SMBIOS Spec v3.0 Section 7.8.2
    UINT16     CurrentSramType;               ///< Format defined in SMBIOS Spec v3.0 Section 7.8.2
    UINT8      CacheSpeed;                   ///< Cache Speed in nanoseconds. 0 if speed is unknown.
    UINT8      ErrorCorrectionType;           ///< ENUM Format defined in SMBIOS Spec v3.0 Section 7.8.3
    UINT8      SystemCacheType;               ///< ENUM Format defined in SMBIOS Spec v3.0 Section 7.8.4
    UINT8      Associativity;                 ///< ENUM Format defined in SMBIOS Spec v3.0 Section 7.8.5
    ///

```

```

UINT64      ProcessorId;                ///< ENUM defined in SMBIOS Spec v3.0 Section 7.5.3
UINT8       ProcessorVersionStrIndex;   ///< Index of the String in the String Buffer
UINT8       Voltage;                   ///< Format defined in SMBIOS Spec v3.0 Section 7.5.4
UINT16      ExternalClockInMHz;         ///< External Clock Frequency. Set to 0 if unknown.
UINT16      CurrentSpeedInMHz;          ///< Snapshot of current processor speed during boot
UINT8       Status;                     ///< Format defined in the SMBIOS Spec v3.0 Table 21
UINT8       ProcessorUpgrade;           ///< ENUM defined in SMBIOS Spec v3.0 Section 7.5.5
///< This info is used for both CoreCount & CoreCount2 fields
///< See detailed description in SMBIOS Spec v3.0 Section 7.5.6
UINT16      CoreCount;
///< This info is used for both CoreEnabled & CoreEnabled2 fields
///< See detailed description in SMBIOS Spec v3.0 Section 7.5.7
UINT16      EnabledCoreCount;
///< This info is used for both ThreadCount & ThreadCount2 fields
///< See detailed description in SMBIOS Spec v3.0 Section 7.5.8
UINT16      ThreadCount;
UINT16      ProcessorCharacteristics;    ///< Format defined in SMBIOS Spec v3.0 Section 7.5.9
///< String Buffer - each string terminated by NULL "0x00"
///< String buffer terminated by double NULL "0x0000"
} SMBIOS_PROCESSOR_INFO;
#define SMBIOS_FIRMWARE_VERSION_INFO_HOB_GUID \
    { 0x798e722e, 0x15b2, 0x4e13, { 0x8a, 0xe9, 0x6b, 0xa3, 0x0f, 0xf7, 0xf1, 0x67 }}

///<
///< Firmware Version Structure
///<
typedef struct {
    UINT8                      MajorVersion;
    UINT8                      MinorVersion;
    UINT8                      Revision;
    UINT16                     BuildNumber;
} FIRMWARE_VERSION;

///<
///< Firmware Version Information Structure
///<
typedef struct {
    UINT8                      ComponentNameIndex;    ///< Offset 0   Index of Component Name
    UINT8                      VersionStringIndex;    ///< Offset 1   Index of Version String
    FIRMWARE_VERSION           Version;               ///< Offset 2-6 Firmware version
} FIRMWARE_VERSION_INFO;

///<
///< The Smbios structure header.
///<
typedef struct {
    UINT8                      Type;
    UINT8                      Length;
    UINT16                     Handle;
} SMBIOS_STRUCTURE;

///<
///< Firmware Version Information HOB Structure
///<
typedef struct {
    SMBIOS_STRUCTURE           Header;                ///< SMBIOS structure header
    UINT8                      Count;                 ///< Number of FVI entries in this structure
    INTEL_FIRMWARE_VERSION_INFO Fvi[1];              ///< FVI structure(s)
} FIRMWARE_VERSION_INFO_HOB;

```

7.3 CHIPSETINIT INFO HOB

The FSP will report the ChipsetInit CRC through a HOB with below GUID. This information can be consumed by the bootloader to check if ChipsetInit CRC is matched between BIOS and ME. These structures are included as part of FspUpd.h

```

#define CHIPSETINIT_INFO_HOB_GUID \
    { 0xc1392859, 0x1f65, 0x446e, { 0xb3, 0xf5, 0x84, 0x35, 0xfc, 0xc7, 0xd1, 0xc4 }}

///<
///< The ChipsetInit Info structure provides the information of ME ChipsetInit CRC and BIOS ChipsetInit CRC.
///<
typedef struct {
    UINT8                      Revision;
    UINT8                      Rsvd[3];
    UINT16                     MeChipInitCrc;
    UINT16                     BiosChipInitCrc;
} CHIPSET_INIT_INFO;

```

7.4 HOB USAGE INFO HOB

The FSP will report the Hob memory usage through a HOB with below GUID. This information can be consumed by the bootloader to check how much temporary RAM is left.

```
#define HOB_USAGE_DATA_HOB_GUID \
{0xc764a821, 0xec41, 0x450d, { 0x9c, 0x99, 0x27, 0x20, 0xfc, 0x7c, 0xe1, 0xf6 }}

typedef struct {
    EFI_PHYSICAL_ADDRESS EfiMemoryTop;
    EFI_PHYSICAL_ADDRESS EfiMemoryBottom;
    EFI_PHYSICAL_ADDRESS EfiFreeMemoryTop;
    EFI_PHYSICAL_ADDRESS EfiFreeMemoryBottom;
    UINTN FreeMemory;
} HOB_USAGE_DATA_HOB;
```

7.5 FSP_ERROR_INFO_HOB

In the case of an error occurring during the execution of the FSP, the FSP may produce this HOB which describes the error in more detail. [FSP_ERROR_INFO_HOB](#) is only used in FSP API Mode. In FSP Dispatch Mode, FSP may call ReportStatusCode() and provide a FSP_ERROR_INFO structure using the PI status code services.

```
#define FSP_ERROR_INFO_HOB_GUID \
{0x611e6a88, 0xad7, 0x4301, { 0x93, 0xff, 0xe4, 0x73, 0x04, 0xb4, 0x3d, 0xa6 }}

typedef struct {
    EFI_HOB_GUID_TYPE GuidHob;
    EFI_STATUS_CODE_TYPE Type;
    EFI_STATUS_CODE_VALUE Value;
    UINT32 Instance;
    EFI_GUID CallerId;
    EFI_GUID ErrorType;
    UINT32 Status;
} FSP_ERROR_INFO_HOB;
```

The FSP for this platform implements the following CallerId GUIDs:

CallerId	Description
{0x1f4dc7e9, 0x26ca, 0x4336, {0x8c, 0xe3, 0x39, 0x31, 0x03, 0xb5, 0xf3, 0xd7}}	ME
{0x98230916, 0xe632, 0x49ff, {0x81, 0x81, 0x55, 0xce, 0xe5, 0x10, 0x36, 0x89}}	System Agent
{0x5a47c211, 0x642f, 0x4f92, {0x9c, 0xb3, 0x7f, 0xeb, 0x93, 0xda, 0xdd, 0xba}}	MRC

The following ErrorType GUIDs are implemented:

ErrorType	Description
{0x948585c4, 0x76a4, 0x45bb, {0xbe, 0x6c, 0x39, 0x61, 0xc3, 0xab, 0xde, 0x15}}	ME EOP failure
{0x8106a5cc, 0x30ba, 0x41cf, {0xa1, 0x78, 0x63, 0x38, 0x91, 0x11, 0xae, 0xb2}}	SA PEI GOP Init failure
{0x348cc7fe, 0x1e9a, 0x4c7a, {0x86, 0x28, 0xae, 0x48, 0x5b, 0x42, 0x10, 0xf0}}	SA PEI GOP GetMode failure
{0x5de1c071, 0x2c9c, 0x4a53, {0x80, 0x21, 0x4e, 0x80, 0xd2, 0x5d, 0x44, 0xa8}}	MRC training failure

Chapter 8

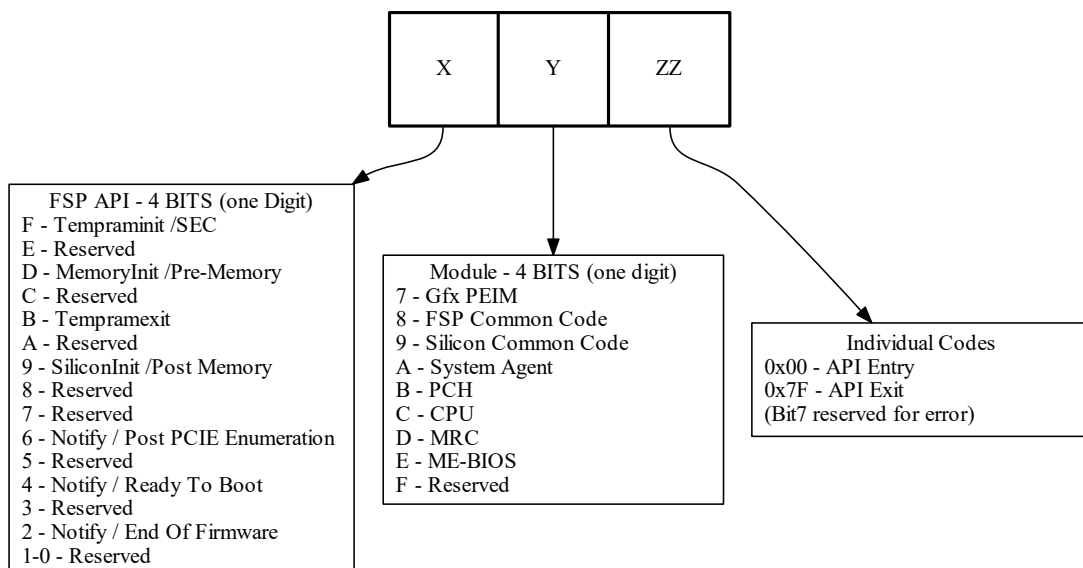
POST Codes

The FSP outputs 16-bit POST codes to indicate which API and which module is currently executing.

Bit Range	Description
Bit15 - Bit12 (X)	used to indicate the phase/api under which the code is executing
Bit11 - Bit8 (Y)	used to indicate the module
Bit7 (ZZ bit 7)	reserved for error
Bit6 - Bit0 (ZZ)	individual codes

8.1 POST Code Info

The diagram below illustrates how sub-fields are encoded in the POST codes produced by the FSP.



8.1.1 TempRamInit API Status Codes (0xFxxx)

PostCode	Module	Description
0x0000	FSP	TempRamInit API Entry (The change in upper byte is due to not enabling of the Port81 early in the boot)
0x007F	FSP	TempRamInit API Exit

8.1.2 FspMemoryInit API Status Codes (0xDxxx)

PostCode	Module	Description
0xD800	FSP	FspMemoryInit API Entry
0xD87F	FSP	FSpMemoryInit API Exit
0xDA00	SA	Pre-Mem Salnit Entry
0xDA02	SA	OverrideDev0Did Start
0xDA04	SA	OverrideDev2Did Start
0xDA06	SA	Programming SA Bars
0xDA08	SA	Install SA HOBs
0xDA0A	SA	Reporting SA PCIe code version
0xDA0C	SA	SaSvInit Start
0xDA10	SA	Initializing DMI
0xDA15	SA	Initialize TCSS PreMem
0xDA1F	SA	Initializing DMI/OPI Max PayLoad Size
0xDA20	SA	Initializing SwitchableGraphics
0xDA30	SA	Initializing SA PCIe
0xDA3F	SA	Programming PEG credit values Start
0xDA40	SA	Initializing DMI Tc/Vc mapping
0xDA42	SA	CheckOffboardPcieVga
0xDA44	SA	CheckAndInitializePegVga
0xDA50	SA	Initializing Graphics
0xDA52	SA	Initializing System Agent Overclocking
0xDA7F	SA	Pre-Mem Salnit Exit
0xDB00	PCH	Pre-Mem PchInit Entry
0xDB02	PCH	Pre-Mem Disable PCH fused controllers
0xDB15	PCH	Pre-Mem SMBUS configuration
0xDB48	PCH	Pre-Mem PchOnPolicyInstalled Entry
0xDB49	PCH	Pre-Mem Program HSIO
0xDB4A	PCH	Pre-Mem DCI configuration
0xDB4C	PCH	Pre-Mem Host DCI enabled
0xDB4D	PCH	Pre-Mem Trace Hub - Early configuration
0xDB4E	PCH	Pre-Mem Trace Hub - Device disabled
0xDB4F	PCH	Pre-Mem TraceHub - Programming MSR
0xDB50	PCH	Pre-Mem Trace Hub - Power gating configuration
0xDB51	PCH	Pre-Mem Trace Hub - Power gating Trace Hub device and locking HSWPGCR1 register
0xDB52	PCH	Pre-Mem Initialize HPET timer
0xDB55	PCH	Pre-Mem PchOnPolicyInstalled Exit
0xDB7F	PCH	Pre-Mem PchInit Exit
0xDC00	CPU	CPU Pre-Mem Entry
0xDC0F	CPU	CpuAddPreMemConfigBlocks Done
0xDC20	CPU	CpuOnPolicyInstalled Start
0xDC2F	CPU	XmmInit Start
0xDC3F	CPU	TxtInit Start

PostCode	Module	Description
0xDC4F	CPU	Init CPU Straps
0xDC5F	CPU	Init Overclocking
0xDC6F	CPU	CPU Pre-Mem Exit
0x**55	SA	MRC_MEM_INIT_DONE
0x**D5	SA	MRC_MEM_INIT_DONE_WITH_ERRORS
0xDD00	SA	MRC_INITIALIZATION_START
0xDD10	SA	MRC_CMD_PLOT_2D
0xDD1B	SA	MRC_FAST_BOOT_PERMITTED
0xDD1C	SA	MRC_RESTORE_NON_TRAINING
0xDD1D	SA	MRC_PRINT_INPUT_PARAMS
0xDD1E	SA	MRC_SET_OVERRIDES_PSPD
0xDD20	SA	MRC_SPD_PROCESSING
0xDD21	SA	MRC_SET_OVERRIDES
0xDD22	SA	MRC_MC_CAPABILITY
0xDD23	SA	MRC_MC_CONFIG
0xDD24	SA	MRC_MC_MEMORY_MAP
0xDD25	SA	MRC_PRINT_DDR_MRS
0xDD26	SA	MRC_RESET_SEQUENCE
0xDD27	SA	MRC_PRE_TRAINING
0xDD28	SA	MRC_EARLY_COMMAND
0xDD29	SA	MRC_SENSE_AMP_OFFSET
0xDD2A	SA	MRC_READ_MPR
0xDD2B	SA	MRC_RECEIVE_ENABLE
0xDD2C	SA	MRC_JEDEC_WRITE_LEVELING
0xDD2D	SA	MRC_ROUNDTRIP_MATCH
0xDD2E	SA	MRC_WRITE_TIMING_1D
0xDD2F	SA	MRC_READ_TIMING_1D
0xDD30	SA	MRC_DIMM_ODT
0xDD31	SA	MRC_EARLY_WRITE_TIMING_2D
0xDD32	SA	MRC_WRITE_DS
0xDD33	SA	MRC_WRITE_EQ
0xDD34	SA	MRC_EARLY_READ_TIMING_2D
0xDD35	SA	MRC_READ_ODT
0xDD36	SA	MRC_READ_EQ
0xDD37	SA	MRC_READ_AMP_POWER
0xDD38	SA	MRC_WRITE_TIMING_2D
0xDD39	SA	MRC_READ_TIMING_2D
0xDD3A	SA	MRC_CMD_VREF
0xDD3B	SA	MRC_WRITE_VREF_2D
0xDD3C	SA	MRC_READ_VREF_2D
0xDD3D	SA	MRC_POST_TRAINING
0xDD3E	SA	MRC_LATE_COMMAND
0xDD3F	SA	MRC_ROUND_TRIP_LAT
0xDD40	SA	MRC_TURN_AROUND
0xDD41	SA	MRC_CMP_OPT
0xDD42	SA	MRC_SAVE_MC_VALUES
0xDD43	SA	MRC_RESTORE_TRAINING
0xDD44	SA	MRC_RMT_TOOL
0xDD45	SA	MRC_WRITE_SR
0xDD46	SA	MRC_DIMM_RON
0xDD47	SA	MRC_RCVEN_TIMING_1D

PostCode	Module	Description
0xDD48	SA	MRC_MR_FILL
0xDD49	SA	MRC_PWR_MTR
0xDD4A	SA	MRC_MR_WRITE0
0xDD4B	SA	MRC_WRITE_VOLTAGE_1D
0xDD4C	SA	MRC_EARLY_RDMPR_TIMING_2D
0xDD4D	SA	MRC_FORCE_OLTM
0xDD50	SA	MRC_MC_ACTIVATE
0xDD51	SA	MRC_RH_PREVENTION
0xDD52	SA	MRC_GET_MRC_DATA
0xDD53	SA	MRC_WRITE_DQ_DQS_RETRAINING
0xDD58	SA	MRC_RETRAIN_CHECK
0xDD5A	SA	MRC_SA_GV_SWITCH
0xDD5B	SA	MRC_ALIAS_CHECK
0xDD5C	SA	MRC_ECC_CLEAN_START
0xDD5D	SA	MRC_DONE
0xDD5F	SA	MRC_CPGC_MEMORY_TEST
0xDD60	SA	MRC_TXT_ALIAS_CHECK
0xDD61	SA	MRC_ENG_PERF_GAIN
0xDD68	SA	MRC_MEMORY_TEST
0xDD69	SA	MRC_FILL_BDAT_STRUCTURE
0xDD70	SA	MRC_SELF_REFRESH_EXIT
0xDD71	SA	MRC_NORMAL_MODE
0xDD7D	SA	Reserved
0xDD7F	SA	MRC_MARGIN_LIMIT_CHECK
0xDD90	SA	MRC_CMD_PLOT_2D_ERROR
0xDD9B	SA	MRC_FAST_BOOT_PERMITTED_ERROR
0xDD9C	SA	MRC_RESTORE_NON_TRAINING_ERROR
0xDD9D	SA	MRC_PRINT_INPUT_PARAMS_ERROR
0xDD9E	SA	MRC_SET_OVERRIDES_PSPD_ERROR
0xDDA0	SA	MRC_SPD_PROCESSING_ERROR
0xDDA1	SA	MRC_SET_OVERRIDES_ERROR
0xDDA2	SA	MRC_MC_CAPABILITY_ERROR
0xDDA3	SA	MRC_MC_CONFIG_ERROR
0xDDA4	SA	MRC_MC_MEMORY_MAP_ERROR
0xDDA5	SA	MRC_JEDEC_INIT_LPDDR3_ERROR
0xDDA6	SA	MRC_RESET_ERROR
0xDDA7	SA	MRC_PRE_TRAINING_ERROR
0xDDA8	SA	MRC_EARLY_COMMAND_ERROR
0xDDA9	SA	MRC_SENSE_AMP_OFFSET_ERROR
0xDDAA	SA	MRC_READ_MPR_ERROR
0xDDAB	SA	MRC_RECEIVE_ENABLE_ERROR
0xDDAC	SA	MRC_JEDEC_WRITE_LEVELING_ERROR
0xDDAD	SA	MRC_LPDDR_LATENCY_SET_B_ERROR
0xDDAE	SA	MRC_WRITE_TIMING_1D_ERROR
0xDDAF	SA	MRC_READ_TIMING_1D_ERROR
0xDDB0	SA	MRC_DIMM_ODT_ERROR
0xDDB1	SA	MRC_EARLY_WRITE_TIMING_ERROR
0xDDB2	SA	MRC_WRITE_DS_ERROR
0xDDB3	SA	MRC_WRITE_EQ_ERROR
0xDDB4	SA	MRC_EARLY_READ_TIMING_ERROR
0xDDB5	SA	MRC_READ_ODT_ERROR

PostCode	Module	Description
0xDDB6	SA	MRC_READ_EQ_ERROR
0xDDB7	SA	MRC_READ_AMP_POWER_ERROR
0xDDB8	SA	MRC_WRITE_TIMING_2D_ERROR
0xDDB9	SA	MRC_READ_TIMING_2D_ERROR
0xDDBA	SA	MRC_CMD_VREF_ERROR
0xDDBB	SA	MRC_WRITE_VREF_2D_ERROR
0xDDBC	SA	MRC_READ_VREF_2D_ERROR
0xDDBD	SA	MRC_POST_TRAINING_ERROR
0xDDBE	SA	MRC_LATE_COMMAND_ERROR
0xDDBF	SA	MRC_ROUND_TRIP_LAT_ERROR
0xDDC0	SA	MRC_TURN_AROUND_ERROR
0xDDC1	SA	MRC_CMP_OPT_ERROR
0xDDC2	SA	MRC_SAVE_MC_VALUES_ERROR
0xDDC3	SA	MRC_RESTORE_TRAINING_ERROR
0xDDC4	SA	MRC_RMT_TOOL_ERROR
0xDDC5	SA	MRC_WRITE_SR_ERROR
0xDDC6	SA	MRC_DIMM_RON_ERROR
0xDDC7	SA	MRC_RCVEN_TIMING_1D_ERROR
0xDDC8	SA	MRC_MR_FILL_ERROR
0xDDC9	SA	MRC_PWR_MTR_ERROR
0xDDCA	SA	MRC_DDR4_MAPPING_ERROR
0xDDCB	SA	MRC_WRITE_VOLTAGE_1D_ERROR
0xDDCC	SA	MRC_EARLY_RDMPR_TIMING_2D_ERROR
0xDDCD	SA	MRC_FORCE_OLTM_ERROR
0xDDD0	SA	MRC_MC_ACTIVATE_ERROR
0xDDD1	SA	MRC_RH_PREVENTION_ERROR
0xDDD2	SA	MRC_GET_MRC_DATA_ERROR
0xDDD3	SA	Reserved
0xDDD8	SA	MRC_RETRAIN_CHECK_ERROR
0xDDDA	SA	MRC_SA_GV_SWITCH_ERROR
0xDDDB	SA	MRC_ALIAS_CHECK_ERROR
0xDDDC	SA	MRC_ECC_CLEAN_ERROR
0xDDDD	SA	MRC_DONE_WITH_ERROR
0xDDDF	SA	MRC_CPGC_MEMORY_TEST_ERROR
0xDDE0	SA	MRC_TXT_ALIAS_CHECK_ERROR
0xDDE1	SA	MRC_ENG_PERF_GAIN_ERROR
0xDDE8	SA	MRC_MEMORY_TEST_ERROR
0xDDE9	SA	MRC_FILL_RMT_STRUCTURE_ERROR
0xDDF0	SA	MRC_SELF_REFRESH_EXIT_ERROR
0xDDF1	SA	MRC_MRC_NORMAL_MODE_ERROR
0xDDFD	SA	MRC_SSA_PRE_STOP_POINT_ERROR
0xDDFE	SA	MRC_NO_MEMORY_DETECTED

8.1.3 TempRamExit API Status Codes (0xBxxx)

PostCode	Module	Description
0xB800	FSP	TempRamExit API Entry
0xB87F	FSP	TempRamExit API Exit

8.1.4 FspSiliconInit API Status Codes (0x9xxx)

PostCode	Module	Description
0x9800	FSP	FspSiliconInit API Entry
0x987F	FSP	FspSiliconInit API Exit
0x9A00	SA	PostMem Salnit Entry
0x9A01	SA	DeviceConfigure Start
0x9A02	SA	UpdateSaHobPostMem Start
0x9A03	SA	Initializing Pei Display
0x9A04	SA	PeiGraphicsNotifyCallback Entry
0x9A05	SA	CallPpiAndFillFrameBuffer
0x9A06	SA	GraphicsPpiInit
0x9A07	SA	GraphicsPpiGetMode
0x9A08	SA	FillFrameBufferAndShowLogo
0x9A0F	SA	PeiGraphicsNotifyCallback Exit
0x9A14	SA	Initializing SA IPU device
0x9A16	SA	Initializing SA GNA device
0x9A1A	SA	SaProgramLlcWays Start
0x9A20	SA	Initializing PciExpressInitPostMem
0x9A22	SA	Initializing ConfigureNorthIntelTraceHub
0x9A30	SA	Initializing Vtd
0x9A31	SA	Initializing TCSS
0x9A32	SA	Initializing Pvp
0x9A34	SA	PeiInstallSmmAccessPpi Start
0x9A36	SA	EdramWa Start
0x9A4F	SA	Post-Mem Salnit Exit
0x9A50	SA	SaSecurityLock Start
0x9A5F	SA	SaSecurityLock End
0x9A60	SA	SaSResetComplete Entry
0x9A61	SA	Set BIOS_RESET_CPL to indicate all configurations complete
0x9A62	SA	SaSvInit2 Start
0x9A63	SA	GraphicsPmInit Start
0x9A64	SA	SaPciPrint Start
0x9A6F	SA	SaSResetComplete Exit
0x9A70	SA	SaS3ResumeAtEndOfPei Callback Entry
0x9A7F	SA	SaS3ResumeAtEndOfPei Callback Exit
0x9B00	PCH	Post-Mem PchInit Entry
0x9B03	PCH	Post-Mem Tune the USB 2.0 high-speed signals quality
0x9B04	PCH	Post-Mem Tune the USB 3.0 signals quality
0x9B05	PCH	Post-Mem Configure PCH xHCI
0x9B06	PCH	Post-Mem Performs configuration of PCH xHCI SSIC
0x9B07	PCH	Post-Mem Configure PCH xHCI after init
0x9B08	PCH	Post-Mem Configures PCH USB device (xHCI)
0x9B0A	PCH	Post-Mem DMI/OP-DMI configuration
0x9B0B	PCH	Post-Mem Initialize P2SB controller
0x9B0C	PCH	Post-Mem IOAPIC initialization
0x9B0D	PCH	Post-Mem PCH devices interrupt configuration
0x9B0E	PCH	Post-Mem HD Audio initialization
0x9B0F	PCH	Post-Mem HD Audio Codec enumeration
0x9B10	PCH	Post-Mem HD Audio Codec not detected

PostCode	Module	Description
0x9B13	PCH	Post-Mem SCS initialization
0x9B14	PCH	Post-Mem ISH initialization
0x9B15	PCH	Post-Mem Configure SMBUS power management
0x9B16	PCH	Post-Mem Reserved
0x9B17	PCH	Post-Mem Performing global reset
0x9B18	PCH	Post-Mem Reserved
0x9B19	PCH	Post-Mem Reserved
0x9B40	PCH	Post-Mem OnEndOfPEI Entry
0x9B41	PCH	Post-Mem Initialize Thermal controller
0x9B42	PCH	Post-Mem Configure Memory Throttling
0x9B47	PCH	Post-Mem OnEndOfPEI Exit
0x9B4D	PCH	Post-Mem Trace Hub - Memory configuration
0x9B4E	PCH	Post-Mem Trace Hub - MSC0 configured
0x9B4F	PCH	Post-Mem Trace Hub - MSC1 configured
0x9B7F	PCH	Post-Mem PchInit Exit
0x9C00	CPU	CPU Post-Mem Entry
0x9C09	CPU	CpuAddConfigBlocks Done
0x9C0A	CPU	SetCpuStrapAndEarlyPowerOnConfig Start
0x9C13	CPU	SetCpuStrapAndEarlyPowerOnConfig Reset
0x9C14	CPU	SetCpuStrapAndEarlyPowerOnConfig Done
0x9C15	CPU	Cpulnit Start
0x9C17	CPU	CollectProcessorFeature Start
0x9C18	CPU	ProgramProcessorFeature Start
0x9C19	CPU	ProgramProcessorFeature Done
0x9C20	CPU	CpulnitPreResetCpl Start
0x9C21	CPU	ProcessorsPrefetcherInitialization Start
0x9C22	CPU	InitRatl Start
0x9C23	CPU	ConfigureSvidVrs Start
0x9C24	CPU	ConfigurePidSettings Start
0x9C25	CPU	SetBootFrequency Start
0x9C26	CPU	CpuOclnitPreMem Start
0x9C27	CPU	CpuOclnit Reset
0x9C28	CPU	BiosGuardInit Start
0x9C29	CPU	BiosGuardInit Reset
0x9C3F	CPU	CpulnitPreResetCpl Done
0x9C42	CPU	ApSafePostMicrocodePatchInit Start
0x9C43	CPU	InitializeCpuDataHob Start
0x9C44	CPU	InitializeCpuDataHob Done
0x9C4F	CPU	Cpulnit Done
0x9C50	CPU	S3InitializeCpu Start
0x9C55	CPU	MpRendezvousProcedure Start
0x9C56	CPU	MpRendezvousProcedure Done
0x9C69	CPU	S3InitializeCpu Done
0x9C6A	CPU	CpuPowerMgmtInit Start
0x9C71	CPU	InitPpm
0x9C7F	CPU	CPU Post-Mem Exit
0x9C80	CPU	ReloadMicrocodePatch Start
0x9C81	CPU	ReloadMicrocodePatch Done
0x9C82	CPU	ApSafePostMicrocodePatchInit Start

PostCode	Module	Description
0x9C83	CPU	ApSafePostMicrocodePatchInit Done

8.1.5 NotifyPhase API Status Codes (0x6xxx)

PostCode	Module	Description
0x6800	FSP	NotifyPhase API Entry
0x687F	FSP	NotifyPhase API Exit

Chapter 9

Deprecated List

Member `CPU_CONFIG::Avx3Disable`

Enable or Disable Avx3.

- **1: Disable**
- 0: Enable

Member `CPU_CONFIG_LIB_PREMEM_CONFIG::ActiveCoreCount`

due to core active number limitaion.

Member `CPU_CONFIG_LIB_PREMEM_CONFIG::PeciC10Reset`

no longer used.

Member `CPU_POWER_MGMT_BASIC_CONFIG::EnableItbmDriver`

: Platform doesn't have Intel Turbo Boost Max Technology 3.0 Driver Enabling it will load the driver upon ACPI device with HID = INT3510. **0: Disable**; 1: Enable;

Member `CPU_POWER_MGMT_TEST_CONFIG::C3AutoDemotion`

no longer used

Member `CPU_POWER_MGMT_TEST_CONFIG::C3UnDemotion`

no longer used

Member `CPU_POWER_MGMT_TEST_CONFIG::ConfigTdpLevel`

Move to premem phase.

Member `CPU_SECURITY_PREMEM_CONFIG::EnableSgx`

due to Non-POR feature.

Member `AMT_PEI_CONFIG::ManageabilityMode`

: Manageability Mode sync with Mebx, 0: Disabled; **1: AMT**

Member `AMT_PEI_CONFIG::RemoteAssistance`

: Remote Assistance is enabled if platform is provisioned. **0: Disable**, 1: Enable

Member `AMT_PEI_CONFIG::AmtKvmEnabled`

: KVM retrieved from Mebx. The default value depends on CSME/AMT. 0: Disable, **1: Enable**

Member `AMT_DXE_CONFIG::MebxDebugMsg`

: OEMFlag Bit 14: Enable OEM debug menu in MEBx. **0: Disable**; 1: Enable

Member `AMT_DXE_CONFIG::HideUnConfigureMeConfirm`

: OEMFlag Bit 6: Hide Unconfigure ME confirmation prompt when attempting ME unconfiguration. **0: Don't hide**; 1: Hide

Member `AMT_DXE_CONFIG::AmtbxHotkeyPressed`

: OEMFlag Bit 1: Enable automatic MEBx hotkey press. **0: Disable**; 1: Enable

Member [AMT_DXE_CONFIG::AmtbxSelectionScreen](#)

Member [AMT_DXE_CONFIG::CiraTimeout](#)

Member [AMT_DXE_CONFIG::CpuReplacementTimeout](#)

Member [AMT_DXE_CONFIG::MebxNonUiTextMode](#)

: Resolution for non-UI text mode. **0: Auto**; 1: 80x25; 2: 100x31

Member [AMT_DXE_CONFIG::MebxUiTextMode](#)

: Resolution for UI text mode. **0: Auto**; 1: 80x25; 2: 100x31

Member [AMT_DXE_CONFIG::MebxGraphicsMode](#)

: Resolution for graphics mode. **0: Auto**; 1: 640x480; 2: 800x600; 3: 1024x768

Member [AMT_DXE_CONFIG::OemResolutionSettingsRsvd](#)

: Reserved for future use & Config block alignment

Member [CpuPcieEqDefault](#)

since revision 3. Behaves as PchPcieEqHardware.

Member [FIVR_EXT_RAIL_CONFIG::IccMax](#)

THIS POLICY IS DEPRECATED, PLEASE USE IccMaximum INSTEAD VR rail Icc Max Value Granularity of this setting is 1mA and maximal possible value is 500mA The default is **0mA**.

Member [GRAPHICS_DXE_CONFIG::IgdDvmtMemSize](#)

IGD DVMT Memory Size: 1=128MB, **2=256MB**, 3=MAX

Member [ME_PEI_CONFIG::Heci3Enabled](#)

Member [SI_CONFIG::CsmFlag](#)

Member [CPU_POWER_MGMT_VR_CONFIG::TdcTimeWindow](#) [MAX_NUM_VRS]

. PCODE MMIO Mailbox: Thermal Design Current time window. Defined in milli seconds. **1ms default**

Member [SA_MISC_PEI_PREMEM_CONFIG::IedSize](#)

Chapter 10

Todo List

Member [CPU_POWER_MGMT_TEST_CONFIG::CstateLatencyControl0TimeUnit](#)

: The following enums have to be replaced with policies.

Page [Config Blocks](#)

This list is from TigerLake, needs to be updated for AlderLake

This list is from TigerLake, needs to be updated for AlderLake

Chapter 11

Class Index

11.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

_CONFIG_BLOCK	
Config Block	59
_CONFIG_BLOCK_HEADER	
Config Block Header	60
_CONFIG_BLOCK_TABLE_STRUCT	
Config Block Table Header	61
_EFI_MM_RESERVED_MMRAM_REGION	
Structure describing a MMRAM region which cannot be used for the MMRAM heap	62
_EFI_PEI_MP_SERVICES_PPI	
This PPI is installed by some platform or chipset-specific PEIM that abstracts handling multipro- cessor support	63
_ITBT_GENERIC_CONFIG	
ITBT Controller Data Structure to be used cross to RP and controller to be shared by CONF↔ G_BLOCK and HOB	63
_ITBT_ROOTPORT_CONFIG	
ITBT RootPort Data Structure	64
_LIST_ENTRY	
_LIST_ENTRY structure definition	65
_PEI_ITBT_CONFIG	
ITBT PEI configuration	
Revision 1:	66
_PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI	
This PPI provides function to install default silicon policy	67
_PEI_SI_DEFAULT_POLICY_INIT_PPI	
This PPI provides function to install default silicon policy	67
_PPM_CUSTOM_CTDp_TABLE	
PPM Custom ConfigTdp Settings	68
_SI_POLICY_STRUCT	
SI Policy PPI	
All SI config block change history will be listed here	
68	
_SI_PREMEM_POLICY_STRUCT	
SI Policy PPI in Pre-Mem	
All SI config block change history will be listed here	

ADR_CONFIG	
ADR Configuration Revision 1 : - Initial version	71
ADR_SOURCE_ENABLE	
ADR Source Enable	72
AMT_DXE_CONFIG	
AMT Dxe Configuration Structure	73
AMT_PEI_CONFIG	
AMT Pei Configuration Structure	77
BIOS_GUARD_CONFIG	
BIOS Guard Configuration Structure	80
CNVI_CONFIG	
The CNVI_CONFIG block describes the expected configuration of the CNVi IP	81
CNVI_PIN_MUX	
CNVi signals pin muxing settings	83
CNVI_PREMEM_CONFIG	
The CNVI_PREMEM_CONFIG block describes the expected configuration of the CNVi IP	84
CPU_CONFIG	
CPU Configuration Structure	85
CPU_CONFIG_LIB_PREMEM_CONFIG	
CPU Config Library PreMemory Configuration Structure	89
CPU_DMI_EQ_PARAM	
Represent lane specific Dmi Gen3 equalization parameters	97
CPU_DMI_PREMEM_CONFIG	
The CPU_DMI_CONFIG block describes the expected configuration of the CPU for DMI	98
CPU_PCIE_CONFIG	
The CPU_PCIE_CONFIG block describes the expected configuration of the CPU PCI Express controllers Revision 1 < / b>: -Initial version	101
CPU_PCIE_EQ_LANE_PARAM	
Represent lane specific PCIe Gen3 equalization parameters	104
CPU_PCIE_GPIO_INFO	
CPU PCIe GPIO Data Structure	105
CPU_PCIE_ROOT_PORT_CONFIG	
The CPU_PCIE_ROOT_PORT_CONFIG describe the feature and capability of each CPU PCIe root port	105
CPU_PCIE_RP_PREMEM_CONFIG	
CPU PCIe Root Port Pre-Memory Configuration Contains Root Port settings and capabilities Revision 1 : - Initial version	107
CPU_PCIE_RTD3_GPIO	
CPU PCIe RTD3 GPIO Data Structure	110
CPU_POWER_MGMT_BASIC_CONFIG	
CPU Power Management Basic Configuration Structure	111
CPU_POWER_MGMT_CUSTOM_CONFIG	
CPU Power Management Custom Configuration Structure	123
CPU_POWER_MGMT_PSYS_CONFIG	
CPU Power Management Psys(Platform) Configuration Structure	124
CPU_POWER_MGMT_TEST_CONFIG	
CPU Power Management Test Configuration Structure	126
CPU_POWER_MGMT_VR_CONFIG	
CPU Power Management VR Configuration Structure	131
CPU_SECURITY_PREMEM_CONFIG	
CPU Security PreMemory Configuration Structure	138
CPU_TEST_CONFIG	
CPU Test Configuration Structure	141
CPU_TRACE_HUB_PREMEM_CONFIG	
CPU Trace Hub PreMem Configuration Contains Trace Hub settings for CPU side tracing Revision 1 : - Initial version	143
CPU_TXT_PREMEM_CONFIG	
CPU TXT PreMemory Configuration Structure	144

DDI_CONFIGURATION	
This structure configures the Native GPIOs for DDI port per VBT settings	145
DMI_HW_WIDTH_CONTROL	
This structure allows to customize DMI HW Autonomous Width Control for Thermal and Mechanical spec design	146
EFI_HOB_CPU	
Describes processor information, such as address space and I/O space capabilities	147
EFI_HOB_FIRMWARE_VOLUME	
Details the location of firmware volumes that contain firmware files	148
EFI_HOB_FIRMWARE_VOLUME2	
Details the location of a firmware volume that was extracted from a file within another firmware volume	149
EFI_HOB_FIRMWARE_VOLUME3	
Details the location of a firmware volume that was extracted from a file within another firmware volume	151
EFI_HOB_GENERIC_HEADER	
Describes the format and size of the data inside the HOB	153
EFI_HOB_GUID_TYPE	
Allows writers of executable content in the HOB producer phase to maintain and manage HOBs with specific GUID	153
EFI_HOB_HANDOFF_INFO_TABLE	
Contains general state information used by the HOB producer phase	154
EFI_HOB_MEMORY_ALLOCATION	
Describes all memory ranges used during the HOB producer phase that exist outside the HOB list	156
EFI_HOB_MEMORY_ALLOCATION_BSP_STORE	
Defines the location of the boot-strap processor (BSP) BSPStore ("Backing Store Pointer Store")	157
EFI_HOB_MEMORY_ALLOCATION_HEADER	
EFI_HOB_MEMORY_ALLOCATION_HEADER describes the various attributes of the logical memory allocation	158
EFI_HOB_MEMORY_ALLOCATION_MODULE	
Defines the location and entry point of the HOB consumer phase	160
EFI_HOB_MEMORY_ALLOCATION_STACK	
Describes the memory stack that is produced by the HOB producer phase and upon which all post-memory-installed executable content in the HOB producer phase is executing	161
EFI_HOB_MEMORY_POOL	
Describes pool memory allocations	162
EFI_HOB_RESOURCE_DESCRIPTOR	
Describes the resource properties of all fixed, nonrelocatable resource ranges found on the processor host bus during the HOB producer phase	163
EFI_HOB_UEFI_CAPSULE	
Each UEFI capsule HOB details the location of a UEFI capsule	165
EFI_IP_ADDRESS	
16-byte buffer aligned on a 4-byte boundary	166
EFI_MAC_ADDRESS	
32-byte buffer containing a network Media Access Control address	167
EFI_MMRAM_DESCRIPTOR	
Structure describing a MMRAM region and its accessibility attributes	167
EFI_PEI_HOB_POINTERS	
Union of all the possible HOB Types	169
EFI_TIME	
EFI Time Abstraction: Year: 1900 - 9999 Month: 1 - 12 Day: 1 - 31 Hour: 0 - 23 Minute: 0 - 59 Second: 0 - 59 Nanosecond: 0 - 999,999,999 TimeZone: -1440 to 1440 or 2047	169
FIRMWARE_VERSION	
Firmware Version Structure	170
FIRMWARE_VERSION_INFO	
Firmware Version Information Structure	170
FIRMWARE_VERSION_INFO_HOB	
Firmware Version Information HOB Structure	171

FIVR_EXT_RAIL_CONFIG	Structure for V1p05/Vnn VR rail configuration	172
FIVR_VCCIN_AUX_CONFIG	Structure for VCCIN_AUX voltage rail configuration	173
FSP_ERROR_INFO_HOB	FSP Error Information Block	175
FSPM_ARCH_CONFIG_PPI	This PPI provides FSP-M Arch Config PPI	176
GBE_CONFIG	PCH intergrated GBE controller configuration settings	177
GNA_CONFIG	GNA config block for configuring GNA	178
GPIO_CONFIG	GPIO configuration structure used for pin programming	179
GRAPHICS_DXE_CONFIG	This configuration block is to configure IGD related variables used in DXE	183
GRAPHICS_PEI_CONFIG	This configuration block is to configure IGD related variables used in PostMem PEI	185
GRAPHICS_PEI_PREMEM_CONFIG	This Configuration block is to configure GT related PreMem data/variables	187
GUID	128 bit buffer containing a unique identifier value	190
HDA_LINK_DMIC	HD Audio DMIC Interface Policies	190
HDA_LINK_HDA	HD Audio Link Policies	191
HDA_LINK_SNDW	HD Audio SNDW Interface Policies	191
HDA_LINK_SSP	HD Audio SSP Interface Policies	192
HDA_VERB_TABLE_HEADER	Azalia verb table header Every verb table should contain this defined header and followed by azalia verb commands	192
HDAUDIO_CONFIG	This structure contains the policies which are related to HD Audio device (cAVS)	193
HDAUDIO_DXE_CONFIG	This structure contains the DXE policies which are related to HD Audio device (cAVS)	195
HDAUDIO_PREMEM_CONFIG	This structure contains the premem policies which are related to HD Audio device (cAVS)	196
HSIO_PARAMETERS	This structure describes USB3 Port N configuration parameters	198
HYBRID_GRAPHICS_CONFIG	This Configuration block configures CPU PCI Express 0/1/2 RTD3 GPIOs & Root Port	200
HYBRID_STORAGE_CONFIG	The HYBRID_STORAGE_CONFIG block describes the expected configuration for Hybrid Storage device	202
I2C_PIN_MUX	I2C signals pin muxing settings	203
IEH_CONFIG	The IEH_CONFIG block describes the expected configuration of the PCH Integrated Error Handler	203
IOM_AUX_ORI_PAD_CONFIG	The IOM_AUX_ORI_PAD_CONFIG describes IOM TypeC port map GPIO pin	204
IOM_INTERFACE_CONFIG	The IOM_EC_INTERFACE_CONFIG block describes interaction between BIOS and IOM-EC	205
IPU_PREMEM_CONFIG	IPU PreMem configuration	
Revision 1:		205

IPv4_ADDRESS	
4-byte buffer	207
IPv6_ADDRESS	
16-byte buffer	208
ISH_CONFIG	
The ISH_CONFIG block describes Integrated Sensor Hub device	208
ISH_GP	
Struct contains GPIO pins assigned and signal settings of GP	209
ISH_GPIO_CONFIG	
ISH GPIO settings	210
ISH_I2C	
Struct contains GPIO pins assigned and signal settings of I2C	211
ISH_I2C_PIN_CONFIG	
I2C signals settings	212
ISH_I3C_CONFIG	
The ISH_I3C_CONFIG block describes I3C in ISH device	213
ISH_PREMEM_CONFIG	
Premem Policy for Integrated Sensor Hub device	214
ISH_SPI	
Struct contains GPIO pins assigned and signal settings of SPI	215
ISH_SPI_PIN_CONFIG	
SPI signals settings	216
ISH_UART	
Struct contains GPIO pins assigned and signal settings of UART	217
ISH_UART_PIN_CONFIG	
UART signals settings	218
ME_PEI_CONFIG	
ME Pei Post-Memory Configuration Structure	219
ME_PEI_PREMEM_CONFIG	
ME Pei Pre-Memory Configuration Structure	221
MEMORY_CONFIG_NO_CRC	
Memory Configuration The contents of this structure are not CRC'd by the MRC for option change detection	223
MEMORY_CONFIGURATION	
Memory Configuration The contents of this structure are CRC'd by the MRC for option change detection	225
OVERCLOCKING_PREMEM_CONFIG	
Overclocking Configuration Structure	239
PCH_DCI_PREMEM_CONFIG	
The PCH_DCI_PREMEM_CONFIG block describes policies related to Direct Connection Interface (DCI)	256
PCH_DEVICE_INTERRUPT_CONFIG	
The PCH_DEVICE_INTERRUPT_CONFIG block describes interrupt pin, IRQ and interrupt mode for PCH device	258
PCH_DMI_CONFIG	
The PCH_DMI_CONFIG block describes the expected configuration of the PCH for DMI	259
PCH_DMI_EQ_PARAM	
Lane specific Dmi Gen3, Gen4 equalization parameters	261
PCH_ESPI_CONFIG	
This structure contains the policies which are related to ESPI	261
PCH_FIVR_CONFIG	
The PCH_FIVR_CONFIG block describes FIVR settings	263
PCH_GENERAL_CONFIG	
PCH General Configuration Revision 1: - Initial version	264
PCH_GENERAL_PREMEM_CONFIG	
PCH General Pre-Memory Configuration Revision 1: - Initial version	266
PCH_INTERRUPT_CONFIG	
The PCH_INTERRUPT_CONFIG block describes interrupt settings for PCH	269

PCH_IOAPIC_CONFIG	
The PCH_IOAPIC_CONFIG block describes the expected configuration of the PCH IO APIC, it's optional and PCH code would ignore it if the BdfValid bit is not TRUE	270
PCH_LOCK_DOWN_CONFIG	
The PCH_LOCK_DOWN_CONFIG block describes the expected configuration of the PCH for security requirement	272
PCH_MEMORY_THROTTLING	
This structure supports an external memory thermal sensor (TS-on-DIMM or TS-on-Board) . .	275
PCH_P2SB_CONFIG	
This structure contains the policies which are related to P2SB device	276
PCH_PCIE_CLOCK	
PCH_PCIE_CLOCK describes PCIe source clock generated by PCH	277
PCH_PCIE_CONFIG	
The PCH_PCIE_CONFIG block describes the expected configuration of the PCH PCI Express controllers Revision 1:	278
PCH_PCIE_ROOT_PORT_CONFIG	
The PCH_PCIE_ROOT_PORT_CONFIG describe the feature and capability of each PCH PCIe root port	279
PCH_PCIE_RP_PREMEM_CONFIG	
The PCH_PCIE_RP_PREMEM_CONFIG block describes early configuration of the PCH PCI Express controllers Revision 1:	281
PCH_PM_CONFIG	
The PCH_PM_CONFIG block describes expected miscellaneous power management settings	282
PCH_SATA_PORT_CONFIG	
This structure configures the features, property, and capability for each SATA port	289
PCH_SMBUS_PREMEM_CONFIG	
The PCH_SMBUS_CONFIG block lists the reserved addresses for non-ARP capable devices in the platform	291
PCH_TRACE_HUB_PREMEM_CONFIG	
PCH Trace Hub PreMem Configuration Contains Trace Hub settings for PCH side tracing Revision 1: - Initial version	293
PCH_WAKE_CONFIG	
This structure allows to customize PCH wake up capability from S5 or DeepSx by WOL, LAN, PCIe wake events	294
PCH_WDT_PREMEM_CONFIG	
This policy clears status bits and disable watchdog, then lock the WDT registers	295
PCIE_COMMON_CONFIG	
PCIe Common Config	296
PCIE_DEVICE_OVERRIDE	
PCIe device table entry entry	298
PCIE_EQ_PARAM	
Represent lane specific PCIe Gen3 equalization parameters	301
PCIE_IMR_CONFIG	
PCIe IMR Config	301
PCIE_LINK_EQ_PLATFORM_SETTINGS	
PCIe Link EQ Platform Settings	302
PCIE_PEI_PREMEM_CONFIG	
PCI Express and DMI controller configuration	305
PCIE_PREMEM_CONFIG	
PCIe Pre-Memory Configuration Revision 1: - Initial version	308
PCIE_RP_DXE_CONFIG	
The PCIE_RP_DXE_CONFIG block describes the expected configuration of the PCH PCI Express controllers in DXE phase	309
PMC_GLOBAL_RESET_MASK	
Description of Global Reset Trigger/Event Mask register	311
PMC_INTERFACE_CONFIG	
The PMC_INTERFACE_CONFIG block describes interaction between BIOS and PMC	311

PMC_LPM_S0IX_SUB_STATE_EN	
Low Power Mode Enable config	312
PPM_CUSTOM_RATIO_TABLE	
This structure is used to describe the custom processor ratio table desired by the platform . . .	313
PSF_CONFIG	
The PSF_CONFIG block describes the expected configuration of the Primary Sideband Fabric	314
RST_CONFIG	
Rapid Storage Technology settings	316
RST_HARDWARE_REMAPPED_STORAGE_CONFIG	
This structure describes the details of Intel RST for PCIe Storage remapping Note: In order to use this feature, Intel RST Driver is required	318
RTC_CONFIG	
The RTC_CONFIG block describes the expected configuration of RTC configuration	319
SA_ADDRESS_DECODE	
SA memory address decode	321
SA_FUNCTION_CALLS	
Function calls into the SA	321
SA_MEMORY_DQDQS_MAPPING	
DqDqs Mapping	324
SA_MEMORY_FUNCTIONS	
Function calls into the MRC	324
SA_MEMORY_RCOMP	
Rcomp Policies	325
SA_MISC_PEI_CONFIG	
This configuration block is to configure SA Miscellaneous variables during PEI Post-Mem . . .	326
SA_MISC_PEI_PREMEM_CONFIG	
This configuration block is to configure SA Miscellaneous variables during PEI Pre-Mem phase like programming different System Agent BARs, TsegSize, MmioSize required etc	327
SA_XDCI_IRQ_INT_CONFIG	
The SA XDCI INT Pin and IRQ number	331
SATA_CONFIG	
The SATA_CONFIG block describes the expected configuration of the SATA controllers	332
SATA_THERMAL_THROTTLING	
This structure lists PCH supported SATA thermal throttling register setting for customization . .	334
SERIAL_IO_CONFIG	
The SERIAL_IO_CONFIG block provides the configurations to set the Serial IO controllers . .	336
SERIAL_IO_I2C_CONFIG	
Serial IO I2C Controller Configuration	337
SERIAL_IO_SPI_CONFIG	
The SERIAL_IO_SPI_CONFIG provides the configurations to set the Serial IO SPI controller .	338
SERIAL_IO_UART_ATTRIBUTES	
UART Settings	339
SERIAL_IO_UART_CONFIG	
Serial IO UART Controller Configuration	339
SI_CONFIG	
The Silicon Policy allows the platform code to publish a set of configuration information that the RC drivers will use to configure the silicon hardware	340
SI_PREMEM_CONFIG	
The Silicon PreMem Policy allows the platform code to publish a set of configuration information that the RC drivers will use to configure the silicon hardware	344
SMBIOS_STRUCTURE	
The Smbios structure header	346
SPD_DATA_BUFFER	
SPD Data Buffer	346
SPD_OFFSET_TABLE	
SPD Offset Table	346
SPI_PIN_MUX	
SPI signals pin muxing settings	347

SVID_SID_VALUE	
Subsystem Vendor ID / Subsystem ID	348
TCSS_DEVEN_PEI_PREMEM_CONFIG	
The TCSS_DEVEN_PEI_PREMEM_CONFIG block describes Device Enable settings for TCSS	348
TCSS_IOM_ORI_OVERRIDE	
The TCSS_IOM_PEI_CONFIG block describes IOM Aux/HSL override settings for TCSS	348
TCSS_IOM_PEI_CONFIG	
The TCSS_IOM_PEI_CONFIG block describes IOM settings for TCSS	349
TCSS_MISC_PEI_CONFIG	
The TCSS_MISC_PEI_CONFIG block describes MISC settings for TCSS	350
TCSS_MISC_PEI_PREMEM_CONFIG	
The TCSS_MISC_PEI_PREMEM_CONFIG block describes MISC settings for TCSS	351
TCSS_PCIE_PEI_POLICY	
TCSS_PCIE_PEI_POLICY describes PCIe port settings for TCSS	352
TCSS_PCIE_PORT_POLICY	
The TCSS_PCIE_PORT_POLICY block describes PCIe settings for TCSS	353
TCSS_PEI_CONFIG	
The TCSS_PEI_CONFIG block describes TCSS settings for SA	354
TCSS_PEI_PREMEM_CONFIG	
This configuration block describes TCSS settings	355
TCSS_USBTC_PEI_PERMEM_CONFIG	
The TCSS_USBTC_PEI_PERMEM_CONFIG block describes IOM settings for TCSS	356
TELEMETRY_PEI_CONFIG	
This configuration block describes Telemetry settings in PostMem	357
TELEMETRY_PEI_PREMEM_CONFIG	
This configuration block describes Telemetry settings in PreMem	358
THC_CONFIG	
THC_CONFIG block provides the configurations for Touch Host Controllers	359
THC_HID_OVER_SPI	
THC Hid Over SPI mode related settings	360
THC_PORT	
Port Configuration structure required for each Port that THC might use	361
THERMAL_CONFIG	
The THERMAL_CONFIG block describes the expected configuration of the Thermal IP block	362
THERMAL_THROTTLE_LEVELS	
This structure lists PCH supported throttling register setting for customization	364
TRACE_HUB_CONFIG	
TRACE_HUB_CONFIG block describes TraceHub settings	365
TS_GPIO_PIN_SETTING	
This structure configures PCH memory throttling thermal sensor GPIO PIN settings	367
TSN_CONFIG	
The TSN_CONFIG block describes policies related to Time Sensitive Networking(TSN)	368
UART_PIN_MUX	
UART signals pin muxing settings	369
USB2_PHY_CONFIG	
This structure holds info on how to tune electrical parameters of USB2 ports based on board layout	370
USB2_PHY_PARAMETERS	
This structure configures per USB2 AFE settings	371
USB2_PORT_CONFIG	
This structure configures per USB2.0 port settings like enabling and overcurrent protection	372
USB3_HSIO_CONFIG	
Structure for holding USB3 tuning parameters	373
USB3_PORT_CONFIG	
This structure configures per USB3.x port settings like enabling and overcurrent protection	374
USB_CONFIG	
This member describes the expected configuration of the USB controller, Platform modules may need to refer Setup options, schematic, BIOS specification to update this field	375

VMD_PEI_CONFIG	
This configuration block is to configure VMD related variables used in PostMem PEI	378
VTD_CONFIG	
The data elements should be initialized by a Platform Module	380
VTD_DXE_CONFIG	
The data structure is for VT-d driver initialization in DXE	
Revision 1:	382
XDCI_CONFIG	
The XDCI_CONFIG block describes the configurations of the xDCI Usb Device controller . . .	383

Chapter 12

File Index

12.1 File List

Here is a list of all documented files with brief descriptions:

AdrConfig.h	
ADR policy	385
AmtConfig.h	
AMT Config Block for PEI/DXE phase	386
Base.h	
Root include file for Mde Package Base type modules	388
BiosGuardConfig.h	
CPU BIOS Guard Config Block	407
CnviConfig.h	
CNVi policy	409
ConfigBlock.h	
Header file for Config Block Lib implementation	410
ConfigBlockLib.h	
Header file for Config Block Lib implementation	411
CpuConfig.h	
CPU Config Block	413
CpuConfigLibPreMemConfig.h	
CPU Security PreMemory Config Block	414
CpuDmiPreMemConfig.h	
DMI policy	415
CpuPcieConfig.h	
Pcie root port policy	416
CpuPowerMgmtBasicConfig.h	
CPU Power Management Basic Config Block	419
CpuPowerMgmtCustomConfig.h	
CPU Power Management Custom Config Block	419
CpuPowerMgmtPsysConfig.h	
CPU Power Management Psys(Platform) Config Block	421
CpuPowerMgmtTestConfig.h	
CPU Power Management Test Config Block	422
CpuPowerMgmtVrConfig.h	
CPU Power Management VR Config Block	423
CpuSecurityPreMemConfig.h	
CPU Security PreMemory Config Block	424
CpuTestConfig.h	
CPU Test Config Block	425

CpuTxtConfig.h	
CPU TXT PreMemory Config Block	426
DciConfig.h	
Dci policy	427
EspConfig.h	
Esp policy	427
FirmwareVersionInfoHob.h	
Header file for Firmware Version Information	428
FivrConfig.h	
PCH FIVR policy	429
FspErrorInfo.h	
FSP Error Information HOB to describe errors inside FSP that bootloader may take some actions to handle those error scenarios	431
FspFixedPcds.h	
This file lists all FixedAtBuild PCDs referenced in FSP integration guide	431
FspInfoHob.h	
Header file for FSP Information HOB	432
FspmArchConfigPpi.h	
Header file for FSP-M Arch Config PPI	433
GbeConfig.h	
Gigabit Ethernet policy	434
GnaConfig.h	
Policy definition for GNA Config Block	434
GpioConfig.h	
Header file for GpioConfig structure used by GPIO library	435
GpioSampleDef.h	
Copyright (c) 2015 - 2021, Intel Corporation	442
GraphicsConfig.h	
Policy definition for Internal Graphics Config Block	443
HdAudioConfig.h	
HDAUDIO policy	444
HybridGraphicsConfig.h	
Hybrid Graphics policy definitions	446
HybridStorageConfig.h	
Hybrid Storage policy	447
IehConfig.h	
Integrated Error Handler policy	448
InterruptConfig.h	
Interrupt policy	450
IoApicConfig.h	
IoApic policy	451
IpuPreMemConfig.h	
IPU policy definitions	452
IshConfig.h	
ISH policy	453
LockDownConfig.h	
Lock down policy	454
MemoryConfig.h	
Policy definition of Memory Config Block	455
MePeiConfig.h	
ME config block for PEI phase	459
MpServices.h	
This file declares UEFI PI Multi-processor PPI	460
OverclockingConfig.h	
Overclocking Config Block	467
P2sbConfig.h	
P2sb policy	468

PchDmiConfig.h	
DMI policy	469
PchGeneralConfig.h	
PCH General policy	469
PchPcieRpConfig.h	
PCH Pcie root port policy	471
PcieConfig.h	
PCIe Config Block	473
PciePreMemConfig.h	
PCIe Config Block PreMem	475
PeiITbtConfig.h	
Header file for TBT PEI Policy	476
PeiITbtGenericStructure.h	
ITBT Policy definition to be referred in both PEI and DXE phase	478
PeiPreMemSiDefaultPolicy.h	
This file defines the function to initialize default silicon policy PPI	479
PeiSiDefaultPolicy.h	
This file defines the function to initialize default silicon policy PPI	480
PiHob.h	
HOB related definitions in PI	481
PiMultiPhase.h	
Include file matches things in PI for multiple module types	483
PmConfig.h	
Power Management policy	486
PsfConfig.h	
Primary Sideband Fabric policy	488
RstConfig.h	
Rst policy	490
RtcConfig.h	
RTC policy	491
SaMiscPeiConfig.h	
Policy details for miscellaneous configuration in System Agent	492
SaMiscPeiPreMemConfig.h	
Policy details for miscellaneous configuration in System Agent	492
SataConfig.h	
Sata policy	493
SerialIoConfig.h	
Serial IO policy	494
SerialIoDevices.h	
Serial IO policy	496
SiConfig.h	
Si Config Block	500
SiPolicyStruct.h	
Intel reference code configuration policies	501
SiPreMemConfig.h	
Si Config Block PreMem	503
SmbusConfig.h	
Smbus policy	504
TcssPeiConfig.h	
TCSS PEI policy	505
TcssPeiPreMemConfig.h	
TCSS PEI PreMem policy	507
TelemetryPeiConfig.h	
Configurations for Telemetry	508
ThcConfig.h	
Touch Host Controller policy	509
ThermalConfig.h	
Thermal policy	511

TraceHubConfig.h	
Configurations for TraceHub	512
TsnConfig.h	
TSN Config policy	513
UefiBaseType.h	
Defines data types and constants introduced in UEFI	514
Usb2PhyConfig.h	
USB2 PHY configuration policy	518
Usb3HsioConfig.h	
USB3 Mod PHY configuration policy	519
UsbConfig.h	
Common USB policy shared between PCH and CPU Contains general features settings for xHCI and xDCI	520
VmdPeiConfig.h	
VMD PEI policy	521
VtdConfig.h	
VT-d policy definitions	522
WatchDogConfig.h	
WatchDog policy	524

Chapter 13

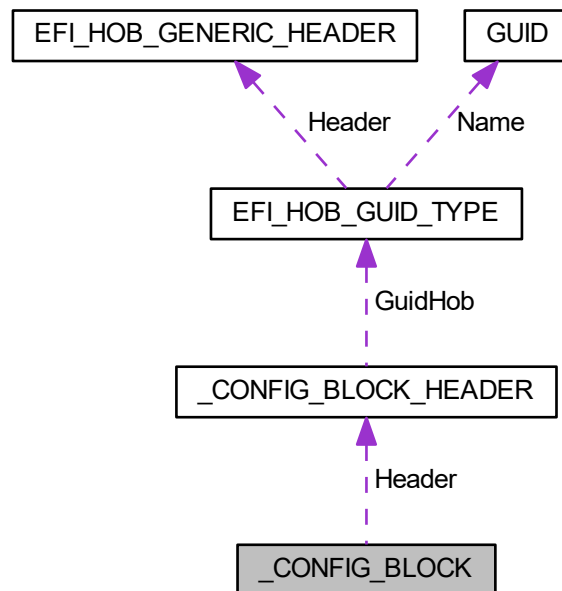
Class Documentation

13.1 _CONFIG_BLOCK Struct Reference

Config Block.

```
#include <ConfigBlock.h>
```

Collaboration diagram for _CONFIG_BLOCK:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Header of config block.

13.1.1 Detailed Description

Config Block.

Definition at line 40 of file ConfigBlock.h.

The documentation for this struct was generated from the following file:

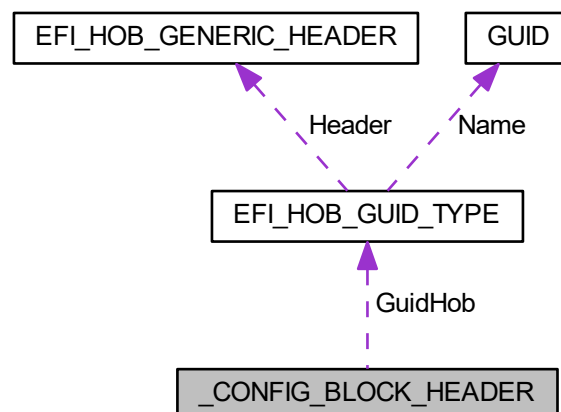
- [ConfigBlock.h](#)

13.2 _CONFIG_BLOCK_HEADER Struct Reference

Config Block Header.

```
#include <ConfigBlock.h>
```

Collaboration diagram for _CONFIG_BLOCK_HEADER:



Public Attributes

- [EFI_HOB_GUID_TYPE](#) [GuidHob](#)
Offset 0-23 [GUID](#) extension HOB header.
- [UINT8](#) [Revision](#)
Offset 24 Revision of this config block.
- [UINT8](#) [Attributes](#)
Offset 25 The main revision for config block.
- [UINT8](#) [Reserved](#) [2]
Offset 26-27 Reserved for future use.

13.2.1 Detailed Description

Config Block Header.

Definition at line 30 of file ConfigBlock.h.

The documentation for this struct was generated from the following file:

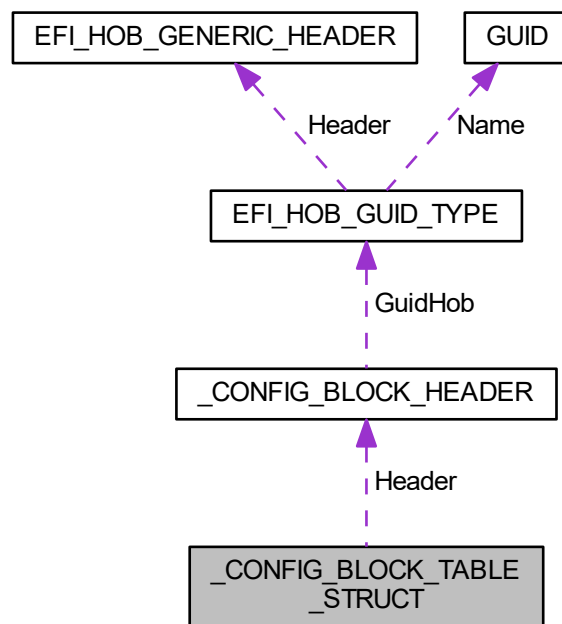
- [ConfigBlock.h](#)

13.3 _CONFIG_BLOCK_TABLE_STRUCT Struct Reference

Config Block Table Header.

```
#include <ConfigBlock.h>
```

Collaboration diagram for _CONFIG_BLOCK_TABLE_STRUCT:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 GUID number for main entry of config block.
- **UINT8** [Rsvd0](#) [2]
Offset 28-29 Reserved for future use.
- **UINT16** [NumberOfBlocks](#)
Offset 30-31 Number of config blocks (N)
- **UINT32** [AvailableSize](#)
Offset 32-35 Current config block table size.

13.3.1 Detailed Description

Config Block Table Header.

Definition at line 50 of file ConfigBlock.h.

The documentation for this struct was generated from the following file:

- [ConfigBlock.h](#)

13.4 _EFI_MM_RESERVED_MMRAM_REGION Struct Reference

Structure describing a MMRAM region which cannot be used for the MMRAM heap.

```
#include <PiMultiPhase.h>
```

Public Attributes

- [EFI_PHYSICAL_ADDRESS](#) `MmramReservedStart`
Starting address of the reserved MMRAM area, as it appears while MMRAM is open.
- `UINT64` [MmramReservedSize](#)
Number of bytes occupied by the reserved MMRAM area.

13.4.1 Detailed Description

Structure describing a MMRAM region which cannot be used for the MMRAM heap.

Definition at line 139 of file PiMultiPhase.h.

13.4.2 Member Data Documentation

13.4.2.1 MmramReservedSize

```
UINT64 _EFI_MM_RESERVED_MMRAM_REGION::MmramReservedSize
```

Number of bytes occupied by the reserved MMRAM area.

A size of zero indicates the last MMRAM area.

Definition at line 149 of file PiMultiPhase.h.

13.4.2.2 MmramReservedStart

`EFI_PHYSICAL_ADDRESS _EFI_MM_RESERVED_MMRAM_REGION::MmramReservedStart`

Starting address of the reserved MMRAM area, as it appears while MMRAM is open.

Ignored if MmramReservedSize is 0.

Definition at line 144 of file PiMultiPhase.h.

The documentation for this struct was generated from the following file:

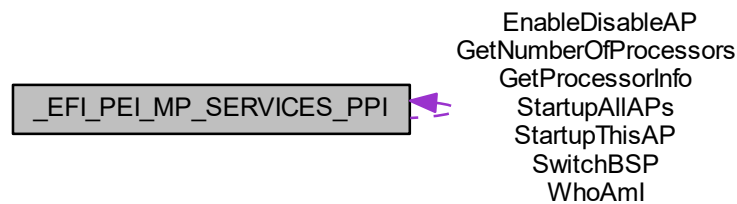
- [PiMultiPhase.h](#)

13.5 _EFI_PEI_MP_SERVICES_PPI Struct Reference

This PPI is installed by some platform or chipset-specific PEIM that abstracts handling multiprocessor support.

```
#include <MpServices.h>
```

Collaboration diagram for _EFI_PEI_MP_SERVICES_PPI:



13.5.1 Detailed Description

This PPI is installed by some platform or chipset-specific PEIM that abstracts handling multiprocessor support.

Definition at line 265 of file MpServices.h.

The documentation for this struct was generated from the following file:

- [MpServices.h](#)

13.6 _ITBT_GENERIC_CONFIG Struct Reference

ITBT Controller Data Structure to be used cross to RP and controller to be shared by CONFIG_BLOCK and HOB.

```
#include <PeiITbtGenericStructure.h>
```

Public Attributes

- UINT16 [ITbtForcePowerOnTimeoutInMs](#)
Timeout value for forcing power iTBT controller on every boot/reboot/Sx exit as a precondition for execution of following mailbox communication.
- UINT16 [ITbtConnectTopologyTimeoutInMs](#)
*Timeout value while sending connect topology mailbox command in order to bring all connected TBT devices are available on PCIe before BIOS will enumerate them in BDS **(Test) default is 5000 ms***
- UINT8 [ITbtSecurityLevel](#)
*iTbt Security Level **Deprecated***
- UINT8 [ITbtPcieTunnelingForUsb4](#)
*Disable/Enable PCIe tunneling for USB4. **default is enable***
- UINT8 [Usb4CmMode](#)
USB4 CM mode.
- UINT8 [Reserved](#) [1]
Reserved for DWORD alignment.

13.6.1 Detailed Description

ITBT Controller Data Structure to be used cross to RP and controller to be shared by CONFIG_BLOCK and HOB.

Definition at line 52 of file `PeiITbtGenericStructure.h`.

13.6.2 Member Data Documentation

13.6.2.1 ITbtForcePowerOnTimeoutInMs

```
UINT16 _ITBT_GENERIC_CONFIG::ITbtForcePowerOnTimeoutInMs
```

Timeout value for forcing power iTBT controller on every boot/reboot/Sx exit as a precondition for execution of following mailbox communication.

After applying Force Power Thunderbolt BIOS shall poll for iTBT readiness for mailbox communication. If TBT cable is disconnected, iTBT microcontrollers are in lower power state. To ensure successful mailbox execution, independently on presence of TBT cable, TBT BIOS shall bring iTBT microcontrollers up by applying Force Power. iTBT microcontrollers will wake up either due to TBT cable presence or Force Power event. **(Test) default is 500 ms**

Definition at line 64 of file `PeiITbtGenericStructure.h`.

The documentation for this struct was generated from the following file:

- [PeiITbtGenericStructure.h](#)

13.7 _ITBT_ROOTPORT_CONFIG Struct Reference

iTBT RootPort Data Structure

```
#include <PeiITbtGenericStructure.h>
```


Public Attributes

- [UINT8 ITbtPcieRootPortEn](#)
Disable/Enable iTBT PCIe Root Port.
- [UINT8 Reserved](#) [3]
Reserved for DWORD alignment.

13.7.1 Detailed Description

iTBT RootPort Data Structure

Definition at line 44 of file [PeiITbtGenericStructure.h](#).

The documentation for this struct was generated from the following file:

- [PeiITbtGenericStructure.h](#)

13.8 _LIST_ENTRY Struct Reference

[_LIST_ENTRY](#) structure definition.

```
#include <Base.h>
```

Collaboration diagram for [_LIST_ENTRY](#):



13.8.1 Detailed Description

[_LIST_ENTRY](#) structure definition.

Definition at line 255 of file [Base.h](#).

The documentation for this struct was generated from the following file:

- [Base.h](#)

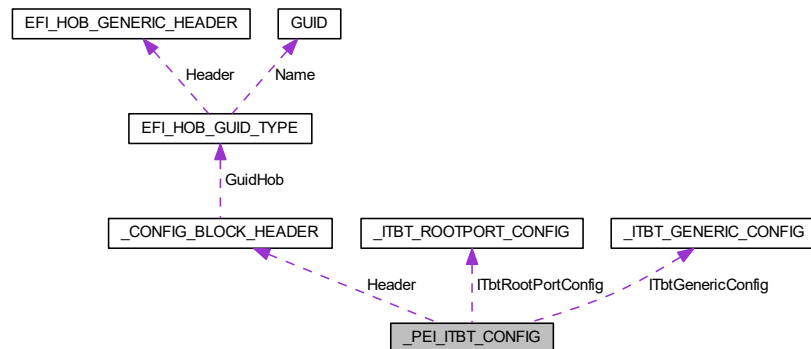
13.9 _PEI_ITBT_CONFIG Struct Reference

ITBT PEI configuration

Revision 1:

```
#include <PeiITbtConfig.h>
```

Collaboration diagram for _PEI_ITBT_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.
- [ITBT_GENERIC_CONFIG](#) ITbtGenericConfig
ITbt Common Configuration.
- [ITBT_ROOTPORT_CONFIG](#) ITbtRootPortConfig [MAX_ITBT_PCIE_PORT]
iTbt Root Port Configuration
- [UINT16 ITbtDmaLtr](#) [MAX_HOST_ITBT_DMA_NUMBER]
iTbt Host controller DMA LTR value

13.9.1 Detailed Description

ITBT PEI configuration

Revision 1:

- Initial version. **Revision 2:**
- Add ITbtPcieTunnelingForUsb4, deprecated ITbtSecurityLevel.

Definition at line 53 of file PeiITbtConfig.h.

The documentation for this struct was generated from the following file:

- [PeiITbtConfig.h](#)

13.10 _PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI Struct Reference

This PPI provides function to install default silicon policy.

```
#include <PeiPreMemSiDefaultPolicy.h>
```

Public Attributes

- [PEI_PREMEM_POLICY_INIT](#) [PeiPreMemPolicyInit](#)
PeiPreMemPolicyInit()

13.10.1 Detailed Description

This PPI provides function to install default silicon policy.

Definition at line 55 of file [PeiPreMemSiDefaultPolicy.h](#).

The documentation for this struct was generated from the following file:

- [PeiPreMemSiDefaultPolicy.h](#)

13.11 _PEI_SI_DEFAULT_POLICY_INIT_PPI Struct Reference

This PPI provides function to install default silicon policy.

```
#include <PeiSiDefaultPolicy.h>
```

Public Attributes

- [PEI_POLICY_INIT](#) [PeiPolicyInit](#)
PeiPolicyInit()

13.11.1 Detailed Description

This PPI provides function to install default silicon policy.

Definition at line 55 of file [PeiSiDefaultPolicy.h](#).

The documentation for this struct was generated from the following file:

- [PeiSiDefaultPolicy.h](#)

13.12 _PPM_CUSTOM_CTDp_TABLE Struct Reference

PPM Custom ConfigTdp Settings.

```
#include <CpuPowerMgmtCustomConfig.h>
```

Public Attributes

- UINT32 [CustomPowerLimit1Time](#): 8
Short term Power Limit time window value for custom cTDP level.
- UINT32 [CustomTurboActivationRatio](#): 8
Turbo Activation Ratio for custom cTDP level.
- UINT32 [RsvdBits](#): 16
Bits reserved for DWORD alignment.
- UINT16 [CustomPowerLimit1](#)
Short term Power Limit value for custom cTDP level. Units are based on POWER_MGMT_CONFIG.CustomPower↔ Unit.
- UINT16 [CustomPowerLimit2](#)
Long term Power Limit value for custom cTDP level. Units are based on POWER_MGMT_CONFIG.CustomPower↔ Unit.

13.12.1 Detailed Description

PPM Custom ConfigTdp Settings.

Definition at line 79 of file CpuPowerMgmtCustomConfig.h.

The documentation for this struct was generated from the following file:

- [CpuPowerMgmtCustomConfig.h](#)

13.13 _SI_POLICY_STRUCT Struct Reference

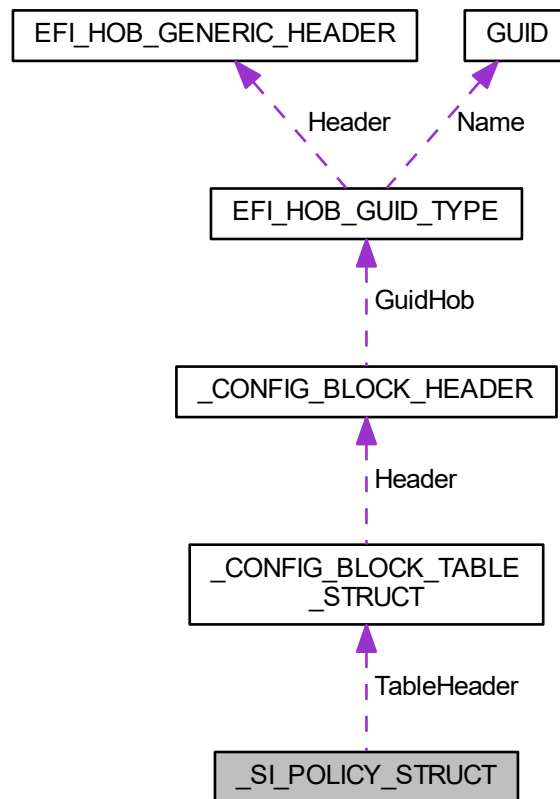
SI Policy PPI

All SI config block change history will be listed here

.

```
#include <SiPolicyStruct.h>
```

Collaboration diagram for _SI_POLICY_STRUCT:



Public Attributes

- [CONFIG_BLOCK_TABLE_HEADER TableHeader](#)
Config Block Table Header.

13.13.1 Detailed Description

SI Policy PPI

All SI config block change history will be listed here

.

- **Revision 1:**
 - Initial version.

Definition at line 85 of file SiPolicyStruct.h.

The documentation for this struct was generated from the following file:

- [SiPolicyStruct.h](#)

13.14 _SI_PREMEM_POLICY_STRUCT Struct Reference

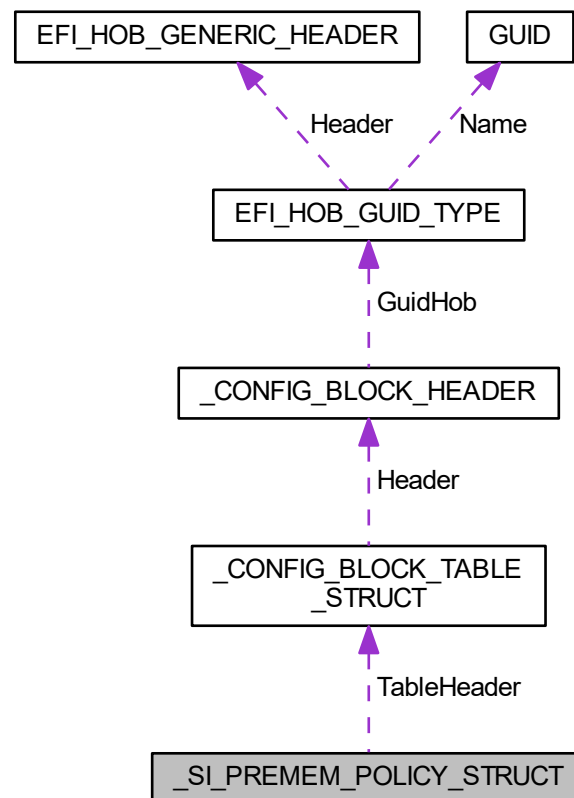
SI Policy PPI in Pre-Mem

All SI config block change history will be listed here

.

```
#include <SiPolicyStruct.h>
```

Collaboration diagram for _SI_PREMEM_POLICY_STRUCT:



Public Attributes

- [CONFIG_BLOCK_TABLE_HEADER TableHeader](#)
Config Block Table Header.

13.14.1 Detailed Description

SI Policy PPI in Pre-Mem

All SI config block change history will be listed here

.

- **Revision 1:**
 - Initial version.

Definition at line 71 of file SiPolicyStruct.h.

The documentation for this struct was generated from the following file:

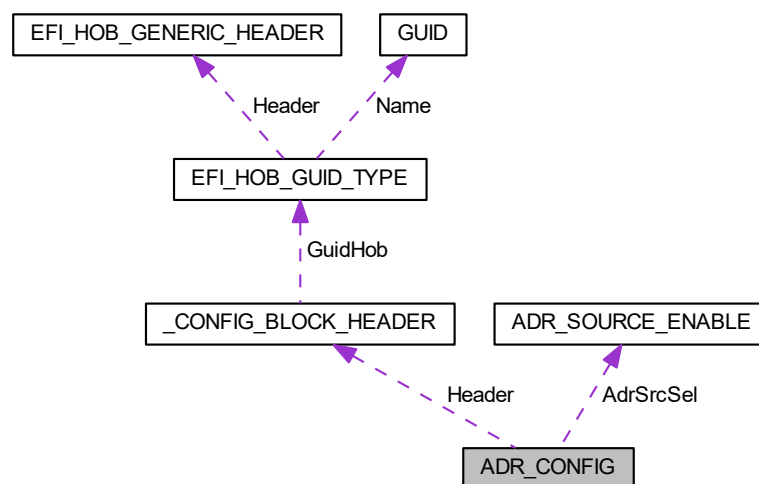
- [SiPolicyStruct.h](#)

13.15 ADR_CONFIG Struct Reference

ADR Configuration **Revision 1**: - Initial version.

```
#include <AdrConfig.h>
```

Collaboration diagram for ADR_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [AdrEn](#): 2
Determine if Adr is enabled - 0: PLATFORM_POR, 1: FORCE_ENABLE, 2: FORCE_DISABLE.
- UINT32 [AdrTimerEn](#): 2
Determine if Adr timer options are enabled - 0: PLATFORM_POR, 1: FORCE_ENABLE, 2: FORCE_DISABLE.
- UINT32 [AdrTimer1Val](#): 8
Determines the Timeout value used for the ADR timer 1. A value of zero bypasses the timer.
- UINT32 [AdrMultiplier1Val](#): 3

Specifies the tick frequency upon which the timer 1 will increment. ADR_TIMER_SCALE should be used to encode values.

- UINT32 [AdrTimer2Val](#): 8

Determines the Timeout value used for the ADR timer 2. A value of zero bypasses the timer.

- UINT32 [AdrMultiplier2Val](#): 3

Specifies the tick frequency upon which the timer 2 will increment. ADR_TIMER_SCALE should be used to encode values.

- UINT32 [AdrHostPartitionReset](#): 2

Determine if Host Partition Reset is enabled - 0: PLATFORM_POR, 1: FORCE_ENABLE, 2: FORCE_DISABLE.

- UINT32 [AdrSrcOverride](#): 1

Check if default ADR sources will be overriten with custom 0: Not overwritten, 1: Overwritten.

- UINT32 [AdrPlatAckEn](#): 2

Determine if Platform Acknowledge is enabled - 0: PLATFORM_POR, 1: FORCE_ENABLE, 2: FORCE_DISABLE.

- [ADR_SOURCE_ENABLE](#) [AdrSrcSel](#)

Determine which ADR sources are enabled - 0: Enabled, 1: Disabled.

13.15.1 Detailed Description

ADR Configuration **Revision 1**: - Initial version.

Definition at line 97 of file [AdrConfig.h](#).

The documentation for this struct was generated from the following file:

- [AdrConfig.h](#)

13.16 ADR_SOURCE_ENABLE Union Reference

ADR Source Enable.

```
#include <AdrConfig.h>
```

13.16.1 Detailed Description

ADR Source Enable.

Definition at line 59 of file [AdrConfig.h](#).

The documentation for this union was generated from the following file:

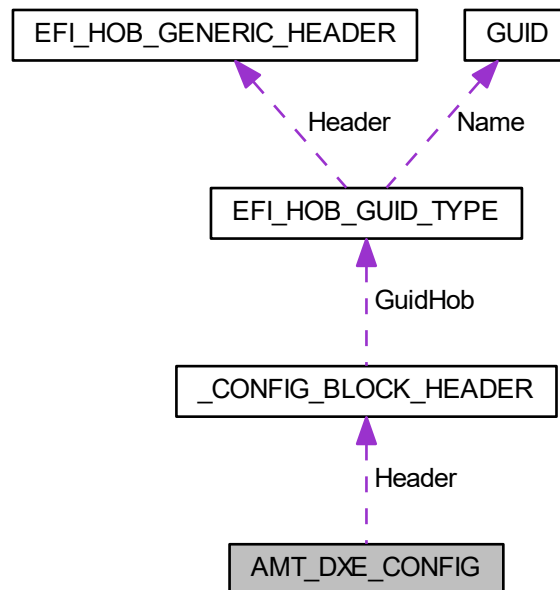
- [AdrConfig.h](#)

13.17 AMT_DXE_CONFIG Struct Reference

AMT Dxe Configuration Structure.

```
#include <AmtConfig.h>
```

Collaboration diagram for AMT_DXE_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- UINT32 [CiraRequest](#): 1
Trigger CIRA boot. 0: No CIRA request; 1: Trigger CIRA request.
- UINT32 [UnConfigureMe](#): 1
OEMFlag Bit 15: Unconfigure ME with resetting MEBx password to default. 0: No; 1: Un-configure ME without password.
- UINT32 [MebxDebugMsg](#): 1
- UINT32 [HideUnConfigureMeConfirm](#): 1
- UINT32 [UsbProvision](#): 1
Enable/Disable of AMT USB Provisioning. 0: Disable; 1: Enable.
- UINT32 [AmtbxHotkeyPressed](#): 1
- UINT32 [AmtbxSelectionScreen](#): 1
OEMFlag Bit 2: Enable MEBx selection screen with 2 options: Press 1 to enter ME Configuration Screens Press 2 to initiate a remote connection.
- UINT32 [RsvdBits](#): 25
Reserved for future use & Config block alignment.

- UINT32 [CiraTimeout](#): 8
- UINT32 [CpuReplacementTimeout](#): 8
- CPU Replacement Timeout **0: 10 seconds** 1: 30 seconds 2~5: Reserved 6: No delay 7: Unlimited delay.
- UINT32 [MebxNonUiTextMode](#): 4
- UINT32 [MebxUiTextMode](#): 4
- UINT32 [MebxGraphicsMode](#): 4
- UINT32 [OemResolutionSettingsRsvd](#): 4
- [AMT_REPORT_ERROR](#) [AmtReportError](#)

Function pointer for displaying error message on screen.

13.17.1 Detailed Description

AMT Dxe Configuration Structure.

Revision 1:

- Initial version.

Definition at line 134 of file AmtConfig.h.

13.17.2 Member Data Documentation

13.17.2.1 AmtbxHotkeyPressed

UINT32 [AMT_DXE_CONFIG::AmtbxHotkeyPressed](#)

Deprecated : OEMFlag Bit 1: Enable automatic MEBx hotkey press. **0: Disable**; 1: Enable

Definition at line 141 of file AmtConfig.h.

13.17.2.2 AmtbxSelectionScreen

UINT32 [AMT_DXE_CONFIG::AmtbxSelectionScreen](#)

OEMFlag Bit 2: Enable MEBx selection screen with 2 options: Press 1 to enter ME Configuration Screens Press 2 to initiate a remote connection.

- **0: Disabled**
- 1: Enabled

Deprecated

Definition at line 149 of file AmtConfig.h.

13.17.2.3 CiraTimeout

UINT32 AMT_DXE_CONFIG::CiraTimeout

Deprecated

Definition at line 151 of file AmtConfig.h.

13.17.2.4 CpuReplacementTimeout

UINT32 AMT_DXE_CONFIG::CpuReplacementTimeout

CPU Replacement Timeout **0: 10 seconds** 1: 30 seconds 2~5: Reserved 6: No delay 7: Unlimited delay.

Deprecated

Definition at line 160 of file AmtConfig.h.

13.17.2.5 HideUnConfigureMeConfirm

UINT32 AMT_DXE_CONFIG::HideUnConfigureMeConfirm

Deprecated : OEMFlag Bit 6: Hide Unconfigure ME confirmation prompt when attempting ME unconfiguration. **0: Don't hide**; 1: Hide

Definition at line 139 of file AmtConfig.h.

13.17.2.6 MebxDebugMsg

UINT32 AMT_DXE_CONFIG::MebxDebugMsg

Deprecated : OEMFlag Bit 14: Enable OEM debug menu in MEBx. **0: Disable**; 1: Enable

Definition at line 138 of file AmtConfig.h.

13.17.2.7 MebxGraphicsMode

UINT32 AMT_DXE_CONFIG::MebxGraphicsMode

Deprecated : Resolution for graphics mode. **0: Auto**; 1: 640x480; 2: 800x600; 3: 1024x768

Definition at line 163 of file AmtConfig.h.

13.17.2.8 MebxNonUiTextMode

UINT32 AMT_DXE_CONFIG::MebxNonUiTextMode

Deprecated : Resolution for non-UI text mode. **0: Auto**; 1: 80x25; 2: 100x31

Definition at line 161 of file AmtConfig.h.

13.17.2.9 MebxUiTextMode

UINT32 AMT_DXE_CONFIG::MebxUiTextMode

Deprecated : Resolution for UI text mode. **0: Auto**; 1: 80x25; 2: 100x31

Definition at line 162 of file AmtConfig.h.

13.17.2.10 OemResolutionSettingsRsvd

UINT32 AMT_DXE_CONFIG::OemResolutionSettingsRsvd

Deprecated : Reserved for future use & Config block alignment

Definition at line 164 of file AmtConfig.h.

The documentation for this struct was generated from the following file:

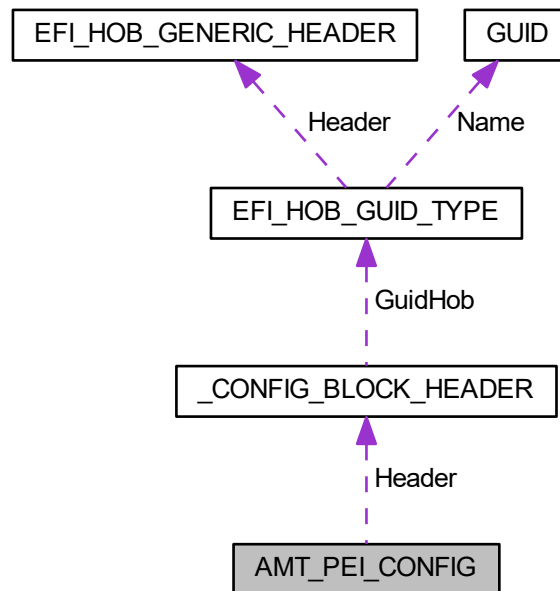
- [AmtConfig.h](#)

13.18 AMT_PEI_CONFIG Struct Reference

AMT Pei Configuration Structure.

```
#include <AmtConfig.h>
```

Collaboration diagram for AMT_PEI_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [AmtEnabled](#): 1
Enable or Disable Intel Active Management Technology feature.
- UINT32 [WatchDogEnabled](#): 1
ME WatchDog timer feature.
- UINT32 [FwProgress](#): 1
*PET Events Progress to receive PET Events. 0: Disable; 1: **Enable***
- UINT32 [ManageabilityMode](#): 1
- UINT32 [AmtSolEnabled](#): 1
*Serial Over Lan retrieved from Mebx. The default value depends on CSME/AMT. 0: Disable, 1: **Enable***
- UINT32 [RemoteAssistance](#): 1
- UINT32 [AmtKvmEnabled](#): 1
- UINT32 [ForcMebxSyncUp](#): 1
*0: **No**; 1: Force MEBX execution*
- UINT32 [RsvdBits](#): 24
Reserved for future use & Config block alignment.
- UINT16 [WatchDogTimerOs](#)
*OS WatchDog Timer 0: **Disable** OS WDT won't be started after stopping BIOS WDT even if WatchDogEnabled is 1.*
- UINT16 [WatchDogTimerBios](#)
*BIOS WatchDog Timer 0: **Disable** BIOS WDT won't be started even if WatchDogEnabled is 1.*

13.18.1 Detailed Description

AMT Pei Configuration Structure.

Revision 1:

- Initial version.

Definition at line 89 of file AmtConfig.h.

13.18.2 Member Data Documentation

13.18.2.1 AmtEnabled

```
UINT32 AMT_PEI_CONFIG::AmtEnabled
```

Enable or Disable Intel Active Management Technology feature.

If disabled, all Intel AMT features, including Alert Standard Format features, will not be supported. 0: Disable **1: Enable**.

Definition at line 97 of file AmtConfig.h.

13.18.2.2 AmtKvmEnabled

```
UINT32 AMT_PEI_CONFIG::AmtKvmEnabled
```

Deprecated : KVM retrieved from Mebx. The default value depends on CSME/AMT. 0: Disable, **1: Enable**

Definition at line 109 of file AmtConfig.h.

13.18.2.3 ManageabilityMode

```
UINT32 AMT_PEI_CONFIG::ManageabilityMode
```

Deprecated : Manageability Mode sync with Mebx, 0: Disabled; **1: AMT**

Definition at line 106 of file AmtConfig.h.

13.18.2.4 RemoteAssistance

UINT32 AMT_PEI_CONFIG::RemoteAssistance

Deprecated : Remote Assistance is enabled if platform is provisioned. **0: Disable**, 1: Enable

Definition at line 108 of file AmtConfig.h.

13.18.2.5 WatchDogEnabled

UINT32 AMT_PEI_CONFIG::WatchDogEnabled

ME WatchDog timer feature.

If disabled, below WatchDogTimerOs/WatchDogTimerBios will be irrelevant. See WatchDogTimerOs and WatchDogTimerBios description. **0: Disable** 1: Enable ME WDT if corresponding timer value is not zero.

Definition at line 104 of file AmtConfig.h.

13.18.2.6 WatchDogTimerBios

UINT16 AMT_PEI_CONFIG::WatchDogTimerBios

BIOS WatchDog Timer **0: Disable** BIOS WDT won't be started even if WatchDogEnabled is 1.

Non zero value - The BIOS WDT is set according to the value and started if WatchDogEnabled is 1.

Definition at line 124 of file AmtConfig.h.

13.18.2.7 WatchDogTimerOs

UINT16 AMT_PEI_CONFIG::WatchDogTimerOs

OS WatchDog Timer **0: Disable** OS WDT won't be started after stopping BIOS WDT even if WatchDogEnabled is 1.

Non zero value - OS WDT will be started after stopping BIOS WDT if WatchDogEnabled is 1. The timer is set according to the value.

Definition at line 118 of file AmtConfig.h.

The documentation for this struct was generated from the following file:

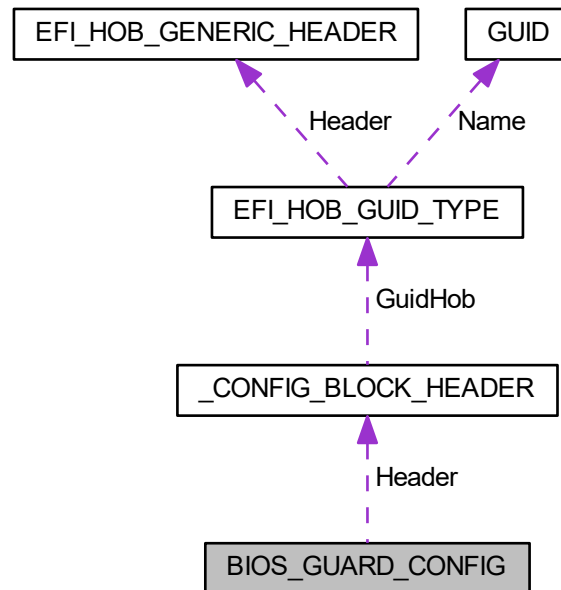
- [AmtConfig.h](#)

13.19 BIOS_GUARD_CONFIG Struct Reference

BIOS Guard Configuration Structure.

```
#include <BiosGuardConfig.h>
```

Collaboration diagram for BIOS_GUARD_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- BIOSGUARD_ATTRIBUTES [BiosGuardAttr](#)
BIT1 - EC Present, BIT2 - EC BIOS Guard protection, BIT3 - Descriptor Override policy, BIT4 - Flash wearout protection, BIT5 - FTU enable.
- UINT64 [BgpdtHash](#) [4]
Hash of the BGPDT that will be programmed to PLAT_FRMW_PROT_HASH_0/1/2/3 MSR.
- EFI_PHYSICAL_ADDRESS [BiosGuardModulePtr](#)
Pointer to the BIOS Guard Module.
- EFI_PHYSICAL_ADDRESS [SendEcCmd](#)
Platform code can provide interface to communicate with EC through this function.
- UINT8 [EcCmdProvisionEav](#)
EC Command Provision Eav.
- UINT8 [EcCmdLock](#)
EC Command Lock.
- UINT8 [Reserved](#) [6]
Reserved for future use and config block alignment.

13.19.1 Detailed Description

BIOS Guard Configuration Structure.

Platform policies for BIOS Guard Configuration for all processor security features configuration. Platform code can pass relevant configuration data through this structure.

Note

Optional. These policies will be ignored if [CPU_SECURITY_PREMEM_CONFIG](#) -> BiosGuard is disabled, or PeiBiosGuardLibNull is used.

Revision 1:

- Initial version.

Definition at line 55 of file BiosGuardConfig.h.

The documentation for this struct was generated from the following file:

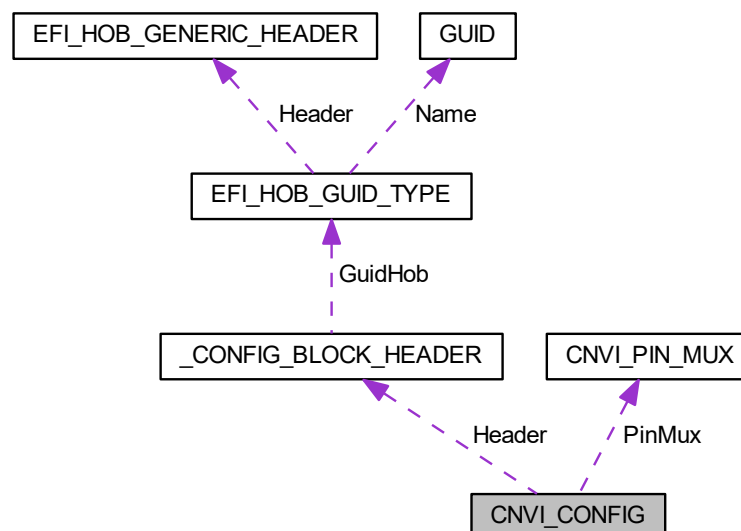
- [BiosGuardConfig.h](#)

13.20 CNVI_CONFIG Struct Reference

The [CNVI_CONFIG](#) block describes the expected configuration of the CNVi IP.

```
#include <CnviConfig.h>
```

Collaboration diagram for CNVI_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [Mode](#): 1
This option allows for automatic detection of Connectivity Solution.
- UINT32 [BtCore](#): 1
*The option to turn ON or OFF the BT Core. 0: Disabled, 1: **Enabled***
- UINT32 [BtAudioOffload](#): 1
The option to enable or disable BT Audio Offload.
- UINT32 [WifiCore](#): 1
*The option to turn ON or OFF the Wi-Fi Core. 0: Disabled, 1: **Enabled***
- [CNVI_PIN_MUX](#) PinMux
CNVi PinMux Configuration RESET#/CLKREQ to CRF, can have two alternative mappings, depending on board routing requirements.

13.20.1 Detailed Description

The [CNVI_CONFIG](#) block describes the expected configuration of the CNVi IP.

Revision 1: - Initial version. **Revision 2:** - Add DdrRfim. **Revision 3:** - Add WifiCore. **Revision 4:** - Move DdrRfim to Pre-Mem config.

Definition at line 73 of file CnviConfig.h.

13.20.2 Member Data Documentation

13.20.2.1 BtAudioOffload

```
UINT32 CNVI_CONFIG::BtAudioOffload
```

The option to enable or disable BT Audio Offload.

0: Disabled, 1: Enabled

Note

This feature only support with Intel(R) Wireless-AX 22560

Definition at line 89 of file CnviConfig.h.

13.20.2.2 Mode

```
UINT32 CNVI_CONFIG::Mode
```

This option allows for automatic detection of Connectivity Solution.

Auto Detection assumes that CNVi will be enabled when available; Disable allows for disabling CNVi. CnviMode↔ Disabled = Disabled, **CnviModeAuto = Auto Detection**

Definition at line 82 of file CnviConfig.h.

The documentation for this struct was generated from the following file:

- [CnviConfig.h](#)

13.21 CNVI_PIN_MUX Struct Reference

CNVi signals pin muxing settings.

```
#include <CnviConfig.h>
```

Public Attributes

- UINT32 [RfReset](#)
*RF_RESET# Pin mux configuration. Refer to GPIO_*_MUXING_CNVI_RF_RESET_.*.*
- UINT32 [Clkreq](#)
*CLKREQ Pin mux configuration. Refer to GPIO_*_MUXING_CNVI_*_CLKREQ_.*.*

13.21.1 Detailed Description

CNVi signals pin muxing settings.

If signal can be enable only on a single pin then this parameter is ignored by RC. Refer to GPIO_*_MUXING_C↔ NVI_* in GpioPins*.h for supported settings on a given platform

Definition at line 60 of file CnviConfig.h.

The documentation for this struct was generated from the following file:

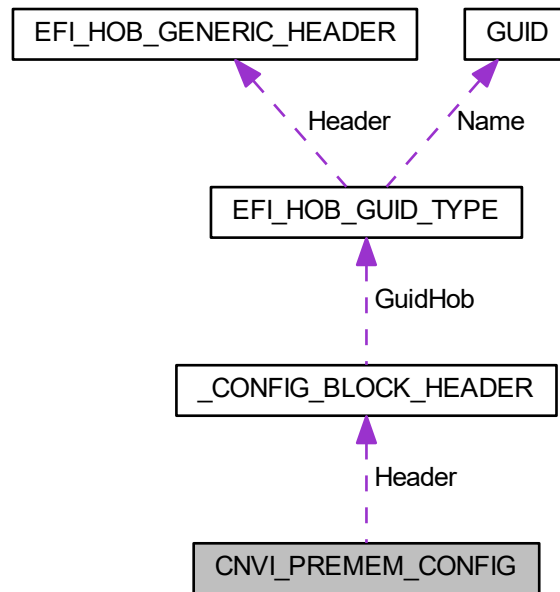
- [CnviConfig.h](#)

13.22 CNVI_PREMEM_CONFIG Struct Reference

The [CNVI_PREMEM_CONFIG](#) block describes the expected configuration of the CNVi IP.

```
#include <CnviConfig.h>
```

Collaboration diagram for CNVI_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [UINT8](#) [DdrRfim](#)
*The option to enable or disable DDR RFI Mitigation. 0: Disabled, 1: **Enabled***

13.22.1 Detailed Description

The [CNVI_PREMEM_CONFIG](#) block describes the expected configuration of the CNVi IP.

Revision 1: - Initial version.

Definition at line 105 of file [CnviConfig.h](#).

The documentation for this struct was generated from the following file:

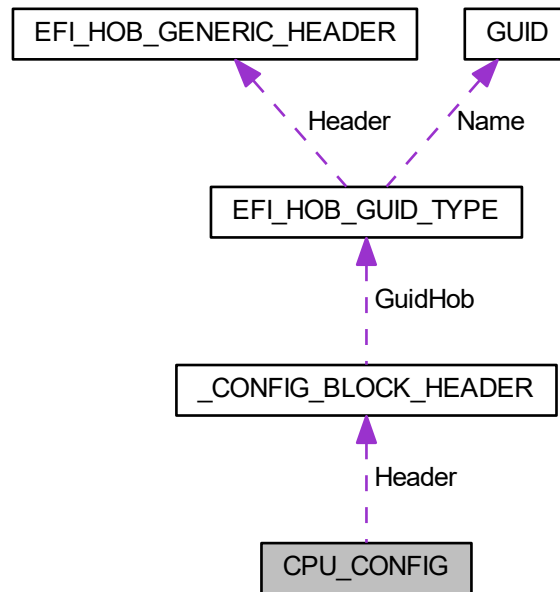
- [CnviConfig.h](#)

13.23 CPU_CONFIG Struct Reference

CPU Configuration Structure.

```
#include <CpuConfig.h>
```

Collaboration diagram for CPU_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- [EFI_PHYSICAL_ADDRESS MicrocodePatchAddress](#)
Pointer to microcode patch that is suitable for this processor.
- UINT32 [AesEnable](#): 1
Enable or Disable Advanced Encryption Standard (AES) feature.
- UINT32 [TxtEnable](#): 1
Enable or Disable Trusted Execution Technology (TXT) feature.
- UINT32 [SkipMplInit](#): 1
For Fsp only, Silicon Initialization will skip MP Initialization (including BSP) if enabled. For non-FSP, this should always be 0.
- UINT32 [PpinSupport](#): 2
Enable or Disable or Auto for PPIN Support to view Protected Processor Inventory Number.
- UINT32 [AcSplitLock](#): 1
Enable or Disable #AC machine check on split lock.
- UINT32 [AvxDisable](#): 1
Enable or Disable Avx.

- UINT32 [Avx3Disable](#): 1
- UINT32 [X2ApicSupport](#): 1
Enable or Disable X2APIC Support.
- UINT32 [RsvdBits](#): 23
Reserved for future use.
- UINT16 [SmbiosType4MaxSpeedOverride](#)
Provide the option for platform to override the MaxSpeed field of Smbios Type 4.
- UINT8 [Reserved0](#) [2]
Reserved for future use.

13.23.1 Detailed Description

CPU Configuration Structure.

Revision 1:

- Initial version. **Revision 2:**
- Add SmbiosType4MaxSpeedOverride. **Revision 3:**
- Add AvxDisable & Avx3Disable. **Revision 4:**
- Add X2ApicSupport. **Revision 5:**
- Deprecate Avx3Disable

Definition at line 58 of file CpuConfig.h.

13.23.2 Member Data Documentation

13.23.2.1 AcSplitLock

```
UINT32 CPU_CONFIG::AcSplitLock
```

Enable or Disable #AC machine check on split lock.

- **0: Disable**
- 1: Enable

Definition at line 88 of file CpuConfig.h.

13.23.2.2 AesEnable

UINT32 CPU_CONFIG::AesEnable

Enable or Disable Advanced Encryption Standard (AES) feature.

For some countries, this should be disabled for legal reasons.

- 0: Disable
- **1: Enable**

Definition at line 68 of file CpuConfig.h.

13.23.2.3 Avx3Disable

UINT32 CPU_CONFIG::Avx3Disable

Deprecated Enable or Disable Avx3.

- **1: Disable**
- 0: Enable

Definition at line 101 of file CpuConfig.h.

13.23.2.4 AvxDisable

UINT32 CPU_CONFIG::AvxDisable

Enable or Disable Avx.

- 1: Disable
- **0: Enable**

Definition at line 94 of file CpuConfig.h.

13.23.2.5 PpinSupport

UINT32 CPU_CONFIG::PpinSupport

Enable or Disable or Auto for PPIN Support to view Protected Processor Inventory Number.

- **0: Disable**
- 1: Enable
- 2: Auto : Feature is based on End Of Manufacturing (EOM) flag. If EOM is set, it is disabled.

Definition at line 82 of file CpuConfig.h.

13.23.2.6 SmbiosType4MaxSpeedOverride

UINT16 CPU_CONFIG::SmbiosType4MaxSpeedOverride

Provide the option for platform to override the MaxSpeed field of Smbios Type 4.

Value 4000 means 4000MHz. If this value is not zero, it dominates the field. If this value is zero, CPU RC will update the field according to the max radio. **default is 0.**

Definition at line 116 of file CpuConfig.h.

13.23.2.7 TxtEnable

UINT32 CPU_CONFIG::TxtEnable

Enable or Disable Trusted Execution Technology (TXT) feature.

- 0: Disable
- **1: Enable**

Definition at line 74 of file CpuConfig.h.

13.23.2.8 X2ApicSupport

UINT32 CPU_CONFIG::X2ApicSupport

Enable or Disable X2APIC Support.

- **0: Disable**
- 1: Enable

Definition at line 107 of file CpuConfig.h.

The documentation for this struct was generated from the following file:

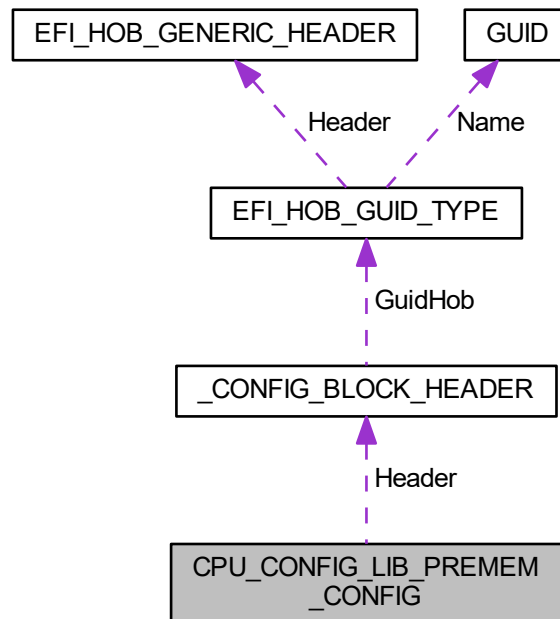
- [CpuConfig.h](#)

13.24 CPU_CONFIG_LIB_PREMEM_CONFIG Struct Reference

CPU Config Library PreMemory Configuration Structure.

```
#include <CpuConfigLibPreMemConfig.h>
```

Collaboration diagram for CPU_CONFIG_LIB_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- UINT32 [HyperThreading](#): 1
*Enable or Disable Hyper Threading; 0: Disable; 1: **Enable**.*
- UINT32 [BootFrequency](#): 2
Sets the boot frequency starting from reset vector.
- UINT32 [ActiveCoreCount](#): 3
Number of processor cores to enable.
- UINT32 [JtagC10PowerGateDisable](#): 1
*False: JTAG is power gated in C10 state. True: keeps the JTAG power up during C10 and deeper power states for debug purpose. 0: **False** < >; 1: **True**.*
- UINT32 [BistOnReset](#): 1
*(Test) Enable or Disable BIST on Reset; 0: **Disable**; 1: **Enable**.*
- UINT32 [VmxEnable](#): 1
Enable or Disable Virtual Machine Extensions (VMX) feature.
- UINT32 [FCIkFrequency](#): 2

- Processor Early Power On Configuration FCLK setting.*

 - UINT32 [CrashLogEnable](#): 1
Enable or Disable CrashLog feature.
 - UINT32 [TmeEnable](#): 1
Enable or Disable Total Memory Encryption (TME) feature.
 - UINT32 [DebugInterfaceEnable](#): 2
*Enable or Disable processor debug features; 0: Disable; 1: Enable; 2: **No Change**.*
 - UINT32 [DebugInterfaceLockEnable](#): 1
*Lock or Unlock debug interface features; 0: Disable; 1: **Enable**.*
 - UINT32 [ActiveCoreCount1](#): 4
Number of big cores in processor to enable.
 - UINT32 [PeciSxReset](#): 1
Enables a mailbox command to resolve rare Peci related Sx issues.
 - UINT32 [PeciC10Reset](#): 1
Enables the mailbox command to resolve Peci reset issues during Pkg-C10 exit.
 - UINT32 [ActiveSmallCoreCount](#): 6
Number of small cores in processor to enable.
 - UINT32 [CrashLogGprs](#): 2
Enable or Disable CrashLog GPRs dump.
 - UINT32 [ComputeDieSscEnable](#): 1
Enable or Disable Compute Die SSC configuration.
 - UINT32 [SocDieSscEnable](#): 1
Enable or Disable Soc Die SSC configuration.
 - UINT8 [CpuRatio](#)
CpuRatio - Max non-turbo ratio (Flexible Ratio Boot) is set to CpuRatio.
 - UINT8 [ActiveSocNorthAtomCoreCount](#)
Number of enabled atom cores on SoC.
 - UINT8 [ComputeDieSscValue](#)
SSC configuration value of Compute Die.
 - UINT8 [SocDieSscValue](#)
SSC configuration value of Soc Die.
 - UINT32 [BclkSource](#): 2
Clock source of BCLK OC Frequency.
 - UINT8 [InfDfxPwrEnable](#)
Determine whether to keep or shutdown INF_DFX power rail.
 - UINT8 [ConfigTdpLevel](#)
*Configuration for boot TDP selection; 0: **TDP Nominal**; 1: TDP Down; 2: TDP Up.*

13.24.1 Detailed Description

CPU Config Library PreMemory Configuration Structure.

Revision 1:

- Initial version. **Revision 2:**
- Expand the supported number of processor cores (ActiveCoreCount1). **Revision 3:**
- Added Peci Sx and C10 Reset. **Revision 4:**
- Added ActiveSmallCoreCount. **Revision 5:**

- Added CrashLogGprs **Revision 6**:
- Added ActiveSocNorthAtomCoreCount **Revision 7**:
- Added ComputeDieSscEnable, SocDieSscEnable, ComputeDieSscValue, and SocDieSscValue **Revision 8**:
- Added BclkSource. **Revision 9**:
- Removed Elixir Springs Patch **Revision 10**
- Add InfDfxPwrEnable **Revision 11**
- Add ConfigTdpLevel, CustomPowerLimit1

Definition at line 61 of file CpuConfigLibPreMemConfig.h.

13.24.2 Member Data Documentation

13.24.2.1 ActiveCoreCount

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::ActiveCoreCount

Number of processor cores to enable.

- **0: All cores**
- 1: 1 core
- 2: 2 cores
- 3: 3 cores

Deprecated due to core active number limitaion.

Definition at line 79 of file CpuConfigLibPreMemConfig.h.

13.24.2.2 ActiveCoreCount1

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::ActiveCoreCount1

Number of big cores in processor to enable.

And support up to 15 cores.

- 0: Disable all cores for Hybrid CPU; Active all cores for Non-Hybrid CPU (MAX 16).
- 1: 1 core
- 2: 2 cores
- 3: 3 cores

Definition at line 121 of file CpuConfigLibPreMemConfig.h.

13.24.2.3 ActiveSmallCoreCount

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::ActiveSmallCoreCount

Number of small cores in processor to enable.

And support up to 63 cores.

- 0: Disable all cores for Hybrid CPU; Active all cores for Non-Hybrid CPU (MAX 64).
- 1: 1 core
- 2: 2 cores
- 3: 3 cores

Definition at line 148 of file CpuConfigLibPreMemConfig.h.

13.24.2.4 ActiveSocNorthAtomCoreCount

UINT8 CPU_CONFIG_LIB_PREMEM_CONFIG::ActiveSocNorthAtomCoreCount

Number of enabled atom cores on SoC.

Default is set to supported core number.

- 0 - all core disabled
- 1 - enable 1 core
- 2 - enable 2 cores
- others - reserved for future use

Definition at line 185 of file CpuConfigLibPreMemConfig.h.

13.24.2.5 BclkSource

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::BclkSource

Clock source of BCLK OC Frequency.

- 0: SOC
- 1: **CPU**
- 2: PCH
- 3: Ext. Clock (optional)

Definition at line 214 of file CpuConfigLibPreMemConfig.h.

13.24.2.6 BootFrequency

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::BootFrequency

Sets the boot frequency starting from reset vector.

- 0: Maximum battery performance.
- 1: Maximum non-turbo performance -2: **Turbo performance.**

Note

If Turbo is selected BIOS will start in max non-turbo mode and switch to Turbo mode.

Definition at line 71 of file CpuConfigLibPreMemConfig.h.

13.24.2.7 ComputeDieSscEnable

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::ComputeDieSscEnable

Enable or Disable Compute Die SSC configuration.

- **0: Disable**
- 1: Enable

Definition at line 163 of file CpuConfigLibPreMemConfig.h.

13.24.2.8 ComputeDieSscValue

UINT8 CPU_CONFIG_LIB_PREMEM_CONFIG::ComputeDieSscValue

SSC configuration value of Compute Die.

SSC downspread field encoding. Downspread in percent = SSC_DOWNSPREAD / 100.

- **0 : invalid**
- 1 : 0.01%
- 2 : 0.02%
- 100 : 1%

Definition at line 195 of file CpuConfigLibPreMemConfig.h.

13.24.2.9 CpuRatio

UINT8 CPU_CONFIG_LIB_PREMEM_CONFIG::CpuRatio

CpuRatio - Max non-turbo ratio (Flexible Ratio Boot) is set to CpuRatio.

0: Disabled If disabled, doesn't override max-non turbo ratio.

Definition at line 175 of file CpuConfigLibPreMemConfig.h.

13.24.2.10 CrashLogEnable

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::CrashLogEnable

Enable or Disable CrashLog feature.

- 0: Disable
- **1: Enable**

Definition at line 101 of file CpuConfigLibPreMemConfig.h.

13.24.2.11 CrashLogGprs

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::CrashLogGprs

Enable or Disable CrashLog GPRs dump.

- **0: Disable**
- 1: Gprs Enabled, Smm Gprs Enabled 2: Gprs Enabled, Smm Gprs Disabled

Definition at line 156 of file CpuConfigLibPreMemConfig.h.

13.24.2.12 FClkFrequency

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::FClkFrequency

Processor Early Power On Configuration FCLK setting.

- **0: 800 MHz (ULT/ULX).**
- **1: 1 GHz (DT/Halo).** Not supported on ULT/ULX.
- 2: 400 MHz.
- 3: Reserved.

Definition at line 95 of file CpuConfigLibPreMemConfig.h.

13.24.2.13 InfDfxPwrEnable

UINT8 CPU_CONFIG_LIB_PREMEM_CONFIG::InfDfxPwrEnable

Determine whether to keep or shutdown INF_DFX power rail.

INF_DFX power is up from cold boot.

- **0 : INF_DFX shutdown**
- 1 : INF_DFX power up

Definition at line 221 of file CpuConfigLibPreMemConfig.h.

13.24.2.14 PeciC10Reset

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::PeciC10Reset

Enables the mailbox command to resolve PECI reset issues during Pkg-C10 exit.

If Enabled, BIOS will send the CPU message to disable peci reset on C10 exit. The default value **0: Disable** for CPU's

- 0: Disable
- 1: Enable

Deprecated no longer used.

Definition at line 138 of file CpuConfigLibPreMemConfig.h.

13.24.2.15 PeciSxReset

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::PeciSxReset

Enables a mailbox command to resolve rare PECI related Sx issues.

Note

This should only be used on systems that observe PECI Sx issues.

- **0: Disable**
- 1: Enable

Definition at line 129 of file CpuConfigLibPreMemConfig.h.

13.24.2.16 SocDieSscEnable

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::SocDieSscEnable

Enable or Disable Soc Die SSC configuration.

- **0: Disable**
- 1: Enable

Definition at line 170 of file CpuConfigLibPreMemConfig.h.

13.24.2.17 SocDieSscValue

UINT8 CPU_CONFIG_LIB_PREMEM_CONFIG::SocDieSscValue

SSC configuration value of Soc Die.

SSC downspread field encoding. Downspread in percent = SSC_DOWNSPREAD / 100.

- **0 : invalid**
- 1 : 0.01%
- 2 : 0.02%
- 100 : 1%

Definition at line 205 of file CpuConfigLibPreMemConfig.h.

13.24.2.18 TmeEnable

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::TmeEnable

Enable or Disable Total Memory Encryption (TME) feature.

- 0: Disable
- **1: Enable**

Definition at line 108 of file CpuConfigLibPreMemConfig.h.

13.24.2.19 VmxEnable

UINT32 CPU_CONFIG_LIB_PREMEM_CONFIG::VmxEnable

Enable or Disable Virtual Machine Extensions (VMX) feature.

- 0: Disable
- **1: Enable**

Definition at line 87 of file CpuConfigLibPreMemConfig.h.

The documentation for this struct was generated from the following file:

- [CpuConfigLibPreMemConfig.h](#)

13.25 CPU_DMI_EQ_PARAM Struct Reference

Represent lane specific Dmi Gen3 equalization parameters.

```
#include <CpuDmiPreMemConfig.h>
```

Public Attributes

- UINT8 [Cm](#)
Coefficient C-1.
- UINT8 [Cp](#)
Coefficient C+1.
- UINT8 [Rsvd0](#) [2]
Reserved bytes.

13.25.1 Detailed Description

Represent lane specific Dmi Gen3 equalization parameters.

Definition at line 61 of file CpuDmiPreMemConfig.h.

The documentation for this struct was generated from the following file:

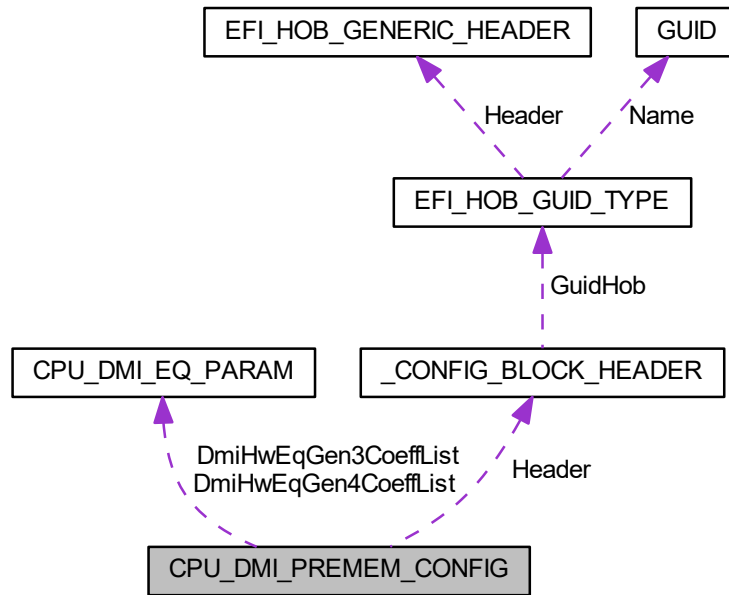
- [CpuDmiPreMemConfig.h](#)

13.26 CPU_DMI_PREMEM_CONFIG Struct Reference

The CPU_DMI_CONFIG block describes the expected configuration of the CPU for DMI.

```
#include <CpuDmiPreMemConfig.h>
```

Collaboration diagram for CPU_DMI_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- [UINT8 DmiMaxLinkSpeed](#)
- [UINT8 DmiGen3EqPh2Enable](#)
(Test) DMI Equalization Phase 2 Enable Control
- [UINT8 DmiGen3EqPh3Method](#)
(Test) Selects the method for performing Phase3 of Gen3 Equalization on DMI
- [UINT8 DmiGen3ProgramStaticEq](#)
(Test) Program DMI Gen3 EQ Phase1 Static Presets
- [UINT8 DmiDeEmphasis](#)
DeEmphasis control for DMI (-6 dB and -3.5 dB are the options)
- [UINT8 DmiAspmCtrl](#)
*ASPM configuration on the CPU side of the DMI/OPI Link. Default is **DmiAspmAutoConfig***
- [UINT8 DmiAspmL1ExitLatency](#)
*ASPM configuration on the CPU side of the DMI/OPI Link. Default is **DmiAspmAutoConfig***
- [UINT8 DmiGen3RootPortPreset](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen3 preset values per lane. Range: 0-9, 8 is default for each lane.

- UINT8 [DmiGen3EndPointPreset](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen3 preset values per lane. Range: 0-9, 7 is default for each lane.
- UINT8 [DmiGen3EndPointHint](#) [SA_DMI_MAX_LANE]
Hint value per lane for the DMI Gen3 End Point. Range: 0-6, 2 is default for each lane.
- UINT8 [DmiGen3RxCtlePeaking](#) [SA_DMI_MAX_BUNDLE]
DMI Gen3 RxCTLEp per-Bundle control.
- UINT8 [DmiGen3DsPortRxPreset](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen3 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen3DsPortTxPreset](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen3 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen3UsPortRxPreset](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen3 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen3UsPortTxPreset](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen3 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen3Ltcpre](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen3 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen3Ltcpo](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen3 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen3RtcoCpre](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen3 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen3RtcoCpo](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen3 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [Rsvd0](#) [1]
Reserved bytes.
- UINT8 [DmiGen4Ltcpre](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen4 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen4Ltcpo](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen4 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen4DsPortTxPreset](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen4 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen4UsPortTxPreset](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen4 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen4RtcoCpre](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen4 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen4RtcoCpo](#) [SA_DMI_MAX_LANE]
Used for programming DMI Gen4 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiNewFom](#)
Used for programming DMI New FOM.
- UINT8 [DmiCdrRelock](#)
Used for programming DMI CDR Relock.

13.26.1 Detailed Description

The CPU_DMI_CONFIG block describes the expected configuration of the CPU for DMI.

Revision 1:

- Initial version. **Revision 2:**
- Added Policies for Dmi Higer Speed Training. **Revision 3:**
- Added DmiCdrRelock Policy.

Definition at line 76 of file CpuDmiPreMemConfig.h.

13.26.2 Member Data Documentation

13.26.2.1 DmiGen3EqPh2Enable

UINT8 CPU_DMI_PREMEM_CONFIG::DmiGen3EqPh2Enable

(Test) DMI Equalization Phase 2 Enable Control

- Disabled (0x0) : Disable phase 2
- Enabled (0x1) : Enable phase 2
- **Auto** (0x2) : Use the current default method (Default)

Definition at line 92 of file CpuDmiPreMemConfig.h.

13.26.2.2 DmiGen3EqPh3Method

UINT8 CPU_DMI_PREMEM_CONFIG::DmiGen3EqPh3Method

(Test) Selects the method for performing Phase3 of Gen3 Equalization on DMI

- **Auto** (0x0) : Use the current default method (Default)
- HwEq (0x1) : Use Adaptive Hardware Equalization
- SwEq (0x2) : Use Adaptive Software Equalization (Implemented in BIOS Reference Code)
- Static (0x3) : Use the Static EQs provided in DmiGen3EndPointPreset array for Phase1 AND Phase3 (Instead of just Phase1)
- Disabled (0x4) : Bypass Equalization Phase 3

Definition at line 101 of file CpuDmiPreMemConfig.h.

13.26.2.3 DmiGen3ProgramStaticEq

UINT8 CPU_DMI_PREMEM_CONFIG::DmiGen3ProgramStaticEq

(Test) Program DMI Gen3 EQ Phase1 Static Presets

- Disabled (0x0) : Disable EQ Phase1 Static Presets Programming
- **Enabled** (0x1) : Enable EQ Phase1 Static Presets Programming (Default)

Definition at line 107 of file CpuDmiPreMemConfig.h.

13.26.2.4 DmiGen3RxCtlePeaking

```
UINT8 CPU_DMI_PREMEM_CONFIG::DmiGen3RxCtlePeaking[SA_DMI_MAX_BUNDLE]
```

DMI Gen3 RxCTLEp per-Bundle control.

The range of the setting is (0-15). This setting has to be specified based upon platform design and must follow the guideline. Default is 12.

Definition at line 121 of file CpuDmiPreMemConfig.h.

13.26.2.5 DmiMaxLinkSpeed

```
UINT8 CPU_DMI_PREMEM_CONFIG::DmiMaxLinkSpeed
```

- **Auto** (0x0) : Maximum possible link speed (Default)
- Gen1 (0x1) : Limit Link to Gen1 Speed
- Gen2 (0x2) : Limit Link to Gen2 Speed CpuDmiPreMemConfig
- Gen3 (0x3) : Limit Link to Gen3 Speed

Definition at line 85 of file CpuDmiPreMemConfig.h.

The documentation for this struct was generated from the following file:

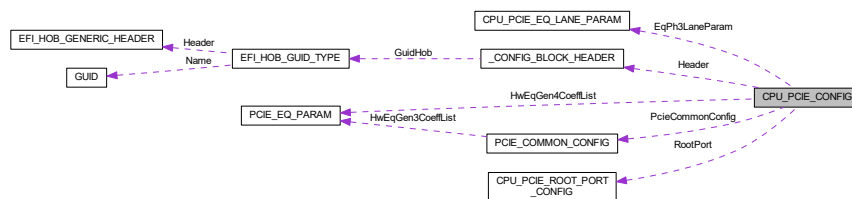
- [CpuDmiPreMemConfig.h](#)

13.27 CPU_PCIE_CONFIG Struct Reference

The [CPU_PCIE_CONFIG](#) block describes the expected configuration of the CPU PCI Express controllers **Revision 1< / b>: -Initial version.**

```
#include <CpuPcieConfig.h>
```

Collaboration diagram for CPU_PCIE_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [CPU_PCIE_ROOT_PORT_CONFIG](#) RootPort [CPU_PCIE_MAX_ROOT_PORTS]
These members describe the configuration of each SA PCIe root port.
- [CPU_PCIE_EQ_LANE_PARAM](#) EqPh3LaneParam [SA_PEG_MAX_LANE]
Gen3 Equalization settings for physical PCIe lane, index 0 represents PCIe lane 1, etc.
- [PCIE_EQ_PARAM](#) HwEqGen4CoeffList [PCIE_HWEQ_COEFFS_MAX]
List of coefficients used during equalization (applicable to both software and hardware EQ)
- UINT32 [FiaProgramming](#): 1
< (Test) Includes policies which are common to both SA and PCH PCIe
- UINT32 [PegGen3ProgramStaticEq](#): 1
< Skip Fia Configuration and lock if enable
- UINT32 [PegGen4ProgramStaticEq](#): 1
(Test) Program PEG Gen4 EQ Phase1 Static Presets
- UINT32 [SetSecuredRegisterLock](#): 1
(Test) Cpu Pcie Secure Register Lock
- UINT32 [SlotSelection](#): 1
*This member allows to select between the PCI Express M2 or CEMx4 slot 1: **PCIe M2**; 0: CEMx4 slot.*
- UINT32 [PcieDeviceOverrideTablePtr](#)
PCIe device override table The PCIe device table is being used to override PCIe device ASPM settings.

13.27.1 Detailed Description

The [CPU_PCIE_CONFIG](#) block describes the expected configuration of the CPU PCI Express controllers **Revision 1** < / b>: **-Initial version.**

Revision 2:

- SlotSelection policy added **Revision 3**
- Deprecate PegGen3ProgramStaticEq and PegGen4ProgramStaticEq **Revision 4:**
- Deprecating SetSecuredRegisterLock **Revision 5:**
- Moved ClockGating policy to PCIE_ROOT_PORT_COMMON_CONFIG
- Moved PowerGating policy to PCIE_ROOT_PORT_COMMON_CONFIG
- Deprecate VcEnabled policy **Revision 7:**
- Deprecating Gen3PresetCoeffSelection and Gen4PresetCoeffSelection **Revision 8:**
- Added SerI policy **Revision 9:**
- Align revision with CPU_PCIE_CONFIG_REVISION value **Revision 10:**
- Deprecate EqPh3LaneParam.Cm and EqPh3LaneParam.Cp

Definition at line 310 of file CpuPcieConfig.h.

13.27.2 Member Data Documentation

13.27.2.1 EqPh3LaneParam

`CPU_PCIE_EQ_LANE_PARAM CPU_PCIE_CONFIG::EqPh3LaneParam[SA_PEG_MAX_LANE]`

Gen3 Equalization settings for physical PCIe lane, index 0 represents PCIe lane 1, etc.

Corresponding entries are used when root port EqPh3Method is PchPcieEqStaticCoeff (default).

Definition at line 320 of file CpuPcieConfig.h.

13.27.2.2 PcieDeviceOverrideTablePtr

`UINT32 CPU_PCIE_CONFIG::PcieDeviceOverrideTablePtr`

PCIe device override table The PCIe device table is being used to override PCIe device ASPM settings.

This is a pointer points to a 32bit address. And it's only used in PostMem phase. Please refer to [PCIE_DEVICE_OVERRIDE](#) structure for the table. Last entry VendorId must be 0. The prototype of this policy is: [PCIE_DEVICE_OVERRIDE](#) *PcieDeviceOverrideTablePtr;

Definition at line 368 of file CpuPcieConfig.h.

13.27.2.3 PegGen3ProgramStaticEq

`UINT32 CPU_PCIE_CONFIG::PegGen3ProgramStaticEq`

< Skip Fia Configuration and lock if enable

(Test) Program PEG Gen3 EQ Phase1 Static Presets

- Disabled (0x0) : Disable EQ Phase1 Static Presets Programming
- **Enabled** (0x1) : Enable EQ Phase1 Static Presets Programming (Default)

Definition at line 335 of file CpuPcieConfig.h.

13.27.2.4 PegGen4ProgramStaticEq

`UINT32 CPU_PCIE_CONFIG::PegGen4ProgramStaticEq`

(Test) Program PEG Gen4 EQ Phase1 Static Presets

- Disabled (0x0) : Disable EQ Phase1 Static Presets Programming
- **Enabled** (0x1) : Enable EQ Phase1 Static Presets Programming (Default)

Definition at line 343 of file CpuPcieConfig.h.

13.27.2.5 SetSecuredRegisterLock

```
UINT32 CPU_PCIE_CONFIG::SetSecuredRegisterLock
```

(Test) Cpu Pcie Secure Register Lock

- Disabled (0x0)
- **Enabled** (0x1)

Definition at line 349 of file CpuPcieConfig.h.

The documentation for this struct was generated from the following file:

- [CpuPcieConfig.h](#)

13.28 CPU_PCIE_EQ_LANE_PARAM Struct Reference

Represent lane specific PCIe Gen3 equalization parameters.

```
#include <CpuPcieConfig.h>
```

Public Attributes

- UINT8 [Cm](#)
Coefficient C-1.
- UINT8 [Cp](#)
Coefficient C+1.
- UINT8 [PegGen3RootPortPreset](#)
(Test) Used for programming PEG Gen3 preset values per lane. Range: 0-9, 8 is default for each lane
- UINT8 [PegGen3EndPointPreset](#)
(Test) Used for programming PEG Gen3 preset values per lane. Range: 0-9, 7 is default for each lane
- UINT8 [PegGen3EndPointHint](#)
(Test) Hint value per lane for the PEG Gen3 End Point. Range: 0-6, 2 is default for each lane
- UINT8 [PegGen4RootPortPreset](#)
(Test) Used for programming PEG Gen4 preset values per lane. Range: 0-9, 8 is default for each lane
- UINT8 [PegGen4EndPointPreset](#)
(Test) Used for programming PEG Gen4 preset values per lane. Range: 0-9, 7 is default for each lane
- UINT8 [PegGen4EndPointHint](#)
(Test) Hint value per lane for the PEG Gen4 End Point. Range: 0-6, 2 is default for each lane

13.28.1 Detailed Description

Represent lane specific PCIe Gen3 equalization parameters.

Definition at line 231 of file CpuPcieConfig.h.

The documentation for this struct was generated from the following file:

- [CpuPcieConfig.h](#)

13.29 CPU_PCIE_GPIO_INFO Struct Reference

CPU PCIe GPIO Data Structure.

```
#include <HybridGraphicsConfig.h>
```

Public Attributes

- UINT8 [ExpanderNo](#)
Offset 0 Expander No For I2C based GPIO.
- BOOLEAN [Active](#)
Offset 1 0=Active Low; 1=Active High.
- UINT8 [Rsvd0](#) [2]
Offset 2 Reserved.
- UINT32 [GpioNo](#)
Offset 4 GPIO pad.

13.29.1 Detailed Description

CPU PCIe GPIO Data Structure.

Definition at line 53 of file HybridGraphicsConfig.h.

The documentation for this struct was generated from the following file:

- [HybridGraphicsConfig.h](#)

13.30 CPU_PCIE_ROOT_PORT_CONFIG Struct Reference

The CPU_PCIE_ROOT_PORT_CONFIG describe the feature and capability of each CPU PCIe root port.

```
#include <CpuPcieConfig.h>
```

Public Attributes

- UINT32 [ExtSync](#): 1
*Indicate whether the extended synch is enabled. 0: **Disable**; 1: **Enable**.*
- UINT32 [MultiVcEnabled](#): 1
*Multiple Virtual Channel. 0: **Disable**; 1: **Enable***
- UINT32 [RsvdBits0](#): 30
Reserved bits.
- UINT8 [Gen4EqPh3Method](#)
PCIe Gen4 Equalization Method.
- UINT8 [FomsCp](#)
FOM Score Board Control Policy.
- UINT8 [RsvdBytes0](#) [2]
Reserved bytes.

- UINT32 [Gen3Utp](#): 4
(Test) Upstream Port Transmitter Preset used during Gen3 Link Equalization. Used for all lanes. Default is 7.
- UINT32 [Gen3Dtp](#): 4
(Test) Downstream Port Transmitter Preset used during Gen3 Link Equalization. Used for all lanes. Default is 7.
- UINT32 [Gen4Utp](#): 4
(Test) Upstream Port Transmitter Preset used during Gen4 Link Equalization. Used for all lanes. Default is 7.
- UINT32 [Gen4Dtp](#): 4
(Test) Downstream Port Transmitter Preset used during Gen4 Link Equalization. Used for all lanes. Default is 7.
- UINT32 [Gen5Utp](#): 4
(Test) Upstream Port Transmitter Preset used during Gen5 Link Equalization. Used for all lanes. Default is 5.
- UINT32 [Gen5Dtp](#): 4
(Test) Downstream Port Transmitter Preset used during Gen5 Link Equalization. Used for all lanes. Default is 7.
- UINT32 [RsvdBits1](#): 8
Reserved Bits.
- PCIE_ROOT_PORT_COMMON_CONFIG [PcieRpCommonConfig](#)
(Test) Includes policies which are common to both SA and PCH RootPort

13.30.1 Detailed Description

The CPU_PCI_ROOT_PORT_CONFIG describe the feature and capability of each CPU PCIe root port.

Definition at line 245 of file CpuPcieConfig.h.

13.30.2 Member Data Documentation

13.30.2.1 Gen4EqPh3Method

UINT8 CPU_PCIE_ROOT_PORT_CONFIG::Gen4EqPh3Method

PCIe Gen4 Equalization Method.

- HwEq (0x1) : Hardware Equalization (Default)
- StaticEq (0x2) : Static Equalization

Definition at line 255 of file CpuPcieConfig.h.

The documentation for this struct was generated from the following file:

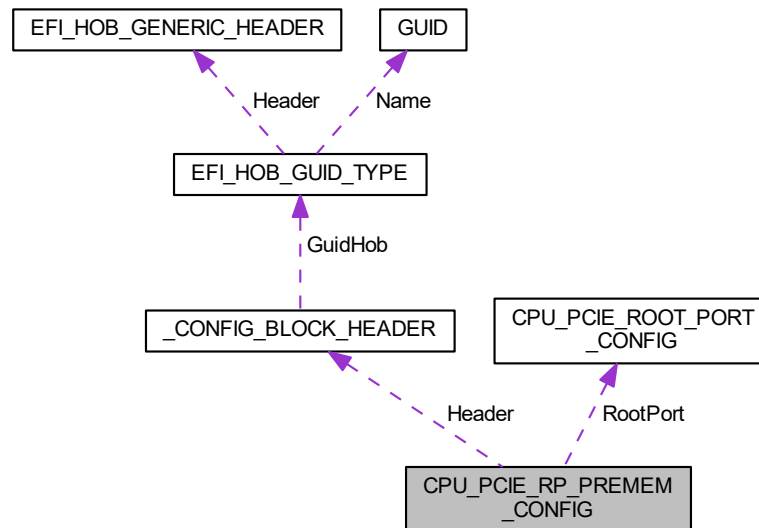
- [CpuPcieConfig.h](#)

13.31 CPU_PCIE_RP_PREMEM_CONFIG Struct Reference

CPU PCIe Root Port Pre-Memory Configuration Contains Root Port settings and capabilities **Revision 1**: - Initial version.

```
#include <CpuPcieConfig.h>
```

Collaboration diagram for CPU_PCIE_RP_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [RpEnabledMask](#)
Root Port enabling mask.
- UINT8 [LinkDownGpios](#)
Assertion on Link Down GPIOs.
- UINT8 [ClkReqMsgEnable](#)
Enable ClockReq Messaging.
- UINT8 [DekelSquelchWa](#)
Dekel Recipe Workaround 2 1=Minimal, 9=Maximum,.
- UINT8 [PcieSpeed](#) [CPU_PCIE_MAX_ROOT_PORTS]
Determines each PCIE Port speed capability.
- UINT8 [ClkReqMsgEnableRp](#) [CPU_PCIE_MAX_ROOT_PORTS]
Enable ClockReq Messaging Policy is for all Rootports.
- UINT8 [NewFom](#) [CPU_PCIE_MAX_ROOT_PORTS]
*To Enable/Disable New FOM 0: **Disable**; 1: Enable.*
- UINT8 [CdrRelock](#) [CPU_PCIE_MAX_ROOT_PORTS]
*To Enable/Disable CDR Relock 0: **Disable**; 1: Enable.*
- UINT32 [SlotImplemented](#): 1
Indicates how this root port is connected to endpoint.

13.31.1 Detailed Description

CPU PCIe Root Port Pre-Memory Configuration Contains Root Port settings and capabilities **Revision 1:** - Initial version.

Revision 2: - Adding Dekel Suqelch Workaround Setup Variable **Revision 3:** - Deprecate Dekel Suqelch Workaround Setup Variable **Revision 4:** - Adding New FOM Setup Variable **Revision 5:** - Add CdrRelock Policy to [CPU_PCIE_RP_PREMEM_CONFIG](#). **Revision 6:** - Add SlotImplemented to CPU_PCIE_HOB. **Revision 7:** - Add [CPU_PCIE_ROOT_PORT_CONFIG](#).

Definition at line 386 of file CpuPcieConfig.h.

13.31.2 Member Data Documentation

13.31.2.1 ClkReqMsgEnable

```
UINT8 CPU_PCIE_RP_PREMEM_CONFIG::ClkReqMsgEnable
```

Enable ClockReq Messaging.

- **Disabled** (0x0) : Disable ClockReq Messaging(Default)
- **Enabled** (0x1) : Enable ClockReq Messaging

Definition at line 405 of file CpuPcieConfig.h.

13.31.2.2 ClkReqMsgEnableRp

```
UINT8 CPU_PCIE_RP_PREMEM_CONFIG::ClkReqMsgEnableRp[CPU_PCIE_MAX_ROOT_PORTS]
```

Enable ClockReq Messaging Policy is for all Rootports.

- **Disabled** (0x0) : Disable ClockReq Messaging(Default)
- **Enabled** (0x1) : Enable ClockReq Messaging

Definition at line 424 of file CpuPcieConfig.h.

13.31.2.3 LinkDownGpios

UINT8 CPU_PCIE_RP_PREMEM_CONFIG::LinkDownGpios

Assertion on Link Down GPIOs.

- **Disabled** (0x0) : Disable assertion on Link Down GPIOs(Default)
- **Enabled** (0x1) : Enable assertion on Link Down GPIOs

Definition at line 399 of file CpuPcieConfig.h.

13.31.2.4 PcieSpeed

UINT8 CPU_PCIE_RP_PREMEM_CONFIG::PcieSpeed[CPU_PCIE_MAX_ROOT_PORTS]

Determines each PCIE Port speed capability.

0: Auto; 1: Gen1; 2: Gen2; 3: Gen3; 4: Gen4 (see: CPU_PCIE_SPEED)

Definition at line 417 of file CpuPcieConfig.h.

13.31.2.5 RpEnabledMask

UINT32 CPU_PCIE_RP_PREMEM_CONFIG::RpEnabledMask

Root Port enabling mask.

Bit0 presents RP1, Bit1 presents RP2, and so on. 0: Disable; **1: Enable**.

Definition at line 393 of file CpuPcieConfig.h.

13.31.2.6 SlotImplemented

UINT32 CPU_PCIE_RP_PREMEM_CONFIG::SlotImplemented

Indicates how this root port is connected to endpoint.

0: built-in device; **1: slot** Built-in is incompatible with hotplug-capable ports.

Definition at line 439 of file CpuPcieConfig.h.

The documentation for this struct was generated from the following file:

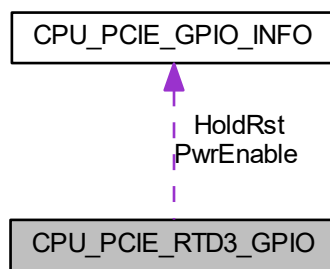
- [CpuPcieConfig.h](#)

13.32 CPU_PCIE_RTD3_GPIO Struct Reference

CPU PCIE RTD3 GPIO Data Structure.

```
#include <HybridGraphicsConfig.h>
```

Collaboration diagram for CPU_PCIE_RTD3_GPIO:



Public Attributes

- [CPU_PCIE_GPIO_INFO HoldRst](#)
Offset 0 This field contain PCIe HLD RESET GPIO value and level information.
- [CPU_PCIE_GPIO_INFO PwrEnable](#)
Offset 8 This field contain PCIe PWR Enable GPIO value and level information.
- UINT32 [WakeGpioNo](#)
Offset 16 This field contain PCIe RTD3 Device Wake GPIO Number.
- UINT8 [GpioSupport](#)
*Offset 20 Depends on board design the GPIO configuration may be different: **0=Not Supported**, 1=PCH Based, 2=I2C based.*
- UINT8 [Rsvd0](#) [3]
Offset 21.

13.32.1 Detailed Description

CPU PCIE RTD3 GPIO Data Structure.

Definition at line 63 of file HybridGraphicsConfig.h.

The documentation for this struct was generated from the following file:

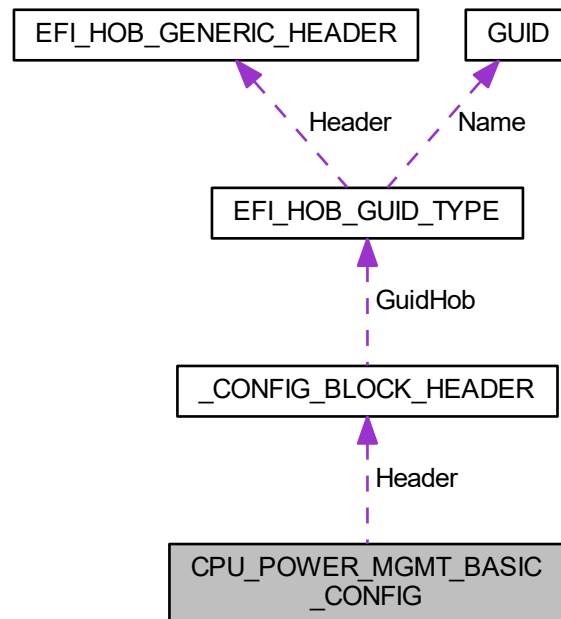
- [HybridGraphicsConfig.h](#)

13.33 CPU_POWER_MGMT_BASIC_CONFIG Struct Reference

CPU Power Management Basic Configuration Structure.

```
#include <CpuPowerMgmtBasicConfig.h>
```

Collaboration diagram for CPU_POWER_MGMT_BASIC_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [BootFrequency](#): 2
Sets the boot frequency starting from reset vector.
- UINT32 [SkipSetBootPState](#): 1
Choose whether to skip SetBootPState function for all APs; 0: Do not skip; 1: Skip.
- UINT32 [Hwp](#): 2
Enable or Disable Intel Speed Shift Technology.
- UINT32 [HdcControl](#): 2
Hardware Duty Cycle Control configuration.
- UINT32 [PowerLimit2](#): 1
Enable or Disable short duration Power Limit (PL2). 0: Disable; 1: Enable
- UINT32 [TurboPowerLimitLock](#): 1
MSR 0x610[63] and 0x618[63]: Locks all Turbo power limit settings to read-only; 0: Disable; 1: Enable (Lock).
- UINT32 [PowerLimit3DutyCycle](#): 8
Package PL3 Duty Cycle. Specifies the PL3 duty cycle percentage, Range 0-100. Default: 0.

- UINT32 [PowerLimit3Lock](#): 1
*Package PL3 MSR 615h lock; 0: **Disable**; 1: Enable (Lock).*
- UINT32 [PowerLimit4Lock](#): 1
*Package PL4 MSR 601h lock; 0: **Disable**; 1: Enable (Lock).*
- UINT32 [TccOffsetClamp](#): 1
Tcc Offset Clamp for Runtime Average Temperature Limit (RATL) allows CPU to throttle below P1.
- UINT32 [TccOffsetLock](#): 1
*Tcc Offset Lock for Runtime Average Temperature Limit (RATL) to lock temperature target MSR 1A2h; 0: Disabled; 1: **Enabled (Lock)**.*
- UINT32 [TurboMode](#): 1
*Enable or Disable Turbo Mode. Disable; 1: **Enable***
- UINT32 [HwplInterruptControl](#): 1
*Set HW P-State Interrupts Enabled for MISC_PWR_MGMT MSR 0x1AA[7]; 0: **Disable**; 1: Enable.*
- UINT32 [ApplyConfigTdp](#): 1
*Switch TDP applied setting based on non-cTDP or TDP; 0: non-cTDP; 1: **cTDP**.*
- UINT32 [HwpLock](#): 1
*HWP Lock in MISC PWR MGMT MSR 1AAh; 0: **Disable**; 1: Enable (Lock).*
- UINT32 [DualTauBoost](#): 1
Dual Tau Base Performance Boost support.
- UINT32 [PlatformAtxTelemetryUnit](#): 1
PMC_SUS_SPARE_GCR_0 bit 0, set by BIOS to indicate the telemetry type for Desktop Psys support.
- UINT32 [RsvdBits](#): 4
Reserved for future use.
- UINT8 [OneCoreRatioLimit](#)
1-Core Ratio Limit: LFM to Fused 1-Core Ratio Limit.
- UINT8 [TwoCoreRatioLimit](#)
2-Core Ratio Limit: LFM to Fused 2-Core Ratio Limit, For overclocking part: LFM to Fused 2-Core Ratio Limit + OC Bins.
- UINT8 [ThreeCoreRatioLimit](#)
3-Core Ratio Limit: LFM to Fused 3-Core Ratio Limit, For overclocking part: LFM to Fused 3-Core Ratio Limit + OC Bins.
- UINT8 [FourCoreRatioLimit](#)
4-Core Ratio Limit: LFM to Fused 4-Core Ratio Limit, For overclocking part: LFM to Fused 4-Core Ratio Limit + OC Bins.
- UINT8 [FiveCoreRatioLimit](#)
5-Core Ratio Limit: LFM to Fused 5-Core Ratio Limit, For overclocking part: LFM to Fused 5-Core Ratio Limit + OC Bins.
- UINT8 [SixCoreRatioLimit](#)
6-Core Ratio Limit: LFM to Fused 6-Core Ratio Limit, For overclocking part: LFM to Fused 6-Core Ratio Limit + OC Bins.
- UINT8 [SevenCoreRatioLimit](#)
7-Core Ratio Limit: LFM to Fused 7-Core Ratio Limit, For overclocking part: LFM to Fused 7-Core Ratio Limit + OC Bins.
- UINT8 [EightCoreRatioLimit](#)
8-Core Ratio Limit: LFM to Fused 8-Core Ratio Limit, For overclocking part: LFM to Fused 8-Core Ratio Limit + OC Bins.
- UINT8 [TccActivationOffset](#)
TCC Activation Offset.
- UINT8 [EnableIbTbm](#): 1
Intel Turbo Boost Max Technology 3.0 Enabling it on processors with OS support will allow OS to exploit the diversity in max turbo frequency of the cores.
- UINT8 [EnableIbTbmDriver](#): 1
- UINT8 [EnablePerCorePState](#): 1

- Per Core P State OS control mode Disabling will set PCU_MISC_CONFIG (Command 0x06) Bit 31 = 1.*
- UINT8 [EnableHwpAutoPerCorePstate](#): 1
HwP Autonomous Per Core P State Disabling will set Bit 30 = 1, command 0x11.
 - UINT8 [EnableHwpAutoEppGrouping](#): 1
HwP Autonomous EPP grouping.
 - UINT8 [EnableEpbPeciOverride](#): 1
EPB override over PECl Enable by sending pcode command 0x2b , subcommand 0x3 to 1.
 - UINT8 [EnableFastMsrHwpReq](#): 1
Support for Fast MSR for IA32_HWP_REQUEST.
 - UINT8 [EnableHwpScalabilityTracking](#): 1
Enable/disable HWP Scalability tracking Disabling will set Bit 26 = 1, command 0x11.
 - UINT8 [MinRingRatioLimit](#)
Minimum Ring Ratio Limit. Range from 0 to Max Turbo Ratio. 0 = AUTO/HW Default.
 - UINT8 [MaxRingRatioLimit](#)
Maximum Ring Ratio Limit. Range from 0 to Max Turbo Ratio. 0 = AUTO/HW Default.
 - UINT16 [PowerLimit1](#)
Package Long duration turbo mode power limit (PL1).
 - UINT16 [PowerLimit2Power](#)
Package Short duration turbo mode power limit (PL2).
 - UINT16 [PowerLimit3](#)
Package PL3 power limit.
 - UINT16 [PowerLimit4](#)
Package PL4 power limit.
 - UINT32 [PowerLimit1Time](#)
Package Long duration turbo mode power limit (PL1) time window in seconds.
 - UINT32 [PowerLimit3Time](#)
Package PL3 time window. Range from 3ms to 64ms.
 - UINT32 [TccOffsetTimeWindowForRatl](#)
Tcc Offset Time Window can range from 5ms to 448000ms for Runtime Average Temperature Limit (RATL).
 - UINT8 [TurboRatioLimitRatio](#) [TURBO_RATIO_LIMIT_ARRAY_SIZE]
Turbo Ratio Limit Ratio array, TurboRatioLimitRatio[7-0] will pair with TurboRatioLimitNumCore[7-0] to determine the active core ranges for each frequency point.
 - UINT8 [TurboRatioLimitNumCore](#) [TURBO_RATIO_LIMIT_ARRAY_SIZE]
Turbo Ratio Limit Num Core array, TurboRatioLimitNumCore[7-0] will pair with TurboRatioLimitRatio[7-0] to determine the active core ranges for each frequency point.
 - UINT8 [AtomTurboRatioLimitRatio](#) [TURBO_RATIO_LIMIT_ARRAY_SIZE]
ATOM Turbo Ratio Limit Ratio array, AtomTurboRatioLimitRatio[7-0] will pair with AtomTurboRatioLimitNumCore[7-0] to determine the active core ranges for each frequency point.
 - UINT8 [AtomTurboRatioLimitNumCore](#) [TURBO_RATIO_LIMIT_ARRAY_SIZE]
ATOM Turbo Ratio Limit Num Core array, AtomTurboRatioLimitNumCore[7-0] will pair with AtomTurboRatioLimitRatio[7-0] to determine the active core ranges for each frequency point.

13.33.1 Detailed Description

CPU Power Management Basic Configuration Structure.

Revision 1:

- Initial version. **Revision 2:**
- Changed EnableIltbm default to be disable

- Deprecated EnableItbmDriver due to Platform doesn't have ITBMT OS driver **Revision 3:**
- Add ApplyConfigTdp for TDP initialization settings based on non-cTDP or cTDP **Revision 4:**
- Add Hwp Lock support **Revision 5:**
- Add TurboRatioLimitRatio, TurboRatioLimitNumCore, AtomTurboRatioLimitRatio and AtomTurboRatioLimitNumCore **Revision 6:**
- Add DualTauBoost **Revision 7:**
- Changed EnableItbm default to be enable **Revision 8:**
- Add StepDownMode and PlatformAtxTelemetryUnit **Revision 9:**
- Add EnableHwpScalabilityTracking

Definition at line 71 of file CpuPowerMgmtBasicConfig.h.

13.33.2 Member Data Documentation

13.33.2.1 AtomTurboRatioLimitNumCore

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::AtomTurboRatioLimitNumCore[TURBO_RATIO_LIMIT_ARRAY_SIZE]
```

ATOM Turbo Ratio Limit Num Core array, AtomTurboRatioLimitNumCore[7-0] will pair with AtomTurboRatioLimitRatio[7-0] to determine the active core ranges for each frequency point.

Default value is from hardware register MSR 0x651. Note:

- These policies are for platforms support ATOM MSR pair 0x650 and 0x651.

Definition at line 318 of file CpuPowerMgmtBasicConfig.h.

13.33.2.2 AtomTurboRatioLimitRatio

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::AtomTurboRatioLimitRatio[TURBO_RATIO_LIMIT_ARRAY_SIZE]
```

ATOM Turbo Ratio Limit Ratio array, AtomTurboRatioLimitRatio[7-0] will pair with AtomTurboRatioLimitNumCore[7-0] to determine the active core ranges for each frequency point.

Default value is from hardware register MSR 0x650. Note:

- These policies are for platforms support ATOM MSR pair 0x650 and 0x651.

Definition at line 310 of file CpuPowerMgmtBasicConfig.h.

13.33.2.3 BootFrequency

UINT32 CPU_POWER_MGMT_BASIC_CONFIG::BootFrequency

Sets the boot frequency starting from reset vector.

- 0: Maximum battery performance.
- 1: Maximum non-turbo performance.
- **2: Turbo performance.**

Note

If Turbo is selected BIOS will start in max non-turbo mode and switch to Turbo mode.

Definition at line 80 of file CpuPowerMgmtBasicConfig.h.

13.33.2.4 DualTauBoost

UINT32 CPU_POWER_MGMT_BASIC_CONFIG::DualTauBoost

Dual Tau Base Performance Boost support.

This will improve platform performance for desktop sku. When enabled, it exposes 2 MMIO PL1 and MMIO Tau registers, and also provides the default value.

Note

- : This is only applicable for Desktop 35W/65W/125W sku.
- : When this is enabled, it will override the settings in policy PowerLimit1, PowerLimit1Time, PowerLimit2↔ Power. **0: Disable**; 1: Enable

Definition at line 119 of file CpuPowerMgmtBasicConfig.h.

13.33.2.5 EightCoreRatioLimit

UINT8 CPU_POWER_MGMT_BASIC_CONFIG::EightCoreRatioLimit

8-Core Ratio Limit: LFM to Fused 8-Core Ratio Limit, For overclocking part: LFM to Fused 8-Core Ratio Limit + OC Bins.

Default value is from hardware register MSR 0x1AD. Note: OC Bins = 7 means fully unlocked, so range is LFM to 83.

- This 8-Core Ratio Limit Must be Less than or equal to 1-Core Ratio Limit.
- This policy is for platforms only support MSR 0x1AD without MSR 0x1AE support.

Definition at line 187 of file CpuPowerMgmtBasicConfig.h.

13.33.2.6 EnableEpbPeciOverride

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::EnableEpbPeciOverride
```

EPB override over PECI Enable by sending pcode command 0x2b , subcommand 0x3 to 1.

This will allow OOB EPB PECI override control. **0: Disable; 1: Enable;**

Definition at line 235 of file CpuPowerMgmtBasicConfig.h.

13.33.2.7 EnableFastMsrHwpReq

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::EnableFastMsrHwpReq
```

Support for Fast MSR for IA32_HWP_REQUEST.

On systems with HwP enabled, if this feature is available as indicated by MSR 0x65F[0] = 1, set MSR 0x657[0] = 1. **0: Disable; 1: Enable;**

Definition at line 242 of file CpuPowerMgmtBasicConfig.h.

13.33.2.8 EnableHwpAutoEppGrouping

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::EnableHwpAutoEppGrouping
```

HWP Autonomous EPP grouping.

Disabling will set Bit 29 = 1, command 0x11. When set, autonomous will not necessarily request the same value for all cores with same EPP. Enabling will clean Bit 29 = 0, command 0x11. Autonomous will request same values for all cores with same EPP. **0: Disable; 1: Enable;**

Definition at line 228 of file CpuPowerMgmtBasicConfig.h.

13.33.2.9 EnableHwpAutoPerCorePstate

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::EnableHwpAutoPerCorePstate
```

HWP Autonomous Per Core P State Disabling will set Bit 30 = 1, command 0x11.

When set, autonomous will request the same value for all cores all the time. **0: Disable; 1: Enable;**

Definition at line 220 of file CpuPowerMgmtBasicConfig.h.

13.33.2.10 EnableHwpScalabilityTracking

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::EnableHwpScalabilityTracking
```

Enable/disable HWP Scalability tracking Disabling will set Bit 26 = 1, command 0x11.

Enabling will clean Bit 26 = 0, command 0x11. 0: Disable; **1: Enable;**

Definition at line 249 of file CpuPowerMgmtBasicConfig.h.

13.33.2.11 EnableItbm

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::EnableItbm
```

Intel Turbo Boost Max Technology 3.0 Enabling it on processors with OS support will allow OS to exploit the diversity in max turbo frequency of the cores.

0: Disable; 1: Enable;

Definition at line 201 of file CpuPowerMgmtBasicConfig.h.

13.33.2.12 EnableItbmDriver

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::EnableItbmDriver
```

Deprecated : Platform doesn't have Intel Turbo Boost Max Technology 3.0 Driver Enabling it will load the driver upon ACPI device with HID = INT3510. **0: Disable;** 1: Enable;

Definition at line 207 of file CpuPowerMgmtBasicConfig.h.

13.33.2.13 EnablePerCorePState

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::EnablePerCorePState
```

Per Core P State OS control mode Disabling will set PCU_MISC_CONFIG (Command 0x06) Bit 31 = 1.

When set, the highest core request is used for all other core requests. 0: Disable; **1: Enable;**

Definition at line 213 of file CpuPowerMgmtBasicConfig.h.

13.33.2.14 FiveCoreRatioLimit

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::FiveCoreRatioLimit
```

5-Core Ratio Limit: LFM to Fused 5-Core Ratio Limit, For overclocking part: LFM to Fused 5-Core Ratio Limit + OC Bins.

Note: OC Bins = 7 means fully unlocked, so range is LFM to 83.

- This 5-Core Ratio Limit Must be Less than or equal to 1-Core Ratio Limit.
- This policy is for platforms only support MSR 0x1AD without MSR 0x1AE support.

Definition at line 163 of file CpuPowerMgmtBasicConfig.h.

13.33.2.15 FourCoreRatioLimit

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::FourCoreRatioLimit
```

4-Core Ratio Limit: LFM to Fused 4-Core Ratio Limit, For overclocking part: LFM to Fused 4-Core Ratio Limit + OC Bins.

Default value is from hardware register MSR 0x1AD. Note: OC Bins = 7 means fully unlocked, so range is LFM to 83.

- This 4-Core Ratio Limit Must be Less than or equal to 1-Core Ratio Limit.
- This policy is for platforms only support MSR 0x1AD without MSR 0x1AE support.

Definition at line 156 of file CpuPowerMgmtBasicConfig.h.

13.33.2.16 HdcControl

```
UINT32 CPU_POWER_MGMT_BASIC_CONFIG::HdcControl
```

Hardware Duty Cycle Control configuration.

0: Disabled; **1: Enabled** 2-3:Reserved HDC enables the processor to autonomously force components to enter into an idle state to lower effective frequency. This allows for increased package level C6 residency.

Note

Currently this feature is recommended to be enabled only on win10

Definition at line 95 of file CpuPowerMgmtBasicConfig.h.

13.33.2.17 Hwp

UINT32 CPU_POWER_MGMT_BASIC_CONFIG::Hwp

Enable or Disable Intel Speed Shift Technology.

Enabling allows for processor control of P-state transitions. 0: Disable; **1: Enable**; Bit 1 is ignored.

Note

Currently this feature is recommended to be enabled only on win10

Definition at line 88 of file CpuPowerMgmtBasicConfig.h.

13.33.2.18 OneCoreRatioLimit

UINT8 CPU_POWER_MGMT_BASIC_CONFIG::OneCoreRatioLimit

1-Core Ratio Limit: LFM to Fused 1-Core Ratio Limit.

For overclocking parts: LFM to Fused 1-Core Ratio Limit + OC Bins. Default value is from hardware register MSR 0x1AD. Note: OC Bins = 7 means fully unlocked, so range is LFM to 83.

- This 1-Core Ratio Limit Must be greater than or equal to 2-Core Ratio Limit, 3-Core Ratio Limit, 4-Core Ratio Limit.
- This policy is for platforms only support MSR 0x1AD without MSR 0x1AE support.

Definition at line 132 of file CpuPowerMgmtBasicConfig.h.

13.33.2.19 PowerLimit1

UINT16 CPU_POWER_MGMT_BASIC_CONFIG::PowerLimit1

Package Long duration turbo mode power limit (PL1).

Default is the TDP power limit of processor. Units are based on POWER_MGMT_CONFIG.CustomPowerUnit.

Definition at line 256 of file CpuPowerMgmtBasicConfig.h.

13.33.2.20 PowerLimit1Time

UINT32 CPU_POWER_MGMT_BASIC_CONFIG::PowerLimit1Time

Package Long duration turbo mode power limit (PL1) time window in seconds.

Used in calculating the average power over time. Mobile: **28s** Desktop: **8s** Range: 0 - 128s

Definition at line 280 of file CpuPowerMgmtBasicConfig.h.

13.33.2.21 PowerLimit2Power

UINT16 CPU_POWER_MGMT_BASIC_CONFIG::PowerLimit2Power

Package Short duration turbo mode power limit (PL2).

Allows for short excursions above TDP power limit. Default = 1.25 * TDP Power Limit. Units are based on POWER_MGMT_CONFIG.CustomPowerUnit.

Definition at line 261 of file CpuPowerMgmtBasicConfig.h.

13.33.2.22 PowerLimit3

UINT16 CPU_POWER_MGMT_BASIC_CONFIG::PowerLimit3

Package PL3 power limit.

PL3 is the CPU Peak Power Occurences Limit. **Default: 0**. Range 0-65535. Units are based on POWER_MGMT_CONFIG.CustomPowerUnit.

Definition at line 266 of file CpuPowerMgmtBasicConfig.h.

13.33.2.23 PowerLimit4

UINT16 CPU_POWER_MGMT_BASIC_CONFIG::PowerLimit4

Package PL4 power limit.

PL4 is a Preemptive CPU Package Peak Power Limit, it will never be exceeded. Power is preemptively lowered before limit is reached. **Default: 0**. Range 0-65535. Units are based on POWER_MGMT_CONFIG.CustomPowerUnit.

Definition at line 272 of file CpuPowerMgmtBasicConfig.h.

13.33.2.24 SevenCoreRatioLimit

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::SevenCoreRatioLimit
```

7-Core Ratio Limit: LFM to Fused 7-Core Ratio Limit, For overlocking part: LFM to Fused 7-Core Ratio Limit + OC Bins.

Default value is from hardware register MSR 0x1AD. Note: OC Bins = 7 means fully unlocked, so range is LFM to 83.

- This 7-Core Ratio Limit Must be Less than or equal to 1-Core Ratio Limit.
- This policy is for platforms only support MSR 0x1AD without MSR 0x1AE support.

Definition at line 179 of file CpuPowerMgmtBasicConfig.h.

13.33.2.25 SixCoreRatioLimit

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::SixCoreRatioLimit
```

6-Core Ratio Limit: LFM to Fused 6-Core Ratio Limit, For overlocking part: LFM to Fused 6-Core Ratio Limit + OC Bins.

Default value is from hardware register MSR 0x1AD. Note: OC Bins = 7 means fully unlocked, so range is LFM to 83.

- This 6-Core Ratio Limit Must be Less than or equal to 1-Core Ratio Limit.
- This policy is for platforms only support MSR 0x1AD without MSR 0x1AE support.

Definition at line 171 of file CpuPowerMgmtBasicConfig.h.

13.33.2.26 TccActivationOffset

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::TccActivationOffset
```

TCC Activation Offset.

Offset from factory set TCC activation temperature at which the Thermal Control Circuit must be activated. T_{CC} will be activated at (TCC Activation Temperature - TCC Activation Offset), in degrees Celcius. For Y SKU, the recommended default for this policy is **10** For all other SKUs the recommended default are **0**, causing TCC to activate at TCC Activation temperature.

Note

The policy is recommended for validation purpose only.

Definition at line 195 of file CpuPowerMgmtBasicConfig.h.

13.33.2.27 TccOffsetClamp

UINT32 CPU_POWER_MGMT_BASIC_CONFIG::TccOffsetClamp

Tcc Offset Clamp for Runtime Average Temperature Limit (RATL) allows CPU to throttle below P1.

For Y SKU, the recommended default for this policy is **1: Enabled**, which indicates throttling below P1 is allowed. For all other SKUs the recommended default are **0: Disabled**.

Definition at line 106 of file CpuPowerMgmtBasicConfig.h.

13.33.2.28 TccOffsetTimeWindowForRatl

UINT32 CPU_POWER_MGMT_BASIC_CONFIG::TccOffsetTimeWindowForRatl

Tcc Offset Time Window can range from 5ms to 448000ms for Runtime Average Temperature Limit (RATL).

For Y SKU, the recommended default for this policy is **5000: 5 seconds**, For all other SKUs the recommended default are **0: Disabled**

Definition at line 286 of file CpuPowerMgmtBasicConfig.h.

13.33.2.29 ThreeCoreRatioLimit

UINT8 CPU_POWER_MGMT_BASIC_CONFIG::ThreeCoreRatioLimit

3-Core Ratio Limit: LFM to Fused 3-Core Ratio Limit, For overclocking part: LFM to Fused 3-Core Ratio Limit + OC Bins.

Default value is from hardware register MSR 0x1AD. Note: OC Bins = 7 means fully unlocked, so range is LFM to 83.

- This 3-Core Ratio Limit Must be Less than or equal to 1-Core Ratio Limit.
- This policy is for platforms only support MSR 0x1AD without MSR 0x1AE support.

Definition at line 148 of file CpuPowerMgmtBasicConfig.h.

13.33.2.30 TurboRatioLimitNumCore

UINT8 CPU_POWER_MGMT_BASIC_CONFIG::TurboRatioLimitNumCore[TURBO_RATIO_LIMIT_ARRAY_SIZE]

Turbo Ratio Limit Num Core array, TurboRatioLimitNumCore[7-0] will pair with TurboRatioLimitRatio[7-0] to determine the active core ranges for each frequency point.

Default value is from hardware register MSR 0x1AE. Note:

- These policies are for platforms support MSR pair 0x1AD and 0x1AE.

Definition at line 302 of file CpuPowerMgmtBasicConfig.h.

13.33.2.31 TurboRatioLimitRatio

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::TurboRatioLimitRatio[TURBO_RATIO_LIMIT_ARRAY_SIZE]
```

Turbo Ratio Limit Ratio array, TurboRatioLimitRatio[7-0] will pair with TurboRatioLimitNumCore[7-0] to determine the active core ranges for each frequency point.

Default value is from hardware register MSR 0x1AD. Note:

- These policies are for platforms support MSR pair 0x1AD and 0x1AE.

Definition at line 294 of file CpuPowerMgmtBasicConfig.h.

13.33.2.32 TwoCoreRatioLimit

```
UINT8 CPU_POWER_MGMT_BASIC_CONFIG::TwoCoreRatioLimit
```

2-Core Ratio Limit: LFM to Fused 2-Core Ratio Limit, For overclocking part: LFM to Fused 2-Core Ratio Limit + OC Bins.

Default value is from hardware register MSR 0x1AD. Note: OC Bins = 7 means fully unlocked, so range is LFM to 83.

- This 2-Core Ratio Limit Must be Less than or equal to 1-Core Ratio Limit.
- This policy is for platforms only support MSR 0x1AD without MSR 0x1AE support.

Definition at line 140 of file CpuPowerMgmtBasicConfig.h.

The documentation for this struct was generated from the following file:

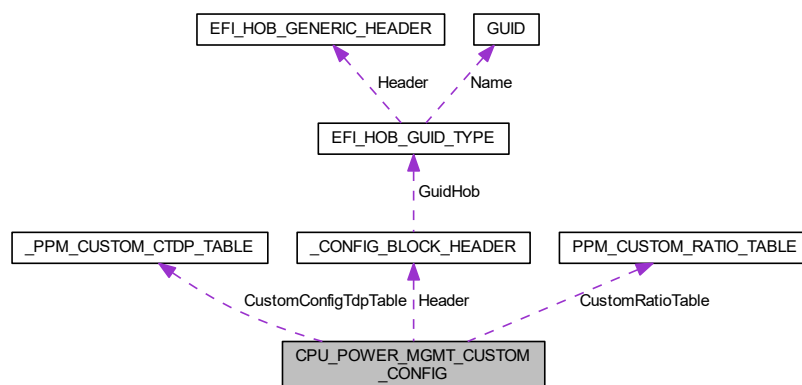
- [CpuPowerMgmtBasicConfig.h](#)

13.34 CPU_POWER_MGMT_CUSTOM_CONFIG Struct Reference

CPU Power Management Custom Configuration Structure.

```
#include <CpuPowerMgmtCustomConfig.h>
```

Collaboration diagram for CPU_POWER_MGMT_CUSTOM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [PPM_CUSTOM_RATIO_TABLE](#) CustomRatioTable
Custom Processor Ratio Table Instance.
- [PPM_CUSTOM_CTDp_TABLE](#) CustomConfigTdpTable [MAX_CUSTOM_CTDp_ENTRIES]
Custom ConfigTdp Settings Instance.
- UINT32 [ConfigTdpLock](#): 1
*Lock the ConfigTdp mode settings from runtime changes; 0: **Disable**; 1: Enable.*
- UINT32 [ConfigTdpBios](#): 1
*Configure whether to load Configurable TDP SSDT; 0: **Disable**; 1: Enable.*
- UINT32 [RsvdBits](#): 30
Reserved for future use.

13.34.1 Detailed Description

CPU Power Management Custom Configuration Structure.

Revision 1:

- Initial version.

Definition at line 93 of file CpuPowerMgmtCustomConfig.h.

The documentation for this struct was generated from the following file:

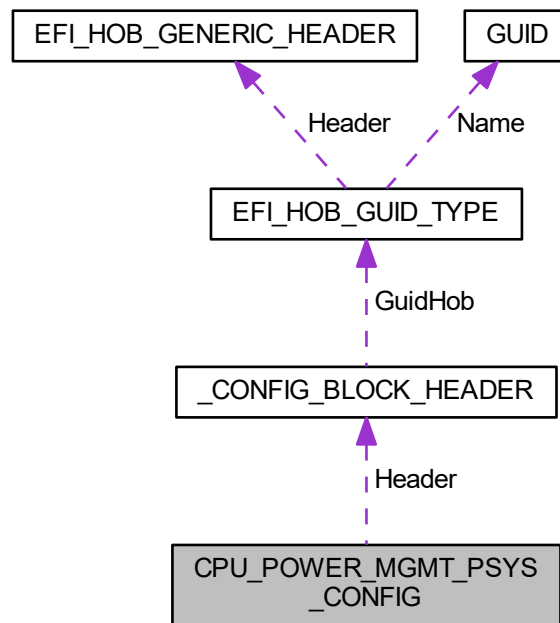
- [CpuPowerMgmtCustomConfig.h](#)

13.35 CPU_POWER_MGMT_PSYS_CONFIG Struct Reference

CPU Power Management Psys(Platform) Configuration Structure.

```
#include <CpuPowerMgmtPsysConfig.h>
```

Collaboration diagram for CPU_POWER_MGMT_PSYS_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- UINT32 [PsysPowerLimit1](#): 1
MSR 0x65C[15]: PL1 Enable activates the PL1 value to limit average platform power.
- UINT32 [PsysPowerLimit1Time](#): 8
MSR 0x65C[23:17]: PL1 timewindow in seconds.
- UINT32 [PsysPowerLimit2](#): 1
MSR 0x65C[47]: PL2 Enable activates the PL2 value to limit average platform power.
- UINT32 [RsvdBits](#): 22
Reserved for future use.
- UINT16 [PsysPowerLimit1Power](#)
MSR 0x65C[14:0]: Platform PL1 power. Units are based on `POWER_MGMT_CONFIG.CustomPowerUnit`.
- UINT16 [PsysPowerLimit2Power](#)
MSR 0x65C[46:32]: Platform PL2 power. Units are based on `POWER_MGMT_CONFIG.CustomPowerUnit`.
- UINT16 [PsysPmax](#)
*PCODE MMIO Mailbox: Platform Power Pmax. **0 - Auto** Specified in 1/8 Watt increments. 0-1024 Watts. Value of 800 = 100W.*
- UINT8 [Rsvd](#) [2]
Reserved for future use and config block alignment.

13.35.1 Detailed Description

CPU Power Management Psys(Platform) Configuration Structure.

Revision 1:

- Initial version.

Definition at line 50 of file CpuPowerMgmtPsysConfig.h.

The documentation for this struct was generated from the following file:

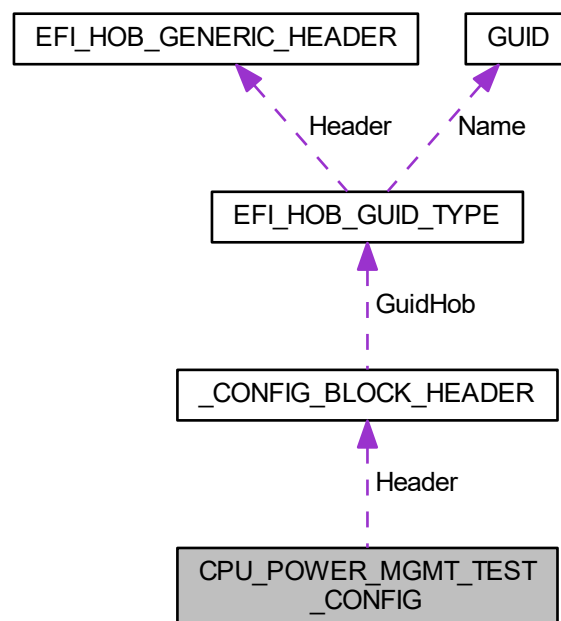
- [CpuPowerMgmtPsysConfig.h](#)

13.36 CPU_POWER_MGMT_TEST_CONFIG Struct Reference

CPU Power Management Test Configuration Structure.

```
#include <CpuPowerMgmtTestConfig.h>
```

Collaboration diagram for CPU_POWER_MGMT_TEST_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.
- UINT32 [Eist](#): 1
*Offset 28-31 Enable or Disable Intel SpeedStep Technology. 0: Disable; 1: **Enable***
- UINT32 [EnergyEfficientPState](#): 1
*Enable or Disable Energy Efficient P-state will be applied in Turbo mode. Disable; 1: **Enable***
- UINT32 [EnergyEfficientTurbo](#): 1
*Enable or Disable Energy Efficient Turbo, will be applied in Turbo mode. 0: **Disable**; 1: Enable.*
- UINT32 [TStates](#): 1
*Enable or Disable T states; 0: **Disable**; 1: Enable.*
- UINT32 [BiProcHot](#): 1
*Enable or Disable Bi-Directional PROCHOT#; 0: Disable; 1: **Enable**.*
- UINT32 [DisableProcHotOut](#): 1
*Enable or Disable PROCHOT# signal being driven externally; 0: Disable; 1: **Enable**.*
- UINT32 [ProcHotResponse](#): 1
*Enable or Disable PROCHOT# Response; 0: Disable; 1: **Enable**.*
- UINT32 [DisableVrThermalAlert](#): 1
*Enable or Disable VR Thermal Alert; 0: **Disable**; 1: Enable.*
- UINT32 [EnableAllThermalFunctions](#): 1
*Enable or Disable Thermal Reporting through ACPI tables; 0: Disable; 1: **Enable**.*
- UINT32 [ThermalMonitor](#): 1
*Enable or Disable Thermal Monitor; 0: Disable; 1: **Enable**.*
- UINT32 [Cx](#): 1
*Enable or Disable CPU power states (C-states). 0: Disable; 1: **Enable***
- UINT32 [PmgCstCfgCtrlLock](#): 1
*If enabled, sets MSR 0xE2[15]; 0: Disable; 1: **Enable**.*
- UINT32 [C1e](#): 1
*Enable or Disable Enhanced C-states. 0: Disable; 1: **Enable***
- UINT32 [C1AutoDemotion](#): 1
*Enable or Disable C6/C7 auto demotion to C1. 0: Disabled; 1: **C1 Auto demotion***
- UINT32 [C1UnDemotion](#): 1
*Enable or Disable C1UnDemotion. 0: Disabled; 1: **C1 Auto undemotion***
- UINT32 [C3AutoDemotion](#): 1
- UINT32 [C3UnDemotion](#): 1
- UINT32 [PkgCStateDemotion](#): 1
*Enable or Disable Package Cstate Demotion. Disable; 1: **Enable** [WhiskeyLake] **Disable**; 1: Enable.*
- UINT32 [PkgCStateUnDemotion](#): 1
*Enable or Disable Package Cstate UnDemotion. Disable; 1: **Enable** [WhiskeyLake] **Disable**; 1: Enable.*
- UINT32 [CStatePreWake](#): 1
*Enable or Disable CState-Pre wake. Disable; 1: **Enable***
- UINT32 [TimedMwait](#): 1
*Enable or Disable TimedMwait Support. **Disable**; 1: Enable.*
- UINT32 [CstCfgCtrlMwaitRedirection](#): 1
*Enable or Disable IO to MWAIT redirection; 0: **Disable**; 1: Enable.*
- UINT32 [ProcHotLock](#): 1
*If enabled, sets MSR 0x1FC[23]; 0: Disable; 1: **Enable**.*
- UINT32 [RaceToHalt](#): 1
*Enable or Disable Race To Halt feature; 0: Disable; 1: **Enable** . RTH will dynamically increase CPU frequency in order to enter pkg C-State faster to reduce overall power. (RTH is controlled through MSR 1FC bit 20)*

- UINT32 [ConfigTdpLevel](#): 8
- UINT16 [CstateLatencyControl1Irtl](#)
Offset 32-33 Interrupt Response Time Limit of LatencyControl1 MSR 0x60B[9:0].0 is Auto.
- UINT16 [CstateLatencyControl2Irtl](#)
Offset 34-35 Interrupt Response Time Limit of LatencyControl2 MSR 0x60C[9:0].0 is Auto.
- UINT16 [CstateLatencyControl3Irtl](#)
Offset 36-37 Interrupt Response Time Limit of LatencyControl3 MSR 0x633[9:0].0 is Auto.
- UINT16 [CstateLatencyControl4Irtl](#)
Offset 38-39 Interrupt Response Time Limit of LatencyControl4 MSR 0x634[9:0].0 is Auto.
- UINT16 [CstateLatencyControl5Irtl](#)
Offset 40-41 Interrupt Response Time Limit of LatencyControl5 MSR 0x635[9:0].0 is Auto.
- UINT16 [CstateLatencyControl0Irtl](#)
Offset 42-43 Interrupt Response Time Limit of LatencyControl0 MSR 0x60A[9:0].0 is Auto.
- [MAX_PKG_C_STATE](#) [PkgCStateLimit](#)
Offset 44 This field is used to set the Max Pkg Cstate. Default set to Auto which limits the Max Pkg Cstate to deep C-state.
- [C_STATE_TIME_UNIT](#) [CstateLatencyControl0TimeUnit](#)
Offset 45 TimeUnit for Latency Control0 MSR 0x60A[12:10]; 2: 1024ns.
- [C_STATE_TIME_UNIT](#) [CstateLatencyControl1TimeUnit](#)
Offset 46 TimeUnit for Latency Control1 MSR 0x60B[12:10]; 2: 1024ns.
- [C_STATE_TIME_UNIT](#) [CstateLatencyControl2TimeUnit](#)
Offset 47 TimeUnit for Latency Control2 MSR 0x60C[12:10]; 2: 1024ns.
- [C_STATE_TIME_UNIT](#) [CstateLatencyControl3TimeUnit](#)
Offset 48 TimeUnit for Latency Control3 MSR 0x633[12:10]; 2: 1024ns.
- [C_STATE_TIME_UNIT](#) [CstateLatencyControl4TimeUnit](#)
Offset 49 TimeUnit for Latency Control4 MSR 0x634[12:10]; 2: 1024ns.
- [C_STATE_TIME_UNIT](#) [CstateLatencyControl5TimeUnit](#)
Offset 50 TimeUnit for Latency Control5 MSR 0x635[12:10]; 2: 1024ns.
- [CUSTOM_POWER_UNIT](#) [CustomPowerUnit](#)
Offset 51 Default power unit in watts or in 125 milliwatt increments.
- [PPM_IRM_SETTING](#) [PpmIrmSetting](#)
Offset 52 Interrupt Redirection Mode Select.
- UINT32 [ProcHotDemotion](#): 1
Offset 53-56 ProcHot Demotion Algorithm configuration; 0: Disable; 1: Hardware Default
- UINT32 [TurboConfiguration](#): 1
Change the PL2 and TAU;Default: 0: Max Transient Turbo 1: 1.2 X Turbo.
- UINT32 [RsvdBits](#): 30
RsvdBits.

13.36.1 Detailed Description

CPU Power Management Test Configuration Structure.

Revision 1:

- Initial version. **Revision 2:**
- Add [CstateLatencyControl0TimeUnit](#) for WHL only
- Add [CstateLatencyControl0Irtl](#) for WHL only **Revision 3:**

- Change C State LatencyControl to Auto as default. **Revision 4:**
- Add CstateLatencyControl0TimeUnit and CstateLatencyControl0Irtl. **Revision 5:**
- Change EnergyEfficientTurbo to Disable as default. **Revision 6:**
- Change ProcHotResponse and ProcHotLock to Enable as default. **Revision 7:**
- Deprecate ConfigTdpLevel. Move to premem. **Revision 8:**
- Add ProcHotDemotion. **Revision 9:**
- Add TurboConfiguration.

Definition at line 121 of file CpuPowerMgmtTestConfig.h.

13.36.2 Member Data Documentation

13.36.2.1 C3AutoDemotion

UINT32 CPU_POWER_MGMT_TEST_CONFIG::C3AutoDemotion

Deprecated no longer used

Definition at line 138 of file CpuPowerMgmtTestConfig.h.

13.36.2.2 C3UnDemotion

UINT32 CPU_POWER_MGMT_TEST_CONFIG::C3UnDemotion

Deprecated no longer used

Definition at line 139 of file CpuPowerMgmtTestConfig.h.

13.36.2.3 ConfigTdpLevel

UINT32 CPU_POWER_MGMT_TEST_CONFIG::ConfigTdpLevel

Deprecated Move to premem phase.

Definition at line 147 of file CpuPowerMgmtTestConfig.h.

13.36.2.4 CstateLatencyControl0TimeUnit

`C_STATE_TIME_UNIT` CPU_POWER_MGMT_TEST_CONFIG::CstateLatencyControl0TimeUnit

Offset 45 TimeUnit for Latency Control0 MSR 0x60A[12:10]; 2: **1024ns**.

Todo : The following enums have to be replaced with policies.

Definition at line 158 of file CpuPowerMgmtTestConfig.h.

13.36.2.5 CustomPowerUnit

`CUSTOM_POWER_UNIT` CPU_POWER_MGMT_TEST_CONFIG::CustomPowerUnit

Offset 51 Default power unit in watts or in 125 milliwatt increments.

- 0: PowerUnitWatts.
- **1: PowerUnit125MilliWatts.**

Definition at line 169 of file CpuPowerMgmtTestConfig.h.

13.36.2.6 PpmIrmSetting

`PPM_IRM_SETTING` CPU_POWER_MGMT_TEST_CONFIG::PpmIrmSetting

Offset 52 Interrupt Redirection Mode Select.

- 0: Fixed priority. //Default under CNL.
- 1: Round robin.
- 2: Hash vector.
- 4: PAIR with fixed priority. //Default under KBL, not available under CNL.
- 5: PAIR with round robin. //Not available under CNL.
- 6: PAIR with hash vector. //Not available under CNL.
- 7: No change.

Definition at line 180 of file CpuPowerMgmtTestConfig.h.

The documentation for this struct was generated from the following file:

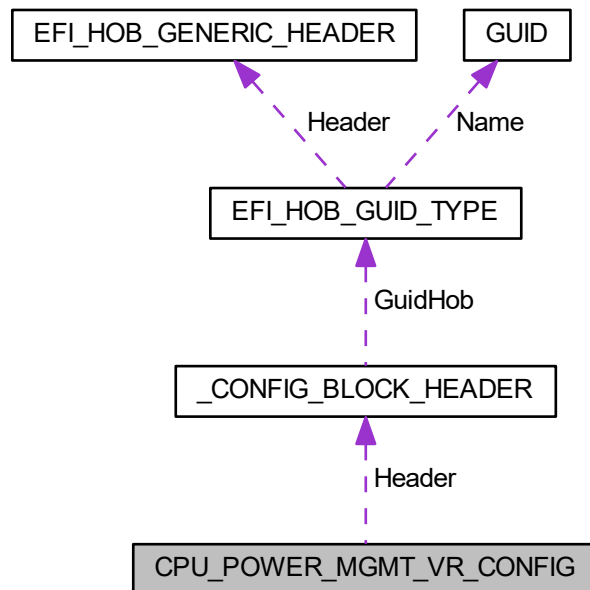
- [CpuPowerMgmtTestConfig.h](#)

13.37 CPU_POWER_MGMT_VR_CONFIG Struct Reference

CPU Power Management VR Configuration Structure.

```
#include <CpuPowerMgmtVrConfig.h>
```

Collaboration diagram for CPU_POWER_MGMT_VR_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [AcousticNoiseMitigation](#): 1
Enable or Disable Acoustic Noise Mitigation feature. 0: Disabled; 1: Enabled.
- UINT32 [SendVrMbxCmd](#): 2
Deprecated.
- UINT32 [EnableMinVoltageOverride](#): 1
Enable or disable Minimum Voltage override for minimum voltage runtime and minimum voltage C8. 0: Disabled 1: Enabled.
- UINT32 [RfiMitigation](#): 1
Enable or Disable RFI Mitigation. 0: Disable - DCM is the IO_N default; 1: Enable - Enable IO_N DCM/CCM switching as RFI mitigation.
- UINT32 [RsvdBits](#): 27
Reserved for future use.
- UINT8 [PsysSlope](#)
PCODE MMIO Mailbox: Platform Psys slope correction. 0: Auto Specified in 1/100 increment values. Range is 0-200. 125 = 1.25.

- UINT8 [PsysOffset](#)
PCODE MMIO Mailbox: Platform Psys offset correction. **0: Auto** Units 1/4, Range 0-255. Value of 100 = 100/4 = 25 offset. Deprecated.
- UINT8 [FivrSpreadSpectrum](#)
PCODE MMIO Mailbox: FIVR RFI Spread Spectrum Range.
- UINT16 [FivrRfiFrequency](#)
PCODE MMIO Mailbox: Set the desired RFI frequency, in increments of 100KHz.
- UINT16 [MinVoltageRuntime](#)
PCODE MMIO Mailbox: Minimum voltage for runtime. Valid if EnableMinVoltageOverride = 1 .Range 0 to 1999mV. **0: 0mV**
- UINT16 [MinVoltageC8](#)
PCODE MMIO Mailbox: Minimum voltage for C8. Valid if EnableMinVoltageOverride = 1. Range 0 to 1999mV. **0: 0mV**
- UINT16 [PsysOffset1](#)
PCODE MMIO Mailbox: Platform Psys offset correction. **0: Auto** Units 1/1000, Range 0-63999. For an offset of 25.348, enter 25348.
- UINT32 [TdcTimeWindow1](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: Thermal Design Current time window. Defined in milli seconds. **1ms default**
- UINT8 [Irms](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: Current root mean square. **0: Disable**; 1: Enable.
- UINT8 [FivrSpectrumEnable](#)
Enable or Disable FIVR Spread Spectrum 0: Disable; **1: Enable**.
- UINT8 [DlvrSpreadSpectrumPercentage](#)
PCODE MMIO Mailbox: Encoded DLVR SSC value. 0x0 = Disabled. u3.2 value from 0% - 7.75%.
- UINT8 [DlvrRfiEnable](#)
PCODE MMIO Mailbox: Enable/Disable RFI frequency hopping. **0: Disable**. 1:Disable.
- UINT8 [PreWake](#)
PCODE MMIO Mailbox: Acoustic Noise Mitigation Range. This can be programmed only if AcousticNoiseMitigation is enabled.**Default Value = 0 micro ticks** Defines the max pre-wake randomization time in micro ticks. Range is 0-255.
- UINT8 [RampUp](#)
PCODE MMIO Mailbox: Acoustic Noise Mitigation Range. This can be programmed only if AcousticNoiseMitigation is enabled.**Default Value = 0 micro ticks** Defines the max ramp up randomization time in micro ticks. Range is 0-255.
- UINT8 [RampDown](#)
PCODE MMIO Mailbox: Acoustic Noise Mitigation Range. This can be programmed only if AcousticNoiseMitigation is enabled.**Default Value = 0 micro ticks** Defines the max ramp down randomization time in micro ticks. Range is 0-255.
- UINT16 [VccInAuxImonIccImax](#)
PCODE MMIO Mailbox: VccIn Aux Imon IccMax. **0 - Auto** Values are in 1/4 Amp increments. Range is 0-128A8.
- UINT16 [VccInAuxImonSlope](#)
PCODE MMIO Mailbox: VccInAuxImon slope correction. Specified in 1/100 increment values. Range is 0-200. 125 = 1.25. **0: Auto**
- INT16 [VccInAuxImonOffset](#)
PCODE MMIO Mailbox: VccInAuxImon offset correction. Value is a 2's complement signed integer. Units 1/1000, Range 0-63999. For an offset = 12.580, use 12580. **0: Auto**
- UINT16 [DlvrRfiFrequency](#)
PCODE MMIO Mailbox: DLVR clock frequency in MHz. Default is 0x55A MHz.
- UINT8 [PS1toPS0DynamicCutoffEnable](#) [MAX_NUM_VRS]
Enable/Disable PS1 to PS0 Dynamic Cutoff; **0: Disable**; 1: Enable.
- UINT8 [PS2toPS1DynamicCutoffEnable](#) [MAX_NUM_VRS]
Enable/Disable:PS2 to PS1 Dynamic Cutoff; **0: Disable**; 1: Enable.
- UINT16 [PS1toPS0MCoef](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: PS1 to PS0 M Coef. This number is in M*100 units. **Default: 100** Range: 0-4096.
- INT16 [PS1toPS0CCoef](#) [MAX_NUM_VRS]

- PCODE MMIO Mailbox: PS1 to PS0 C Coef. This number is in C*100 units. **Default: 2000** Range: 0-4096.
- UINT16 [PS2toPS1MCoef](#) [[MAX_NUM_VRS](#)]
 - PCODE MMIO Mailbox: PS2 to PS1 M Coef. This number is in M*100 units. **Default: 100** Range: 0-4096.
- INT16 [PS2toPS1CCoef](#) [[MAX_NUM_VRS](#)]
 - PCODE MMIO Mailbox: PS2 to PS1 C Coef. This number is in C*100 units. **Default: 500** Range: 0-4096.
- UINT8 [VrPowerDeliveryDesign](#)
 - This field is required to be non-zero on Desktop platforms.
- UINT8 [VccInDemotionEnable](#) [[MAX_NUM_VRS](#)]
 - Enable VCC Demotion.
- UINT8 [VccInDemotionQuiescentPowerInMw](#) [[MAX_NUM_VRS](#)]
 - User configured platform quiescent threshold in milli-watt when VCC Demotion is enabled with user configured threshold.
- UINT8 [Rsvd5](#) [2]
 - Make next field 4-byte aligned.
- UINT32 [VccInDemotionCapacitanceInUf](#) [[MAX_NUM_VRS](#)]
 - User configured VR's quiescent power threshold in micro-farad when VCC Demotion is enabled with user configured threshold.
- UINT8 [EnableVsysCritical](#)
 - PCODE MMIO Mailbox: Enable/Disable Vsys Critical; **0: Disable**; 1: Enable.
- UINT8 [VsysFullScale](#)
 - Set VSYS Critical [0x4A] Write to 0x4A to program Vsys Critical Threshold which is a linear fractional value of the Vsys input full scale voltage where minimum is 0 and maximum is 255.
- UINT8 [VsysCriticalThreshold](#)
 - Deprecated from version 18.
- UINT8 [VsysAssertionDeglitchMantissa](#)
 - Set Assertion Deglitch [0x4F] and De-Assertion Deglitch [0x49] Provide 2 fields in BIOS Menu for Assertion/De-Assertion Deglitch: Mantissa and Exponent.
- UINT16 [IccLimit](#) [[MAX_NUM_VRS](#)]
 - PCODE VR Mailbox: Voltage Regulator Fast Vmode ICC Limit.
- UINT8 [EnableFastVmode](#) [[MAX_NUM_VRS](#)]
 - VR Fast Vmode.
- UINT32 [VsysFullScale1](#)
 - Set Vsys/Psys Critical [0x4A] Write to 0x4A to program Vsys Critical Threshold which is a linear fractional value of the Vsys input full scale voltage where minimum is 0 and maximum is 255000mv For example: Vsys_Crit (0x4A) = (Critical Threshold/Full Scale) * 0xFF.

VR Settings

The VR related settings are sorted in an array where each index maps to the VR domain as defined below:

- 0 = System Agent VR
- 1 = IA Core VR
- 2 = Ring Vr
- 3 = GT VR
- 4 = FIVR VR

The VR settings for a given domain must be populated in the appropriate index.

- UINT16 [TdcCurrentLimit](#) [[MAX_NUM_VRS](#)]
 - PCODE MMIO Mailbox: Thermal Design Current current limit. Specified in 1/8A units. Range is 0-4095. 1000 = 125A. **0: 0 Amps**
- UINT16 [AcLoadline](#) [[MAX_NUM_VRS](#)]
 - PCODE MMIO Mailbox: AcLoadline in 1/100 mOhms (ie. 1250 = 12.50 mOhm); Range is 0-6249. **Intel Recommended Defaults vary by domain and SKU.**

- UINT16 [DcLoadline](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: DcLoadline in 1/100 mOhms (ie. 1250 = 12.50 mOhm); Range is 0-6249. **Intel Recommended Defaults vary by domain and SKU.**
- UINT16 [Psi1Threshold](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: Power State 1 current cutoff in 1/4 Amp increments. Range is 0-128A.
- UINT16 [Psi2Threshold](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: Power State 2 current cutoff in 1/4 Amp increments. Range is 0-128A.
- UINT16 [Psi3Threshold](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: Power State 3 current cutoff in 1/4 Amp increments. Range is 0-128A.
- INT16 [ImonOffset](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: Imon offset correction. Value is a 2's complement signed integer. Units 1/1000, Range 0-63999. For an offset = 12.580, use 12580. **0: Auto**
- UINT16 [IccMax](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: VR Icc Max limit. 0-512A in 1/4 A units. 400 = 100A. **Default: 0 - Auto, no override**
- UINT16 [VrVoltageLimit](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: VR Voltage Limit. Range is 0-7999mV.
- UINT16 [ImonSlope](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: Imon slope correction. Specified in 1/100 increment values. Range is 0-200. 125 = 1.25. **0: Auto**
- UINT8 [Psi3Enable](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: Power State 3 enable/disable; 0: Disable; **1: Enable.**
- UINT8 [Psi4Enable](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: Power State 4 enable/disable; 0: Disable; **1: Enable.**
- UINT8 [VrConfigEnable](#) [MAX_NUM_VRS]
Enable/Disable BIOS configuration of VR; 0: Disable; **1: Enable.**
- UINT8 [TdcEnable](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: Thermal Design Current enable/disable; **0: Disable**; 1: Enable.
- UINT8 [TdcTimeWindow](#) [MAX_NUM_VRS]
- UINT8 [TdcLock](#) [MAX_NUM_VRS]
PCODE MMIO Mailbox: Thermal Design Current Lock; **0: Disable**; 1: Enable.
- UINT8 [FastPkgCRampDisable](#) [MAX_NUM_VRS]
Disable Fast Slew Rate for Deep Package C States for VR IA,GT,SA,VLCC,FIVR domain based on Acoustic Noise Mitigation feature enabled. **0: False**; 1: True.
- UINT8 [SlowSlewRate](#) [MAX_NUM_VRS]
Slew Rate configuration for Deep Package C States for VR VR IA,GT,SA,VLCC,FIVR domain based on Acoustic Noise Mitigation feature enabled. **0: Fast/2**; 1: Fast/4; 2: Fast/8; 3: Fast/16.

13.37.1 Detailed Description

CPU Power Management VR Configuration Structure.

Revision 1:

- Initial version. **Revision 2:**
- Updated Acoustic Noise Mitigation. **Revision 3:**
- Deprecate PsysOffset and added PsysOffset1 for Psys Offset Correction **Revision 4:**
- Deprecate TdcTimeWindow and added TdcTimeWindow1 for TDC Time Added Irms support. **Revision 5:**
- Add RfiMitigation. **Revision 6:**
- Add Dynamic Periodicity Alteration (DPA) tuning feature **Revision 7:**
- Add Vccln Aux Imon feature **Revision 8:**
- Deprecate SendVrMbxCmd **Revision 9:**
- Added an option to Enable/Disable FIVR Spread Spectrum **Revision 10:**

- Add DlvrRfiEnable, DlvrSpreadSpectrumPercentage and DlvrRfiFrequency for RFI control. **Revision 11:**
- Added an option to Enable/Disable PS1 to PS0/PS2 to PS1 dynamic cutoff **Revision 12:**
- Added an option for Vr Power Delivery design **Revision 13:**
- Add VccInDemotionEnable, VccInDemotionQuiescentPowerInMw and VccInDemotionCapacitanceInUf **Revision 14:**
- Add EnableVsysCritical, VsysFullScale, VsysCriticalThreshold, VsysAssertionDeglitchMantissa
- VsysAssertionDeglitchExponent, VsysDeassertionDeglitchMantissa, VsysDeassertionDeglitchExponent **Revision 15:**
- Add Voltage Regulator Fast Vmode ICC Limit **Revision 16:**
- Modified lccMax range up to 512A **Revision 17:**
- Add EnableFastVmode by VR domain. **Revision 18:**
- Deprecate VsysFullScale, VsysCriticalThreshold
- Add VsysFullScale1, VsysCriticalThreshold1, PsysFullScale, PsysCriticalThreshold

Definition at line 93 of file CpuPowerMgmtVrConfig.h.

13.37.2 Member Data Documentation

13.37.2.1 EnableFastVmode

```
UINT8 CPU_POWER_MGMT_VR_CONFIG::EnableFastVmode [MAX_NUM_VRS]
```

VR Fast Vmode.

Use to control Fast Vmode Enable/Disable by VR domain. **Default: 0 - Disable, no override**

Definition at line 239 of file CpuPowerMgmtVrConfig.h.

13.37.2.2 FivrRfiFrequency

```
UINT16 CPU_POWER_MGMT_VR_CONFIG::FivrRfiFrequency
```

PCODE MMIO Mailbox: Set the desired RFI frequency, in increments of 100KHz.

0: Auto Range varies based on XTAL clock:

- 0-1918 (Up to 191.8MHz) for 24MHz clock.
- 0-1535 (Up to 153.5MHz) for 19MHz clock.

Definition at line 119 of file CpuPowerMgmtVrConfig.h.

13.37.2.3 FivrSpreadSpectrum

UINT8 CPU_POWER_MGMT_VR_CONFIG::FivrSpreadSpectrum

PCODE MMIO Mailbox: FIVR RFI Spread Spectrum Range.

1.5% Range: 0.5%, 1%, 1.5%, 2%, 3%, 4%, 5%, 6%. Each Range is translated to an encoded value for FIVR register. 0.5% = 0, 1% = 3, 1.5% = 8, 2% = 18, 3% = 28, 4% = 34, 5% = 39, 6% = 44.

Definition at line 110 of file CpuPowerMgmtVrConfig.h.

13.37.2.4 IccLimit

UINT16 CPU_POWER_MGMT_VR_CONFIG::IccLimit [MAX_NUM_VRS]

PCODE VR Mailbox: Voltage Regulator Fast Vmode ICC Limit.

This value represents the current threshold where the VR would initiate protection if Fast Vmode is enabled. A value of 0 corresponds to feature disabled (no reactive protection). **Default: 0 - Disable, no override**

Definition at line 232 of file CpuPowerMgmtVrConfig.h.

13.37.2.5 TdcTimeWindow

UINT8 CPU_POWER_MGMT_VR_CONFIG::TdcTimeWindow [MAX_NUM_VRS]

Deprecated . PCODE MMIO Mailbox: Thermal Design Current time window. Defined in milli seconds. **1ms default**

Definition at line 146 of file CpuPowerMgmtVrConfig.h.

13.37.2.6 VccInDemotionCapacitanceInUf

UINT32 CPU_POWER_MGMT_VR_CONFIG::VccInDemotionCapacitanceInUf [MAX_NUM_VRS]

User configured VR's quiescent power threshold in micro-farad when VCC Demotion is enabled with user configured threshold.

Range: 0-2000.

Definition at line 205 of file CpuPowerMgmtVrConfig.h.

13.37.2.7 VccInDemotionEnable

```
UINT8 CPU_POWER_MGMT_VR_CONFIG::VccInDemotionEnable [MAX_NUM_VRS]
```

Enable VCC Demotion.

0: Disable; **1: Enable with default threshold**; 2: Enable with user configured threshold.

Definition at line 192 of file CpuPowerMgmtVrConfig.h.

13.37.2.8 VccInDemotionQuiescentPowerInMw

```
UINT8 CPU_POWER_MGMT_VR_CONFIG::VccInDemotionQuiescentPowerInMw [MAX_NUM_VRS]
```

User configured platform quiescent threshold in milli-watt when VCC Demotion is enabled with user configured threshold.

Range: 0-255.

Definition at line 198 of file CpuPowerMgmtVrConfig.h.

13.37.2.9 VrPowerDeliveryDesign

```
UINT8 CPU_POWER_MGMT_VR_CONFIG::VrPowerDeliveryDesign
```

This field is required to be non-zero on Desktop platforms.

Used to communicate the power delivery design capability of the board. This value is an enum of the available power delivery segments that are defined in the Platform Design Guide. This is required due to CPU socket compatibility across different board designs. Without this field populated, the system may experience VR OCP/OVP shutdowns, hangs, thermal and performance loss. **0 - Disable**

Definition at line 184 of file CpuPowerMgmtVrConfig.h.

13.37.2.10 VsysAssertionDeglitchMantissa

```
UINT8 CPU_POWER_MGMT_VR_CONFIG::VsysAssertionDeglitchMantissa
```

Set Assertion Deglitch [0x4F] and De-Assertion Deglitch [0x49] Provide 2 fields in BIOS Menu for Assertion/De-Assertion Deglitch: Mantissa and Exponent.

The format for Assertion/De-Assertion Deglitch: $0.002\text{ms} * \text{Mantissa} * 2^{\text{Exponent}}$.

Definition at line 221 of file CpuPowerMgmtVrConfig.h.

13.37.2.11 VsysFullScale

```
UINT8 CPU_POWER_MGMT_VR_CONFIG::VsysFullScale
```

Set VSYS Critical [0x4A] Write to 0x4A to program Vsys Critical Threshold which is a linear fractional value of the Vsys input full scale voltage where minimum is 0 and maximum is 255.

Default 0 For example: Vsys_Crit (0x4A) = (Critical Threshold/Full Scale) * 0xFF Deprecated from version 18

Definition at line 214 of file CpuPowerMgmtVrConfig.h.

The documentation for this struct was generated from the following file:

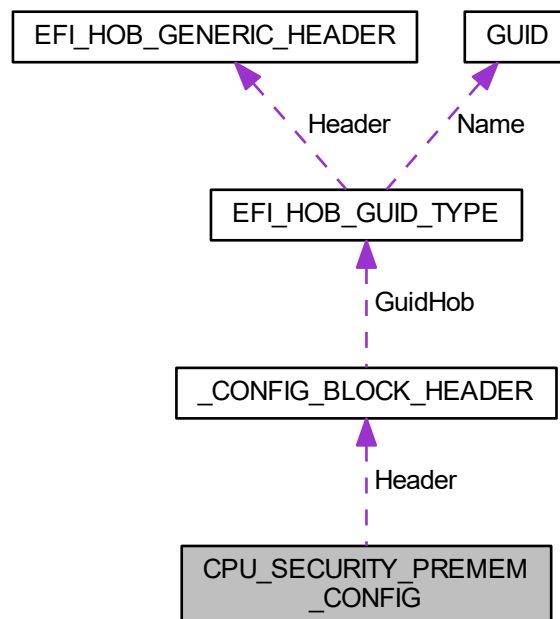
- [CpuPowerMgmtVrConfig.h](#)

13.38 CPU_SECURITY_PREMEM_CONFIG Struct Reference

CPU Security PreMemory Configuration Structure.

```
#include <CpuSecurityPreMemConfig.h>
```

Collaboration diagram for CPU_SECURITY_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [PrmrrSize](#)
*PRMRR Size. **Software Control:** 0x0 32MB: 0x2000000, 64MB: 0x4000000, 128 MB: 0x8000000, 256 MB: 0x10000000, 512 MB: 0x20000000.*
- UINT16 [BiosSize](#)
Flash information for BIOS Guard: BIOS Size in KB.
- UINT8 [Reserved](#) [2]
Reserved for future use.
- UINT32 [BiosGuard](#): 1
*Enable or Disable BIOS Guard; 0: Disable; 1: **Enable**.*
- UINT32 [BiosGuardToolsInterface](#): 1
*BIOS Guard Tools Interface; 0: **Disable**, 1: Enable.*
- UINT32 [EnableSgx](#): 1
*Enable or Disable Software Guard Extensions; 0: **Disable**; 1: Enable.*
- UINT32 [Txt](#): 1
*Enable or Disable Trusted Execution Technology; 0: **Disable**; 1: Enable.*
- UINT32 [SkipStopPbet](#): 1
*(Test) Skip Stop PBET Timer; 0: **Disable**; 1: Enable.*
- UINT32 [EnableC6Dram](#): 1
(Test) This policy indicates whether or not BIOS should allocate PRMRR memory for C6DRAM power gating feature.
- UINT32 [ResetAux](#): 1
*(Test) Reset Auxiliary content, 0: **Disabled**, 1: Enabled*
- UINT32 [TxtAcheckRequest](#): 1
*(Test) AcheckRequest 0: **Disabled**, 1: Enabled. When Enabled, it will call Acheck regardless of crashcode value*
- UINT32 [RsvdBits](#): 24
Reserved for future use.

13.38.1 Detailed Description

CPU Security PreMemory Configuration Structure.

Revision 1:

- Initial version. **Revision 2:**
- Deprecated SGx

Definition at line 52 of file CpuSecurityPreMemConfig.h.

13.38.2 Member Data Documentation

13.38.2.1 BiosGuard

UINT32 CPU_SECURITY_PREMEM_CONFIG::BiosGuard

Enable or Disable BIOS Guard; 0: Disable; **1: Enable**.

- This is an optional feature and can be opted out.
- If this policy is set to Disabled, the policies in the [BIOS_GUARD_CONFIG](#) will be ignored.
- If PeiBiosGuardLibNull is used, this policy will have no effect.

Definition at line 63 of file CpuSecurityPreMemConfig.h.

13.38.2.2 EnableC6Dram

UINT32 CPU_SECURITY_PREMEM_CONFIG::EnableC6Dram

(Test) This policy indicates whether or not BIOS should allocate PRMRR memory for C6DRAM power gating feature.

- 0: Don't allocate any PRMRR memory for C6DRAM power gating feature.
- **1: Allocate PRMRR memory for C6DRAM power gating feature.**

Definition at line 85 of file CpuSecurityPreMemConfig.h.

13.38.2.3 EnableSgx

UINT32 CPU_SECURITY_PREMEM_CONFIG::EnableSgx

Enable or Disable Software Guard Extensions; **0: Disable**; 1: Enable.

- This is an optional feature and can be opted out.
- If this policy is set to Disabled, the policies in the CPU_SGX_CONFIG will be ignored.
- If BaseSoftwareGuardLibNull is used, this policy will have no effect.

Deprecated due to Non-POR feature.

Definition at line 71 of file CpuSecurityPreMemConfig.h.

13.38.2.4 Txt

```
UINT32 CPU_SECURITY_PREMEM_CONFIG::Txt
```

Enable or Disable Trusted Execution Technology; **0: Disable**; 1: Enable.

- This is an optional feature and can be opted out.
- If this policy is set to Disabled, the policies in the [CPU_TXT_PREMEM_CONFIG](#) will be ignored.
- If PeiTxtLibNull is used, this policy will have no effect.

Definition at line 78 of file CpuSecurityPreMemConfig.h.

The documentation for this struct was generated from the following file:

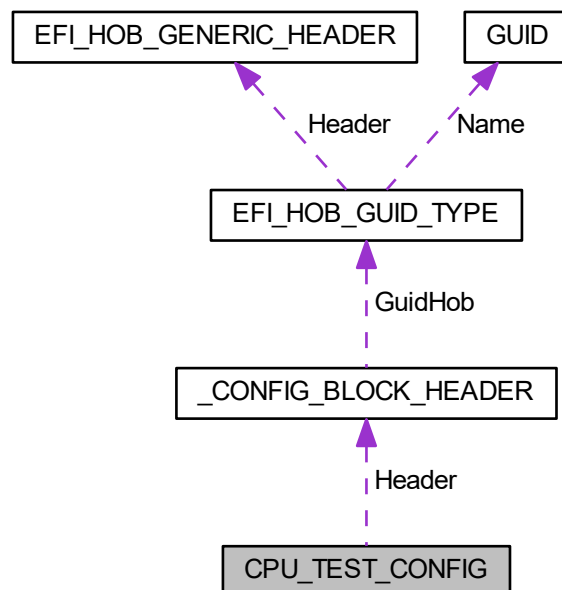
- [CpuSecurityPreMemConfig.h](#)

13.39 CPU_TEST_CONFIG Struct Reference

CPU Test Configuration Structure.

```
#include <CpuTestConfig.h>
```

Collaboration diagram for CPU_TEST_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [MlcStreamerPrefetcher](#): 1
*Enable or Disable MLC Streamer Prefetcher; 0: Disable; 1: **Enable**.*
- UINT32 [MlcSpatialPrefetcher](#): 1
*Enable or Disable MLC Spatial Prefetcher; 0: Disable; 1: **Enable**.*
- UINT32 [MonitorMwaitEnable](#): 1
*Enable or Disable Monitor /MWAIT instructions; 0: Disable; 1: **Enable**.*
- UINT32 [MachineCheckEnable](#): 1
*Enable or Disable initialization of machine check registers; 0: Disable; 1: **Enable**.*
- UINT32 [ProcessorTraceOutputScheme](#): 1
*Control on Processor Trace output scheme; 0: **Single Range Output**; 1: ToPA Output.*
- UINT32 [ProcessorTraceEnable](#): 1
*Enable or Disable Processor Trace feature; 0: **Disable**; 1: Enable.*
- UINT32 [ThreeStrikeCounterDisable](#): 1
*Disable Three strike counter; 0: **FALSE**; 1: TRUE.*
- UINT32 [RsvdBits](#): 25
Reserved for future use.
- UINT32 [ProcessorTraceMemSize](#)
Memory size per thread for Processor Trace.

13.39.1 Detailed Description

CPU Test Configuration Structure.

Revision 1:

- Initial version. **Revision 2:**
- Removed Voltage Optimization feature. **Revision 3:**
- Add PostedInterruptCapability

Definition at line 54 of file CpuTestConfig.h.

13.39.2 Member Data Documentation

13.39.2.1 ProcessorTraceMemSize

```
UINT32 CPU_TEST_CONFIG::ProcessorTraceMemSize
```

Memory size per thread for Processor Trace.

Processor Trace requires 2^N alignment and size in bytes per thread, from 4KB to 128MB.

- **0: Disable**

Definition at line 69 of file CpuTestConfig.h.

The documentation for this struct was generated from the following file:

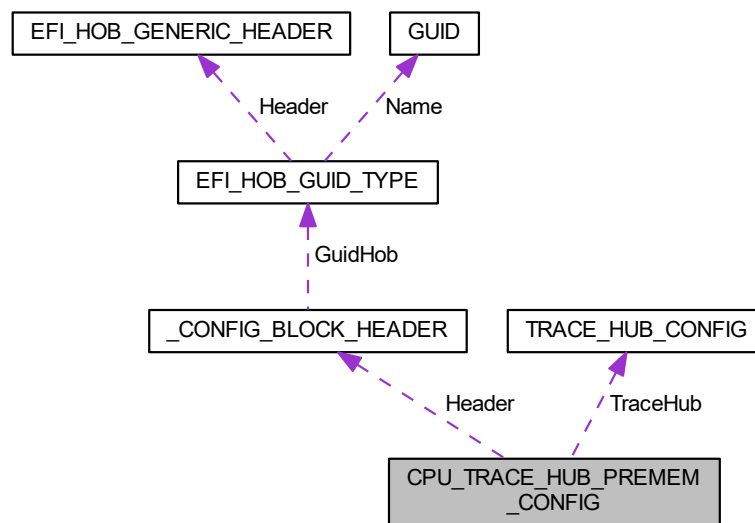
- [CpuTestConfig.h](#)

13.40 CPU_TRACE_HUB_PREMEM_CONFIG Struct Reference

CPU Trace Hub PreMem Configuration Contains Trace Hub settings for CPU side tracing **Revision 1:** - Initial version.

```
#include <TraceHubConfig.h>
```

Collaboration diagram for CPU_TRACE_HUB_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- [TRACE_HUB_CONFIG TraceHub](#)
Trace Hub Config.

13.40.1 Detailed Description

CPU Trace Hub PreMem Configuration Contains Trace Hub settings for CPU side tracing **Revision 1:** - Initial version.

Definition at line 112 of file `TraceHubConfig.h`.

The documentation for this struct was generated from the following file:

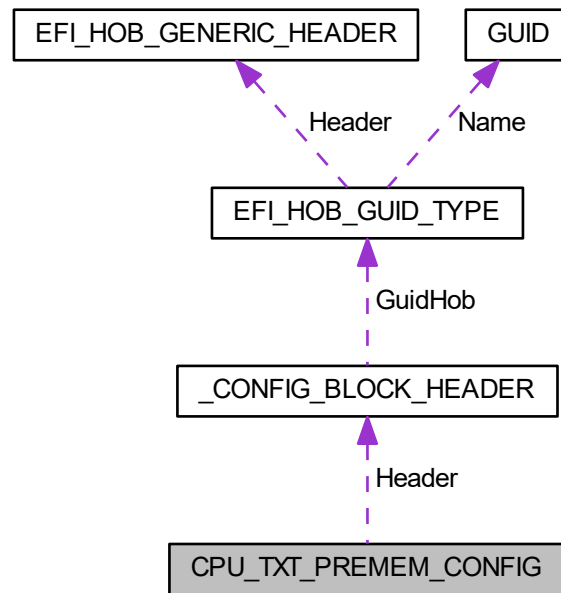
- [TraceHubConfig.h](#)

13.41 CPU_TXT_PREMEM_CONFIG Struct Reference

CPU TXT PreMemory Configuration Structure.

```
#include <CpuTxtConfig.h>
```

Collaboration diagram for CPU_TXT_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [UINT32](#) [SinitMemorySize](#)
Size of SINIT module if installed in flash part. Zero otherwise.
- [UINT32](#) [TxtHeapMemorySize](#)
Size of memory reserved for TXT Heap. This memory is used by MLE.
- [UINT32](#) [TxtDprMemorySize](#)
Size of DPR protected memory reserved for Intel TXT component.
- [EFI_PHYSICAL_ADDRESS](#) [TxtDprMemoryBase](#)
Base address of DPR protected memory reserved for Intel TXT component.
- [UINT64](#) [TxtLcpPdSize](#)
Size of Platform Default Launch Control Policy data if installed in flash part. Zero otherwise.
- [EFI_PHYSICAL_ADDRESS](#) [TxtLcpPdBase](#)
Base address of Platform Default Launch Control Policy data if installed in flash part. Zero otherwise.
- [EFI_PHYSICAL_ADDRESS](#) [ApStartupBase](#)
Base address of TXT AP Startup code.
- [EFI_PHYSICAL_ADDRESS](#) [BiosAcmBase](#)

- *Base address of BIOS ACM in flash part.*
- UINT32 [BiosAcmSize](#)
Size of ACM Binary.
- UINT32 [TgaSize](#)
Size of Trusted Graphics Aperture if supported by chipset.

13.41.1 Detailed Description

CPU TXT PreMemory Configuration Structure.

Note

Optional. These policies will be ignored if [CPU_SECURITY_PREMEM_CONFIG](#) -> Txt is disabled, or [Pei](#)↔
[TxtLibNull](#) is used.

Revision 1:

- Initial version.

Definition at line 51 of file [CpuTxtConfig.h](#).

The documentation for this struct was generated from the following file:

- [CpuTxtConfig.h](#)

13.42 DDI_CONFIGURATION Struct Reference

This structure configures the Native GPIOs for DDI port per VBT settings.

```
#include <GraphicsConfig.h>
```

Public Attributes

- UINT8 [DdiPortBConfig](#)
*The Configuration of DDI port A, this settings must match VBT's settings. **DdiPortDisabled** - No LFP is connected on **DdiPortA**, **DdiPortEdp** - Set **DdiPortA** to **eDP**, **DdiPortMipiDsi** - Set **DdiPortA** to MIPI DSI.*
- UINT8 [DdiPortAHpd](#)
*The Configuration of DDI port B, this settings must match VBT's settings. **DdiPortDisabled** - No LFP is connected on **DdiPortB**, **DdiPortEdp** - Set **DdiPortB** to **eDP**, **DdiPortMipiDsi** - Set **DdiPortB** to MIPI DSI.*
- UINT8 [DdiPortBHpd](#)
*The HPD setting of DDI Port A, this settings must match VBT's settings. **DdiHpdDisable** - Disable HPD, **DdiHpd**↔
Enable - Enable HPD.*
- UINT8 [DdiPortCHpd](#)
*The HPD setting of DDI Port B, this settings must match VBT's settings. **DdiHpdDisable** - Disable HPD, **DdiHpd**↔
Enable - Enable HPD*
- UINT8 [DdiPort1Hpd](#)
*The HPD setting of DDI Port C, this settings must match VBT's settings. **DdiHpdDisable** - Disable HPD, **DdiHpd**↔
Enable - Enable HPD.*

- [UINT8 DdiPort2Hpd](#)
*The HPD setting of DDI Port 1, this settings must match VBT's settings. **DdiHpdDisable - Disable HPD**, **DdiHpdEnable - Enable HPD**.*
- [UINT8 DdiPort3Hpd](#)
*The HPD setting of DDI Port 2, this settings must match VBT's settings. **DdiHpdDisable - Disable HPD**, **DdiHpdEnable - Enable HPD**.*
- [UINT8 DdiPort4Hpd](#)
*The HPD setting of DDI Port 3, this settings must match VBT's settings. **DdiHpdDisable - Disable HPD**, **DdiHpdEnable - Enable HPD**.*
- [UINT8 DdiPortADdc](#)
*The HPD setting of DDI Port 4, this settings must match VBT's settings. **DdiHpdDisable - Disable HPD**, **DdiHpdEnable - Enable HPD**.*
- [UINT8 DdiPortBDdc](#)
*The DDC setting of DDI Port A, this settings must match VBT's settings. **DdiDisable - Disable DDC**, **DdiDdcEnable - Enable DDC**.*
- [UINT8 DdiPortCDdc](#)
*The DDC setting of DDI Port B, this settings must match VBT's settings. **DdiDisable - Disable DDC**, **DdiDdcEnable - Enable DDC**.*
- [UINT8 DdiPort1Ddc](#)
*The DDC setting of DDI Port C, this settings must match VBT's settings. **DdiDisable - Disable DDC**, **DdiDdcEnable - Enable DDC**.*
- [UINT8 DdiPort2Ddc](#)
*The DDC setting of DDI Port 1, this settings must match VBT's settings. **DdiDisable - Disable DDC**, **DdiDdcEnable - Enable DDC**.*
- [UINT8 DdiPort3Ddc](#)
*The DDC setting of DDI Port 2, this settings must match VBT's settings. **DdiDisable - Disable DDC**, **DdiDdcEnable - Enable DDC**.*
- [UINT8 DdiPort4Ddc](#)
*The DDC setting of DDI Port 3, this settings must match VBT's settings. **DdiDisable - Disable DDC**, **DdiDdcEnable - Enable DDC**.*

13.42.1 Detailed Description

This structure configures the Native GPIOs for DDI port per VBT settings.

Definition at line 79 of file GraphicsConfig.h.

The documentation for this struct was generated from the following file:

- [GraphicsConfig.h](#)

13.43 DMI_HW_WIDTH_CONTROL Struct Reference

This structure allows to customize DMI HW Autonomous Width Control for Thermal and Mechanical spec design.

```
#include <ThermalConfig.h>
```

Public Attributes

- UINT32 [DmiTsawEn](#): 1
DMI Thermal Sensor Autonomous Width Enable.
- UINT32 [SuggestedSetting](#): 1
*0: Disable; 1: **Enable** suggested representative values*
- UINT32 [RsvdBits0](#): 6
Reserved bits.
- UINT32 [TS0TW](#): 3
*Thermal Sensor 0 Target Width (**DmiThermSensWidthx8**)*
- UINT32 [TS1TW](#): 3
*Thermal Sensor 1 Target Width (**DmiThermSensWidthx4**)*
- UINT32 [TS2TW](#): 3
*Thermal Sensor 2 Target Width (**DmiThermSensWidthx2**)*
- UINT32 [TS3TW](#): 3
*Thermal Sensor 3 Target Width (**DmiThermSensWidthx1**)*
- UINT32 [RsvdBits1](#): 12
Reserved bits.

13.43.1 Detailed Description

This structure allows to customize DMI HW Autonomous Width Control for Thermal and Mechanical spec design.

When the SuggestedSetting is enabled, the customized values are ignored. Look at DMI_THERMAL_SENSOR_↔TARGET_WIDTH for possible values

Definition at line 89 of file ThermalConfig.h.

The documentation for this struct was generated from the following file:

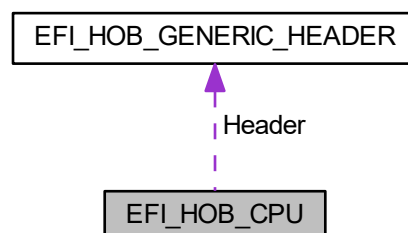
- [ThermalConfig.h](#)

13.44 EFI_HOB_CPU Struct Reference

Describes processor information, such as address space and I/O space capabilities.

```
#include <PiHob.h>
```

Collaboration diagram for EFI_HOB_CPU:



Public Attributes

- [EFI_HOB_GENERIC_HEADER Header](#)
The HOB generic header.
- UINT8 [SizeOfMemorySpace](#)
Identifies the maximum physical memory addressability of the processor.
- UINT8 [SizeOfIoSpace](#)
Identifies the maximum physical I/O addressability of the processor.
- UINT8 [Reserved](#) [6]
This field will always be set to zero.

13.44.1 Detailed Description

Describes processor information, such as address space and I/O space capabilities.

Definition at line 436 of file PiHob.h.

13.44.2 Member Data Documentation

13.44.2.1 Header

[EFI_HOB_GENERIC_HEADER](#) `EFI_HOB_CPU::Header`

The HOB generic header.

`Header.HobType = EFI_HOB_TYPE_CPU.`

Definition at line 440 of file PiHob.h.

The documentation for this struct was generated from the following file:

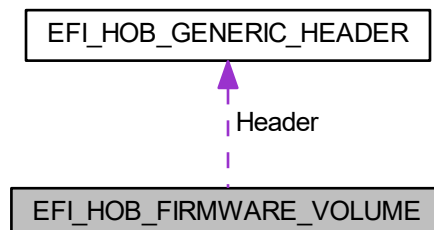
- [PiHob.h](#)

13.45 EFI_HOB_FIRMWARE_VOLUME Struct Reference

Details the location of firmware volumes that contain firmware files.

```
#include <PiHob.h>
```

Collaboration diagram for `EFI_HOB_FIRMWARE_VOLUME`:



Public Attributes

- [EFI_HOB_GENERIC_HEADER](#) Header
The HOB generic header.
- [EFI_PHYSICAL_ADDRESS](#) BaseAddress
The physical memory-mapped base address of the firmware volume.
- [UINT64](#) Length
The length in bytes of the firmware volume.

13.45.1 Detailed Description

Details the location of firmware volumes that contain firmware files.

Definition at line 353 of file PiHob.h.

13.45.2 Member Data Documentation

13.45.2.1 Header

[EFI_HOB_GENERIC_HEADER](#) `EFI_HOB_FIRMWARE_VOLUME::Header`

The HOB generic header.

Header.HobType = `EFI_HOB_TYPE_FV`.

Definition at line 357 of file PiHob.h.

The documentation for this struct was generated from the following file:

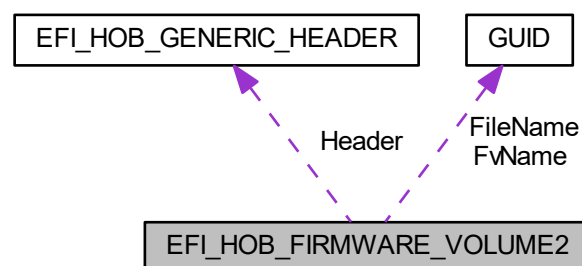
- [PiHob.h](#)

13.46 EFI_HOB_FIRMWARE_VOLUME2 Struct Reference

Details the location of a firmware volume that was extracted from a file within another firmware volume.

```
#include <PiHob.h>
```

Collaboration diagram for `EFI_HOB_FIRMWARE_VOLUME2`:



Public Attributes

- [EFI_HOB_GENERIC_HEADER](#) Header
The HOB generic header.
- [EFI_PHYSICAL_ADDRESS](#) BaseAddress
The physical memory-mapped base address of the firmware volume.
- [UINT64](#) Length
The length in bytes of the firmware volume.
- [EFI_GUID](#) FvName
The name of the firmware volume.
- [EFI_GUID](#) FileName
The name of the firmware file that contained this firmware volume.

13.46.1 Detailed Description

Details the location of a firmware volume that was extracted from a file within another firmware volume.

Definition at line 372 of file PiHob.h.

13.46.2 Member Data Documentation

13.46.2.1 Header

[EFI_HOB_GENERIC_HEADER](#) `EFI_HOB_FIRMWARE_VOLUME2::Header`

The HOB generic header.

`Header.HobType = EFI_HOB_TYPE_FV2.`

Definition at line 376 of file PiHob.h.

The documentation for this struct was generated from the following file:

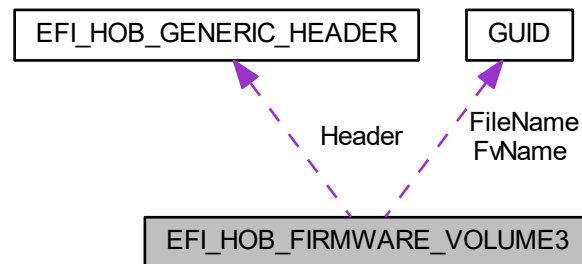
- [PiHob.h](#)

13.47 EFI_HOB_FIRMWARE_VOLUME3 Struct Reference

Details the location of a firmware volume that was extracted from a file within another firmware volume.

```
#include <PiHob.h>
```

Collaboration diagram for EFI_HOB_FIRMWARE_VOLUME3:



Public Attributes

- [EFI_HOB_GENERIC_HEADER Header](#)
The HOB generic header.
- [EFI_PHYSICAL_ADDRESS BaseAddress](#)
The physical memory-mapped base address of the firmware volume.
- [UINT64 Length](#)
The length in bytes of the firmware volume.
- [UINT32 AuthenticationStatus](#)
The authentication status.
- [BOOLEAN ExtractedFv](#)
TRUE if the FV was extracted as a file within another firmware volume.
- [EFI_GUID FvName](#)
The name of the firmware volume.
- [EFI_GUID FileName](#)
The name of the firmware file that contained this firmware volume.

13.47.1 Detailed Description

Details the location of a firmware volume that was extracted from a file within another firmware volume.

Definition at line 399 of file PiHob.h.

13.47.2 Member Data Documentation

13.47.2.1 ExtractedFv

`BOOLEAN EFI_HOB_FIRMWARE_VOLUME3::ExtractedFv`

TRUE if the FV was extracted as a file within another firmware volume.

FALSE otherwise.

Definition at line 420 of file PiHob.h.

13.47.2.2 FileName

`EFI_GUID EFI_HOB_FIRMWARE_VOLUME3::FileName`

The name of the firmware file that contained this firmware volume.

Valid only if IsExtractedFv is TRUE.

Definition at line 430 of file PiHob.h.

13.47.2.3 FvName

`EFI_GUID EFI_HOB_FIRMWARE_VOLUME3::FvName`

The name of the firmware volume.

Valid only if IsExtractedFv is TRUE.

Definition at line 425 of file PiHob.h.

13.47.2.4 Header

`EFI_HOB_GENERIC_HEADER EFI_HOB_FIRMWARE_VOLUME3::Header`

The HOB generic header.

Header.HobType = EFI_HOB_TYPE_FV3.

Definition at line 403 of file PiHob.h.

The documentation for this struct was generated from the following file:

- [PiHob.h](#)

13.48 EFI_HOB_GENERIC_HEADER Struct Reference

Describes the format and size of the data inside the HOB.

```
#include <PiHob.h>
```

Public Attributes

- UINT16 [HobType](#)
Identifies the HOB data structure type.
- UINT16 [HobLength](#)
The length in bytes of the HOB.
- UINT32 [Reserved](#)
This field must always be set to zero.

13.48.1 Detailed Description

Describes the format and size of the data inside the HOB.

All HOBs must contain this generic HOB header.

Definition at line 36 of file PiHob.h.

The documentation for this struct was generated from the following file:

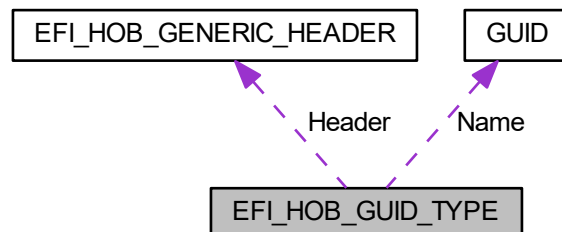
- [PiHob.h](#)

13.49 EFI_HOB_GUID_TYPE Struct Reference

Allows writers of executable content in the HOB producer phase to maintain and manage HOBs with specific [GUID](#).

```
#include <PiHob.h>
```

Collaboration diagram for EFI_HOB_GUID_TYPE:



Public Attributes

- [EFI_HOB_GENERIC_HEADER](#) Header
The HOB generic header.
- [EFI_GUID](#) Name
A [GUID](#) that defines the contents of this HOB.

13.49.1 Detailed Description

Allows writers of executable content in the HOB producer phase to maintain and manage HOBs with specific [GUID](#).

Definition at line 336 of file PiHob.h.

13.49.2 Member Data Documentation

13.49.2.1 Header

[EFI_HOB_GENERIC_HEADER](#) `EFI_HOB_GUID_TYPE::Header`

The HOB generic header.

`Header.HobType = EFI_HOB_TYPE_GUID_EXTENSION.`

Definition at line 340 of file PiHob.h.

The documentation for this struct was generated from the following file:

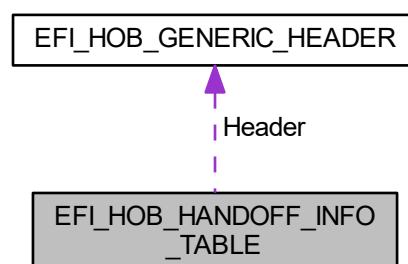
- [PiHob.h](#)

13.50 EFI_HOB_HANDOFF_INFO_TABLE Struct Reference

Contains general state information used by the HOB producer phase.

```
#include <PiHob.h>
```

Collaboration diagram for `EFI_HOB_HANDOFF_INFO_TABLE`:



Public Attributes

- [EFI_HOB_GENERIC_HEADER](#) Header
The HOB generic header.
- [UINT32](#) Version
The version number pertaining to the PHIT HOB definition.
- [EFI_BOOT_MODE](#) BootMode
The system boot mode as determined during the HOB producer phase.
- [EFI_PHYSICAL_ADDRESS](#) EfiMemoryTop
The highest address location of memory that is allocated for use by the HOB producer phase.
- [EFI_PHYSICAL_ADDRESS](#) EfiMemoryBottom
The lowest address location of memory that is allocated for use by the HOB producer phase.
- [EFI_PHYSICAL_ADDRESS](#) EfiFreeMemoryTop
The highest address location of free memory that is currently available for use by the HOB producer phase.
- [EFI_PHYSICAL_ADDRESS](#) EfiFreeMemoryBottom
The lowest address location of free memory that is available for use by the HOB producer phase.
- [EFI_PHYSICAL_ADDRESS](#) EfiEndOfHobList
The end of the HOB list.

13.50.1 Detailed Description

Contains general state information used by the HOB producer phase.

This HOB must be the first one in the HOB list.

Definition at line 60 of file PiHob.h.

13.50.2 Member Data Documentation

13.50.2.1 EfiMemoryTop

[EFI_PHYSICAL_ADDRESS](#) `EFI_HOB_HANDOFF_INFO_TABLE::EfiMemoryTop`

The highest address location of memory that is allocated for use by the HOB producer phase.

This address must be 4-KB aligned to meet page restrictions of UEFI.

Definition at line 79 of file PiHob.h.

13.50.2.2 Header

[EFI_HOB_GENERIC_HEADER](#) `EFI_HOB_HANDOFF_INFO_TABLE::Header`

The HOB generic header.

Header.HobType = `EFI_HOB_TYPE_HANDOFF`.

Definition at line 64 of file PiHob.h.

13.50.2.3 Version

```
UINT32 EFI_HOB_HANDOFF_INFO_TABLE::Version
```

The version number pertaining to the PHIT HOB definition.

This value is four bytes in length to provide an 8-byte aligned entry when it is combined with the 4-byte BootMode.

Definition at line 70 of file PiHob.h.

The documentation for this struct was generated from the following file:

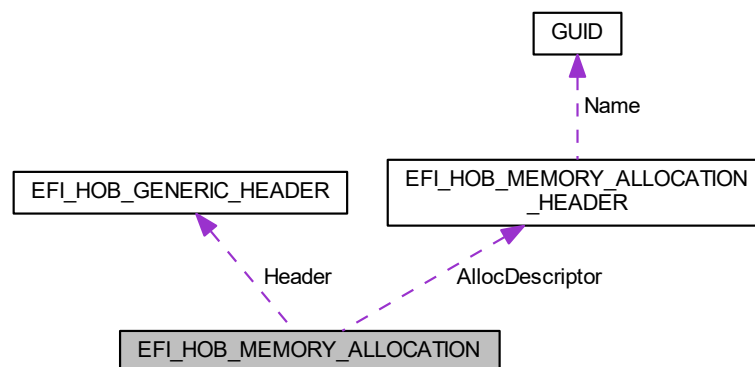
- [PiHob.h](#)

13.51 EFI_HOB_MEMORY_ALLOCATION Struct Reference

Describes all memory ranges used during the HOB producer phase that exist outside the HOB list.

```
#include <PiHob.h>
```

Collaboration diagram for EFI_HOB_MEMORY_ALLOCATION:



Public Attributes

- [EFI_HOB_GENERIC_HEADER](#) Header
The HOB generic header.
- [EFI_HOB_MEMORY_ALLOCATION_HEADER](#) AllocDescriptor
An instance of the [EFI_HOB_MEMORY_ALLOCATION_HEADER](#) that describes the various attributes of the logical memory allocation.

13.51.1 Detailed Description

Describes all memory ranges used during the HOB producer phase that exist outside the HOB list.

This HOB type describes how memory is used, not the physical attributes of memory.

Definition at line 144 of file PiHob.h.

13.51.2 Member Data Documentation

13.51.2.1 Header

[EFI_HOB_GENERIC_HEADER](#) `EFI_HOB_MEMORY_ALLOCATION::Header`

The HOB generic header.

Header.HobType = EFI_HOB_TYPE_MEMORY_ALLOCATION.

Definition at line 148 of file PiHob.h.

The documentation for this struct was generated from the following file:

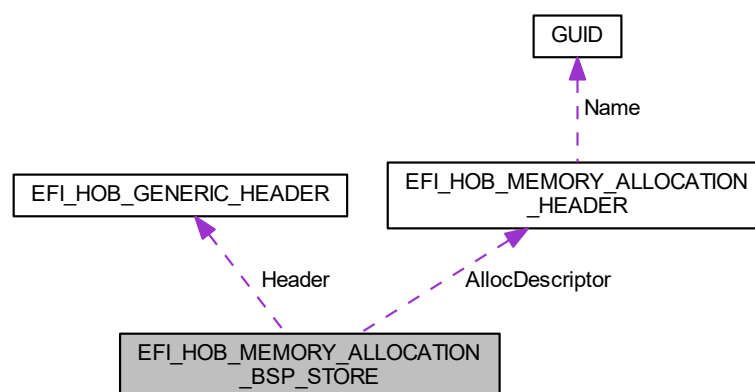
- [PiHob.h](#)

13.52 EFI_HOB_MEMORY_ALLOCATION_BSP_STORE Struct Reference

Defines the location of the boot-strap processor (BSP) BSPStore ("Backing Store Pointer Store").

```
#include <PiHob.h>
```

Collaboration diagram for EFI_HOB_MEMORY_ALLOCATION_BSP_STORE:



Public Attributes

- [EFI_HOB_GENERIC_HEADER](#) Header
The HOB generic header.
- [EFI_HOB_MEMORY_ALLOCATION_HEADER](#) AllocDescriptor
An instance of the [EFI_HOB_MEMORY_ALLOCATION_HEADER](#) that describes the various attributes of the logical memory allocation.

13.52.1 Detailed Description

Defines the location of the boot-strap processor (BSP) BSPStore ("Backing Store Pointer Store").

This HOB is valid for the Itanium processor family only register overflow store.

Definition at line 183 of file PiHob.h.

13.52.2 Member Data Documentation

13.52.2.1 Header

[EFI_HOB_GENERIC_HEADER](#) [EFI_HOB_MEMORY_ALLOCATION_BSP_STORE::Header](#)

The HOB generic header.

Header.HobType = [EFI_HOB_TYPE_MEMORY_ALLOCATION](#).

Definition at line 187 of file PiHob.h.

The documentation for this struct was generated from the following file:

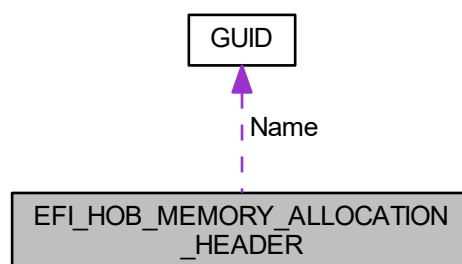
- [PiHob.h](#)

13.53 EFI_HOB_MEMORY_ALLOCATION_HEADER Struct Reference

[EFI_HOB_MEMORY_ALLOCATION_HEADER](#) describes the various attributes of the logical memory allocation.

```
#include <PiHob.h>
```

Collaboration diagram for [EFI_HOB_MEMORY_ALLOCATION_HEADER](#):



Public Attributes

- [EFI_GUID](#) [Name](#)
A *GUID* that defines the memory allocation region's type and purpose, as well as other fields within the memory allocation HOB.
- [EFI_PHYSICAL_ADDRESS](#) [MemoryBaseAddress](#)
The base address of memory allocated by this HOB.
- [UINT64](#) [MemoryLength](#)
The length in bytes of memory allocated by this HOB.
- [EFI_MEMORY_TYPE](#) [MemoryType](#)
Defines the type of memory allocated by this HOB.
- [UINT8](#) [Reserved](#) [4]
Padding for Itanium processor family.

13.53.1 Detailed Description

[EFI_HOB_MEMORY_ALLOCATION_HEADER](#) describes the various attributes of the logical memory allocation.

The type field will be used for subsequent inclusion in the UEFI memory map.

Definition at line 104 of file PiHob.h.

13.53.2 Member Data Documentation

13.53.2.1 MemoryBaseAddress

```
EFI\_PHYSICAL\_ADDRESS EFI\_HOB\_MEMORY\_ALLOCATION\_HEADER::MemoryBaseAddress
```

The base address of memory allocated by this HOB.

Type [EFI_PHYSICAL_ADDRESS](#) is defined in [AllocatePages\(\)](#) in the UEFI 2.0 specification.

Definition at line 119 of file PiHob.h.

13.53.2.2 MemoryType

```
EFI\_MEMORY\_TYPE EFI\_HOB\_MEMORY\_ALLOCATION\_HEADER::MemoryType
```

Defines the type of memory allocated by this HOB.

The memory type definition follows the [EFI_MEMORY_TYPE](#) definition. Type [EFI_MEMORY_TYPE](#) is defined in [AllocatePages\(\)](#) in the UEFI 2.0 specification.

Definition at line 131 of file PiHob.h.

13.53.2.3 Name

[EFI_GUID](#) `EFI_HOB_MEMORY_ALLOCATION_HEADER::Name`

A [GUID](#) that defines the memory allocation region's type and purpose, as well as other fields within the memory allocation HOB.

This [GUID](#) is used to define the additional data within the HOB that may be present for the memory allocation HOB. Type `EFI_GUID` is defined in `InstallProtocolInterface()` in the UEFI 2.0 specification.

Definition at line 112 of file `PiHob.h`.

The documentation for this struct was generated from the following file:

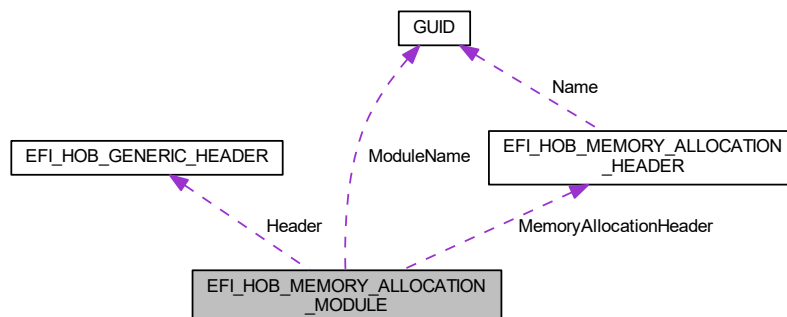
- [PiHob.h](#)

13.54 EFI_HOB_MEMORY_ALLOCATION_MODULE Struct Reference

Defines the location and entry point of the HOB consumer phase.

```
#include <PiHob.h>
```

Collaboration diagram for `EFI_HOB_MEMORY_ALLOCATION_MODULE`:



Public Attributes

- [EFI_HOB_GENERIC_HEADER](#) `Header`
The HOB generic header.
- [EFI_HOB_MEMORY_ALLOCATION_HEADER](#) `MemoryAllocationHeader`
An instance of the [EFI_HOB_MEMORY_ALLOCATION_HEADER](#) that describes the various attributes of the logical memory allocation.
- [EFI_GUID](#) `ModuleName`
The [GUID](#) specifying the values of the firmware file system name that contains the HOB consumer phase component.
- [EFI_PHYSICAL_ADDRESS](#) `EntryPoint`
The address of the memory-mapped firmware volume that contains the HOB consumer phase firmware file.

13.54.1 Detailed Description

Defines the location and entry point of the HOB consumer phase.

Definition at line 198 of file PiHob.h.

13.54.2 Member Data Documentation

13.54.2.1 Header

[EFI_HOB_GENERIC_HEADER](#) `EFI_HOB_MEMORY_ALLOCATION_MODULE::Header`

The HOB generic header.

`Header.HobType = EFI_HOB_TYPE_MEMORY_ALLOCATION.`

Definition at line 202 of file PiHob.h.

The documentation for this struct was generated from the following file:

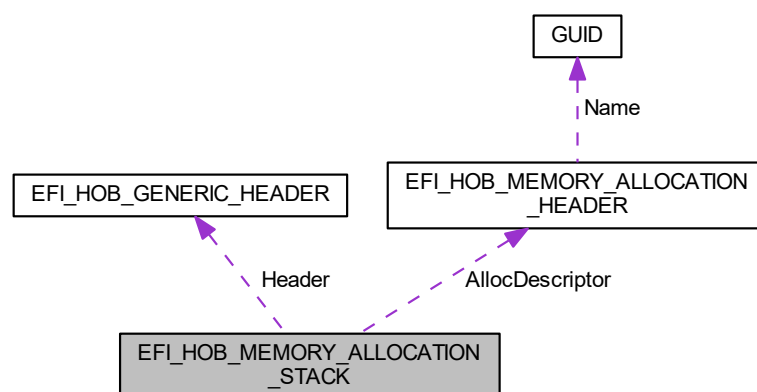
- [PiHob.h](#)

13.55 EFI_HOB_MEMORY_ALLOCATION_STACK Struct Reference

Describes the memory stack that is produced by the HOB producer phase and upon which all post-memory-installed executable content in the HOB producer phase is executing.

```
#include <PiHob.h>
```

Collaboration diagram for `EFI_HOB_MEMORY_ALLOCATION_STACK`:



Public Attributes

- [EFI_HOB_GENERIC_HEADER](#) Header
The HOB generic header.
- [EFI_HOB_MEMORY_ALLOCATION_HEADER](#) AllocDescriptor
An instance of the [EFI_HOB_MEMORY_ALLOCATION_HEADER](#) that describes the various attributes of the logical memory allocation.

13.55.1 Detailed Description

Describes the memory stack that is produced by the HOB producer phase and upon which all post-memory-installed executable content in the HOB producer phase is executing.

Definition at line 165 of file PiHob.h.

13.55.2 Member Data Documentation

13.55.2.1 Header

[EFI_HOB_GENERIC_HEADER](#) `EFI_HOB_MEMORY_ALLOCATION_STACK::Header`

The HOB generic header.

Header.HobType = `EFI_HOB_TYPE_MEMORY_ALLOCATION`.

Definition at line 169 of file PiHob.h.

The documentation for this struct was generated from the following file:

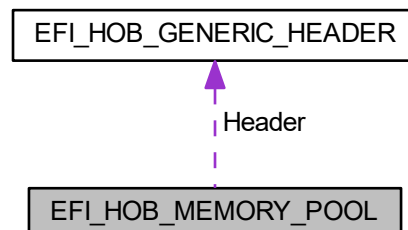
- [PiHob.h](#)

13.56 EFI_HOB_MEMORY_POOL Struct Reference

Describes pool memory allocations.

```
#include <PiHob.h>
```

Collaboration diagram for `EFI_HOB_MEMORY_POOL`:



Public Attributes

- [EFI_HOB_GENERIC_HEADER](#) Header

The HOB generic header.

13.56.1 Detailed Description

Describes pool memory allocations.

Definition at line 458 of file PiHob.h.

13.56.2 Member Data Documentation

13.56.2.1 Header

[EFI_HOB_GENERIC_HEADER](#) `EFI_HOB_MEMORY_POOL::Header`

The HOB generic header.

`Header.HobType = EFI_HOB_TYPE_MEMORY_POOL.`

Definition at line 462 of file PiHob.h.

The documentation for this struct was generated from the following file:

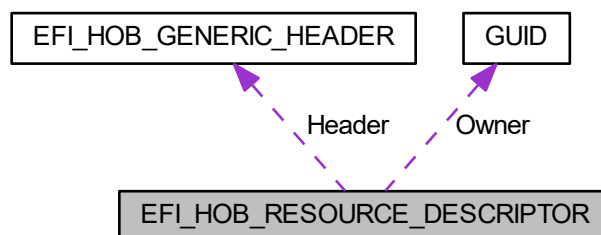
- [PiHob.h](#)

13.57 EFI_HOB_RESOURCE_DESCRIPTOR Struct Reference

Describes the resource properties of all fixed, nonrelocatable resource ranges found on the processor host bus during the HOB producer phase.

```
#include <PiHob.h>
```

Collaboration diagram for `EFI_HOB_RESOURCE_DESCRIPTOR`:



Public Attributes

- [EFI_HOB_GENERIC_HEADER](#) Header
The HOB generic header.
- [EFI_GUID](#) Owner
A [GUID](#) representing the owner of the resource.
- [EFI_RESOURCE_TYPE](#) ResourceType
The resource type enumeration as defined by [EFI_RESOURCE_TYPE](#).
- [EFI_RESOURCE_ATTRIBUTE_TYPE](#) ResourceAttribute
Resource attributes as defined by [EFI_RESOURCE_ATTRIBUTE_TYPE](#).
- [EFI_PHYSICAL_ADDRESS](#) PhysicalStart
The physical start address of the resource region.
- [UINT64](#) [ResourceLength](#)
The number of bytes of the resource region.

13.57.1 Detailed Description

Describes the resource properties of all fixed, nonrelocatable resource ranges found on the processor host bus during the HOB producer phase.

Definition at line 304 of file PiHob.h.

13.57.2 Member Data Documentation

13.57.2.1 Header

[EFI_HOB_GENERIC_HEADER](#) [EFI_HOB_RESOURCE_DESCRIPTOR::Header](#)

The HOB generic header.

Header.HobType = [EFI_HOB_TYPE_RESOURCE_DESCRIPTOR](#).

Definition at line 308 of file PiHob.h.

13.57.2.2 Owner

[EFI_GUID](#) [EFI_HOB_RESOURCE_DESCRIPTOR::Owner](#)

A [GUID](#) representing the owner of the resource.

This [GUID](#) is used by HOB consumer phase components to correlate device ownership of a resource.

Definition at line 313 of file PiHob.h.

The documentation for this struct was generated from the following file:

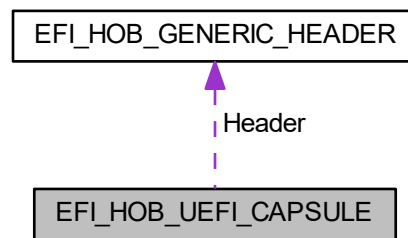
- [PiHob.h](#)

13.58 EFI_HOB_UEFI_CAPSULE Struct Reference

Each UEFI capsule HOB details the location of a UEFI capsule.

```
#include <PiHob.h>
```

Collaboration diagram for EFI_HOB_UEFI_CAPSULE:



Public Attributes

- [EFI_HOB_GENERIC_HEADER Header](#)
The HOB generic header where Header.HobType = EFI_HOB_TYPE_UEFI_CAPSULE.
- [EFI_PHYSICAL_ADDRESS BaseAddress](#)
The physical memory-mapped base address of an UEFI capsule.

13.58.1 Detailed Description

Each UEFI capsule HOB details the location of a UEFI capsule.

It includes a base address and length which is based upon memory blocks with a `EFI_CAPSULE_HEADER` and the associated `CapsuleImageSize`-based payloads. These HOB's shall be created by the PEI PI firmware sometime after the UEFI UpdateCapsule service invocation with the `CAPSULE_FLAGS_POPULATE_SYSTEM_TABLE` flag set in the `EFI_CAPSULE_HEADER`.

Definition at line 472 of file `PiHob.h`.

13.58.2 Member Data Documentation

13.58.2.1 BaseAddress

`EFI_PHYSICAL_ADDRESS` `EFI_HOB_UEFI_CAPSULE::BaseAddress`

The physical memory-mapped base address of an UEFI capsule.

This value is set to point to the base of the contiguous memory of the UEFI capsule. The length of the contiguous memory in bytes.

Definition at line 483 of file PiHob.h.

The documentation for this struct was generated from the following file:

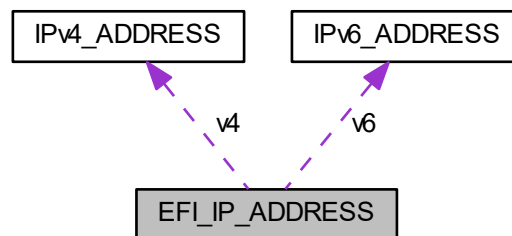
- [PiHob.h](#)

13.59 EFI_IP_ADDRESS Union Reference

16-byte buffer aligned on a 4-byte boundary.

```
#include <UefiBaseType.h>
```

Collaboration diagram for EFI_IP_ADDRESS:



13.59.1 Detailed Description

16-byte buffer aligned on a 4-byte boundary.

An IPv4 or IPv6 internet protocol address.

Definition at line 102 of file UefiBaseType.h.

The documentation for this union was generated from the following file:

- [UefiBaseType.h](#)

13.60 EFI_MAC_ADDRESS Struct Reference

32-byte buffer containing a network Media Access Control address.

```
#include <UefiBaseType.h>
```

13.60.1 Detailed Description

32-byte buffer containing a network Media Access Control address.

Definition at line 94 of file UefiBaseType.h.

The documentation for this struct was generated from the following file:

- [UefiBaseType.h](#)

13.61 EFI_MMRAM_DESCRIPTOR Struct Reference

Structure describing a MMRAM region and its accessibility attributes.

```
#include <PiMultiPhase.h>
```

Public Attributes

- [EFI_PHYSICAL_ADDRESS](#) [PhysicalStart](#)
Designates the physical address of the MMRAM in memory.
- [EFI_PHYSICAL_ADDRESS](#) [CpuStart](#)
Designates the address of the MMRAM, as seen by software executing on the processors.
- UINT64 [PhysicalSize](#)
Describes the number of bytes in the MMRAM region.
- UINT64 [RegionState](#)
Describes the accessibility attributes of the MMRAM.

13.61.1 Detailed Description

Structure describing a MMRAM region and its accessibility attributes.

Definition at line 109 of file PiMultiPhase.h.

13.61.2 Member Data Documentation

13.61.2.1 CpuStart

`EFI_PHYSICAL_ADDRESS` `EFI_MMRAM_DESCRIPTOR::CpuStart`

Designates the address of the MMRAM, as seen by software executing on the processors.

This address may or may not match `PhysicalStart`.

Definition at line 120 of file `PiMultiPhase.h`.

13.61.2.2 PhysicalStart

`EFI_PHYSICAL_ADDRESS` `EFI_MMRAM_DESCRIPTOR::PhysicalStart`

Designates the physical address of the MMRAM in memory.

This view of memory is the same as seen by I/O-based agents, for example, but it may not be the address seen by the processors.

Definition at line 115 of file `PiMultiPhase.h`.

13.61.2.3 RegionState

`UINT64` `EFI_MMRAM_DESCRIPTOR::RegionState`

Describes the accessibility attributes of the MMRAM.

These attributes include the hardware state (e.g., Open/Closed/Locked), capability (e.g., cacheable), logical allocation (e.g., allocated), and pre-use initialization (e.g., needs testing/ECC initialization).

Definition at line 131 of file `PiMultiPhase.h`.

The documentation for this struct was generated from the following file:

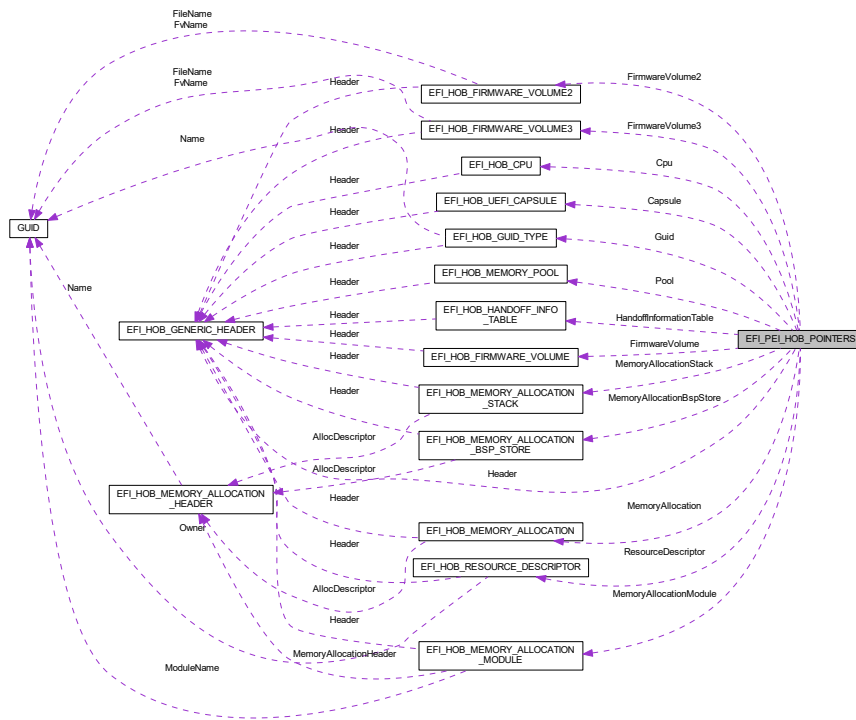
- [PiMultiPhase.h](#)

13.62 EFI_PEI_HOB_POINTERS Union Reference

Union of all the possible HOB Types.

```
#include <PiHob.h>
```

Collaboration diagram for EFI_PEI_HOB_POINTERS:



13.62.1 Detailed Description

Union of all the possible HOB Types.

Definition at line 490 of file `PiHob.h`.

The documentation for this union was generated from the following file:

- [PiHob.h](#)

13.63 EFI_TIME Struct Reference

EFI Time Abstraction: Year: 1900 - 9999 Month: 1 - 12 Day: 1 - 31 Hour: 0 - 23 Minute: 0 - 59 Second: 0 - 59 Nanosecond: 0 - 999,999,999 TimeZone: -1440 to 1440 or 2047.

```
#include <UefiBaseType.h>
```

13.63.1 Detailed Description

EFI Time Abstraction: Year: 1900 - 9999 Month: 1 - 12 Day: 1 - 31 Hour: 0 - 23 Minute: 0 - 59 Second: 0 - 59 Nanosecond: 0 - 999,999,999 TimeZone: -1440 to 1440 or 2047.

Definition at line 67 of file UefiBaseType.h.

The documentation for this struct was generated from the following file:

- [UefiBaseType.h](#)

13.64 FIRMWARE_VERSION Struct Reference

Firmware Version Structure.

```
#include <FirmwareVersionInfoHob.h>
```

13.64.1 Detailed Description

Firmware Version Structure.

Definition at line 28 of file FirmwareVersionInfoHob.h.

The documentation for this struct was generated from the following file:

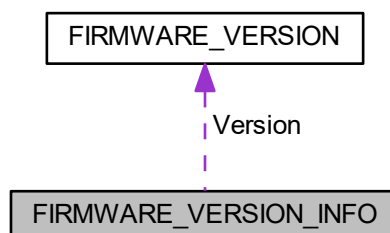
- [FirmwareVersionInfoHob.h](#)

13.65 FIRMWARE_VERSION_INFO Struct Reference

Firmware Version Information Structure.

```
#include <FirmwareVersionInfoHob.h>
```

Collaboration diagram for FIRMWARE_VERSION_INFO:



Public Attributes

- [UINT8 ComponentNameIndex](#)
Offset 0 Index of Component Name.
- [UINT8 VersionStringIndex](#)
Offset 1 Index of Version String.
- [FIRMWARE_VERSION Version](#)
Offset 2-6 Firmware version.

13.65.1 Detailed Description

Firmware Version Information Structure.

Definition at line 38 of file `FirmwareVersionInfoHob.h`.

The documentation for this struct was generated from the following file:

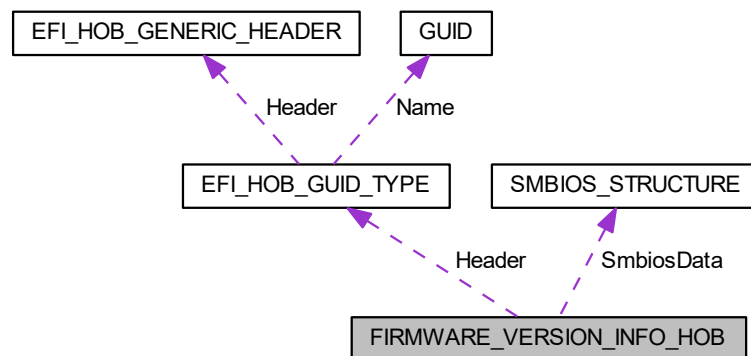
- [FirmwareVersionInfoHob.h](#)

13.66 FIRMWARE_VERSION_INFO_HOB Struct Reference

Firmware Version Information HOB Structure.

```
#include <FirmwareVersionInfoHob.h>
```

Collaboration diagram for FIRMWARE_VERSION_INFO_HOB:



Public Attributes

- [EFI_HOB_GUID_TYPE Header](#)
Offset 0-23 The header of FVI HOB.
- [SMBIOS_STRUCTURE SmbiosData](#)
Offset 24-27 The SMBIOS header of FVI HOB.
- [UINT8 Count](#)
Offset 28 Number of FVI elements included.

13.66.1 Detailed Description

Firmware Version Information HOB Structure.

Definition at line 58 of file FirmwareVersionInfoHob.h.

The documentation for this struct was generated from the following file:

- [FirmwareVersionInfoHob.h](#)

13.67 FIVR_EXT_RAIL_CONFIG Struct Reference

Structure for V1p05/Vnn VR rail configuration.

```
#include <FivrConfig.h>
```

Public Attributes

- UINT32 [EnabledStates](#): 6
*Mask to enable the usage of external VR rail in specific S0ix or Sx states Use values from FIVR_RAIL_SX_STATE The default is **FivrRailDisabled**.*
- UINT32 [Voltage](#): 11
VR rail voltage value that will be used in S0i2/S0i3 states.
- UINT32 [IccMax](#): 8
- UINT32 [RsvdBits1](#): 7
UINT32 Alignment.
- UINT32 [CtrlRampTmr](#): 8
This register holds the control hold off values to be used when changing the rail control for external bypass value in us.
- UINT32 [SupportedVoltageStates](#): 4
Mask to set the supported configuration in VR rail.
- UINT32 [IccMaximum](#): 16
*VR rail Icc Maximum Value Granularity of this setting is 1mA and maximal possible value is 500mA The default is **0mA**.*
- UINT32 [RsvdBits2](#): 4
UINT32 Alignment.

13.67.1 Detailed Description

Structure for V1p05/Vnn VR rail configuration.

Definition at line 74 of file FivrConfig.h.

13.67.2 Member Data Documentation

13.67.2.1 IccMax

```
UINT32 FIVR_EXT_RAIL_CONFIG::IccMax
```

Deprecated THIS POLICY IS DEPRECATED, PLEASE USE IccMaximum INSTEAD VR rail Icc Max Value Granularity of this setting is 1mA and maximal possible value is 500mA The default is **0mA** .

Definition at line 95 of file FivrConfig.h.

13.67.2.2 SupportedVoltageStates

```
UINT32 FIVR_EXT_RAIL_CONFIG::SupportedVoltageStates
```

Mask to set the supported configuration in VR rail.

Use values from FIVR_RAIL_SUPPORTED_VOLTAGE

Definition at line 110 of file FivrConfig.h.

13.67.2.3 Voltage

```
UINT32 FIVR_EXT_RAIL_CONFIG::Voltage
```

VR rail voltage value that will be used in S0i2/S0i3 states.

This value is given in 2.5mV increments (0=0mV, 1=2.5mV, 2=5mV...) The default for Vnn is set to **420 - 1050 mV**.

Definition at line 87 of file FivrConfig.h.

The documentation for this struct was generated from the following file:

- [FivrConfig.h](#)

13.68 FIVR_VCCIN_AUX_CONFIG Struct Reference

Structure for VCCIN_AUX voltage rail configuration.

```
#include <FivrConfig.h>
```

Public Attributes

- UINT8 [LowToHighCurModeVolTranTime](#)
Transition time in microseconds from Low Current Mode Voltage to High Current Mode Voltage.
- UINT8 [RetToHighCurModeVolTranTime](#)
Transition time in microseconds from Retention Mode Voltage to High Current Mode Voltage.
- UINT8 [RetToLowCurModeVolTranTime](#)
Transition time in microseconds from Retention Mode Voltage to Low Current Mode Voltage.
- UINT32 [OffToHighCurModeVolTranTime](#): 11
Transition time in microseconds from Off (0V) to High Current Mode Voltage.

13.68.1 Detailed Description

Structure for VCCIN_AUX voltage rail configuration.

Definition at line 129 of file FivrConfig.h.

13.68.2 Member Data Documentation

13.68.2.1 LowToHighCurModeVolTranTime

```
UINT8 FIVR_VCCIN_AUX_CONFIG::LowToHighCurModeVolTranTime
```

Transition time in microseconds from Low Current Mode Voltage to High Current Mode Voltage.

Voltage transition time required by motherboard voltage regulator when PCH changes the VCCIN_AUX regulator set point from the low current mode voltage and high current mode voltage. This field has 1us resolution. When value is 0 PCH will not transition VCCIN_AUX to low current mode voltage. The default is **0xC**.

Definition at line 138 of file FivrConfig.h.

13.68.2.2 OffToHighCurModeVolTranTime

```
UINT32 FIVR_VCCIN_AUX_CONFIG::OffToHighCurModeVolTranTime
```

Transition time in microseconds from Off (0V) to High Current Mode Voltage.

Voltage transition time required by motherboard voltage regulator when PCH changes the VCCIN_AUX regulator set point from 0V to the high current mode voltage. This field has 1us resolution. 0 = Transition to 0V is disabled Setting this field to 0 sets VCCIN_AUX as a fixed rail that stays on in all S0 & Sx power states after initial start up on G3 exit The default is **0x96**.

Definition at line 170 of file FivrConfig.h.

13.68.2.3 RetToHighCurModeVolTranTime

```
UINT8 FIVR_VCCIN_AUX_CONFIG::RetToHighCurModeVolTranTime
```

Transition time in microseconds from Retention Mode Voltage to High Current Mode Voltage.

Voltage transition time required by motherboard voltage regulator when PCH changes the VCCIN_AUX regulator set point from the retention mode voltage to high current mode voltage. This field has 1us resolution. When value is 0 PCH will not transition VCCIN_AUX to retention voltage. The default is **0x36**.

Definition at line 148 of file FivrConfig.h.

13.68.2.4 RetToLowCurModeVolTranTime

```
UINT8 FIVR_VCCIN_AUX_CONFIG::RetToLowCurModeVolTranTime
```

Transition time in microseconds from Retention Mode Voltage to Low Current Mode Voltage.

Voltage transition time required by motherboard voltage regulator when PCH changes the VCCIN_AUX regulator set point from the retention mode voltage to low current mode voltage. This field has 1us resolution. When value is 0 PCH will not transition VCCIN_AUX to retention voltage. The default is **0x2B**.

Definition at line 158 of file FivrConfig.h.

The documentation for this struct was generated from the following file:

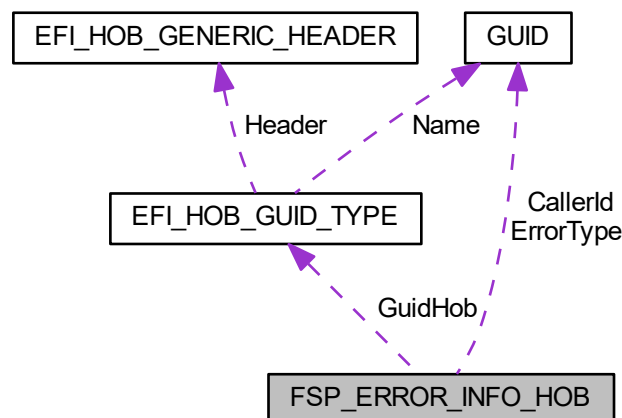
- [FivrConfig.h](#)

13.69 FSP_ERROR_INFO_HOB Struct Reference

FSP Error Information Block.

```
#include <FspErrorInfo.h>
```

Collaboration diagram for FSP_ERROR_INFO_HOB:



Public Attributes

- [EFI_HOB_GUID_TYPE](#) [GuidHob](#)
GUID HOB header.
- [EFI_STATUS_CODE_TYPE](#) [Type](#)
ReportStatusCode () type identifier.
- [EFI_STATUS_CODE_VALUE](#) [Value](#)
ReportStatusCode () value.
- [UINT32](#) [Instance](#)
ReportStatusCode () Instance number.
- [EFI_GUID](#) [CallerId](#)
Optional GUID which may be used to identify which internal component of the FSP was executing at the time of the error.
- [EFI_GUID](#) [ErrorType](#)
GUID identifying the nature of the fatal error.
- [UINT32](#) [Status](#)
EFI_STATUS code describing the error encountered.

13.69.1 Detailed Description

FSP Error Information Block.

Definition at line 60 of file FspErrorInfo.h.

The documentation for this struct was generated from the following file:

- [FspErrorInfo.h](#)

13.70 FSPM_ARCH_CONFIG_PPI Struct Reference

This PPI provides FSP-M Arch Config PPI.

```
#include <FspmArchConfigPpi.h>
```

13.70.1 Detailed Description

This PPI provides FSP-M Arch Config PPI.

Definition at line 32 of file FspmArchConfigPpi.h.

The documentation for this struct was generated from the following file:

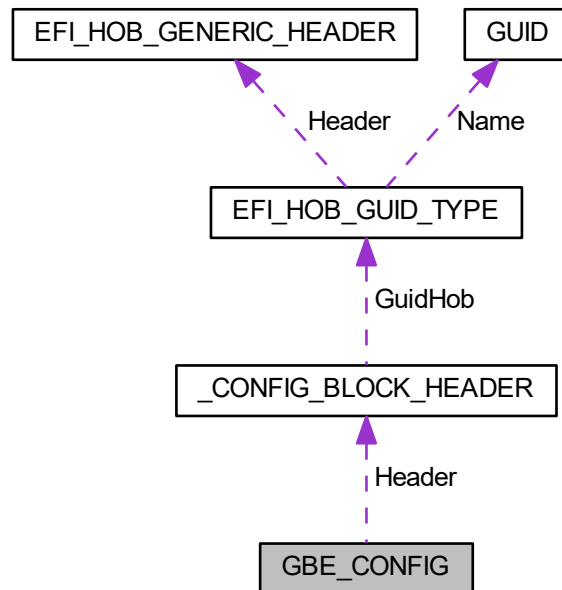
- [FspmArchConfigPpi.h](#)

13.71 GBE_CONFIG Struct Reference

PCH intergrated GBE controller configuration settings.

```
#include <GbeConfig.h>
```

Collaboration diagram for GBE_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- UINT32 [Enable](#): 1
*Determines if enable PCH internal GBE, 0: Disable; 1: **Enable**.*
- UINT32 [LtrEnable](#): 1
***0**: **Disable**; 1: Enable LTR capabilty of PCH internal LAN.*
- UINT32 [RsvdBits0](#): 30
Reserved bits.

13.71.1 Detailed Description

PCH intergrated GBE controller configuration settings.

Definition at line 46 of file GbeConfig.h.

13.71.2 Member Data Documentation

13.71.2.1 Enable

UINT32 GBE_CONFIG::Enable

Determines if enable PCH internal GBE, 0: Disable; 1: **Enable**.

When Enable is changed (from disabled to enabled or from enabled to disabled), it needs to set LAN Disable register, which might be locked by FDSWL register. So it's recommended to issue a global reset when changing the status for PCH Internal LAN.

Definition at line 54 of file GbeConfig.h.

The documentation for this struct was generated from the following file:

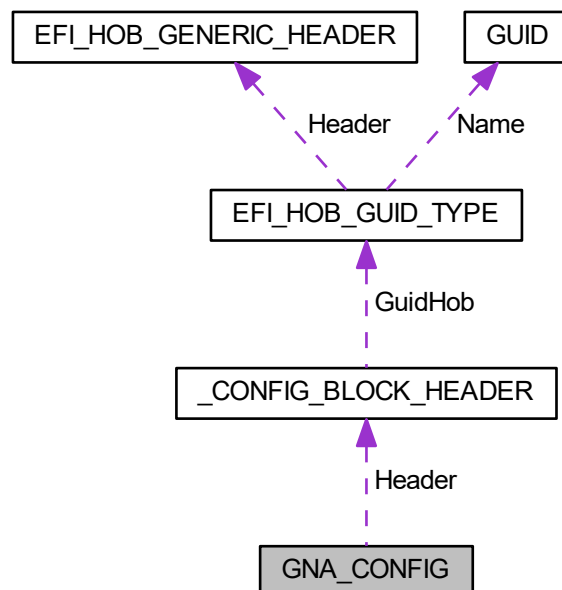
- [GbeConfig.h](#)

13.72 GNA_CONFIG Struct Reference

GNA config block for configuring GNA.

```
#include <GnaConfig.h>
```

Collaboration diagram for GNA_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.
- UINT32 [GnaEnable](#): 1
Offset 28:0 This policy enables the GNA Device (SA Device 8) if supported.
- UINT32 [RsvdBits0](#): 31
Offset 28:1 :Reserved for future use.

13.72.1 Detailed Description

GNA config block for configuring GNA.

Revision 1:

- Initial version.

Definition at line 45 of file GnaConfig.h.

13.72.2 Member Data Documentation

13.72.2.1 GnaEnable

UINT32 GNA_CONFIG::GnaEnable

Offset 28:0 This policy enables the GNA Device (SA Device 8) if supported.

If FALSE, all other policies in this config block will be ignored. **1=TRUE**; 0=FALSE.

Definition at line 54 of file GnaConfig.h.

The documentation for this struct was generated from the following file:

- [GnaConfig.h](#)

13.73 GPIO_CONFIG Struct Reference

GPIO configuration structure used for pin programming.

```
#include <GpioConfig.h>
```

Public Attributes

- UINT32 [PadMode](#): 5
Pad Mode Pad can be set as GPIO or one of its native functions.
- UINT32 [HostSoftPadOwn](#): 2
Host Software Pad Ownership Set pad to ACPI mode or GPIO Driver Mode.
- UINT32 [Direction](#): 6
GPIO Direction Can choose between In, In with inversion, Out, both In and Out, both In with inversion and out or disabling both.
- UINT32 [OutputState](#): 2
Output State Set Pad output value.
- UINT32 [InterruptConfig](#): 9
GPIO Interrupt Configuration Set Pad to cause one of interrupts (IOxAPIC/SCI/SMI/NMI).
- UINT32 [PowerConfig](#): 8
GPIO Power Configuration.
- UINT32 [ElectricalConfig](#): 9
GPIO Electrical Configuration This setting controls pads termination and voltage tolerance.
- UINT32 [LockConfig](#): 4
GPIO Lock Configuration This setting controls pads lock.
- UINT32 [OtherSettings](#): 2
Additional GPIO configuration Refer to definition of GPIO_OTHER_CONFIG for supported settings.
- UINT32 [RsvdBits](#): 17
Reserved bits for future extension.

13.73.1 Detailed Description

GPIO configuration structure used for pin programming.

Structure contains fields that can be used to configure pad.

Definition at line 55 of file GpioConfig.h.

13.73.2 Member Data Documentation

13.73.2.1 Direction

```
UINT32 GPIO_CONFIG::Direction
```

GPIO Direction Can choose between In, In with inversion, Out, both In and Out, both In with inversion and out or disabling both.

Refer to definition of GPIO_DIRECTION for supported settings.

Definition at line 76 of file GpioConfig.h.

13.73.2.2 ElectricalConfig

```
UINT32 GPIO_CONFIG::ElectricalConfig
```

GPIO Electrical Configuration This setting controls pads termination and voltage tolerance.

Refer to definition of GPIO_ELECTRICAL_CONFIG for supported settings.

Definition at line 102 of file GpioConfig.h.

13.73.2.3 HostSoftPadOwn

```
UINT32 GPIO_CONFIG::HostSoftPadOwn
```

Host Software Pad Ownership Set pad to ACPI mode or GPIO Driver Mode.

Refer to definition of GPIO_HOSTSW_OWN.

Definition at line 70 of file GpioConfig.h.

13.73.2.4 InterruptConfig

```
UINT32 GPIO_CONFIG::InterruptConfig
```

GPIO Interrupt Configuration Set Pad to cause one of interrupts (IOxAPIC/SCI/SMI/NMI).

This setting is applicable only if GPIO is in GpioMode with input enabled. Refer to definition of GPIO_INT_CONFIG for supported settings.

Definition at line 90 of file GpioConfig.h.

13.73.2.5 LockConfig

```
UINT32 GPIO_CONFIG::LockConfig
```

GPIO Lock Configuration This setting controls pads lock.

Refer to definition of GPIO_LOCK_CONFIG for supported settings.

Definition at line 108 of file GpioConfig.h.

13.73.2.6 OutputState

UINT32 GPIO_CONFIG::OutputState

Output State Set Pad output value.

Refer to definition of GPIO_OUTPUT_STATE for supported settings. This setting takes place when output is enabled.

Definition at line 83 of file GpioConfig.h.

13.73.2.7 PadMode

UINT32 GPIO_CONFIG::PadMode

Pad Mode Pad can be set as GPIO or one of its native functions.

When in native mode setting Direction (except Inversion), OutputState, InterruptConfig, Host Software Pad Ownership and OutputStateLock are unnecessary. Refer to definition of GPIO_PAD_MODE. Refer to EDS for each native mode according to the pad.

Definition at line 64 of file GpioConfig.h.

13.73.2.8 PowerConfig

UINT32 GPIO_CONFIG::PowerConfig

GPIO Power Configuration.

This setting controls Pad Reset Configuration. Refer to definition of GPIO_RESET_CONFIG for supported settings.

Definition at line 96 of file GpioConfig.h.

The documentation for this struct was generated from the following file:

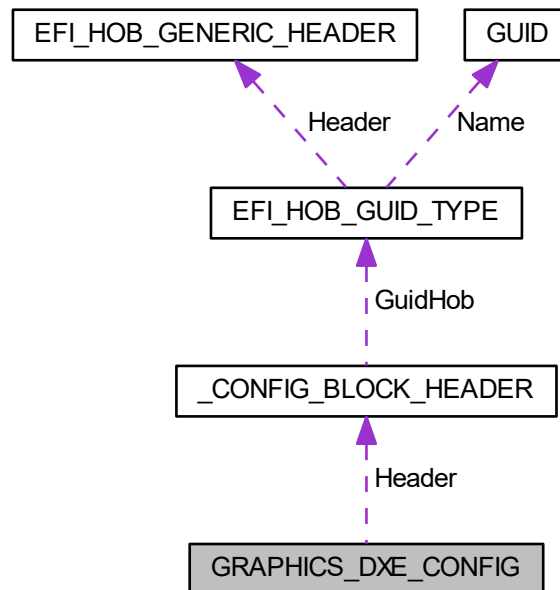
- [GpioConfig.h](#)

13.74 GRAPHICS_DXE_CONFIG Struct Reference

This configuration block is to configure IGD related variables used in DXE.

```
#include <GraphicsConfig.h>
```

Collaboration diagram for GRAPHICS_DXE_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Offset 0-27: Config Block Header.
- [UINT32 Size](#)
Offset 28 - 31: This field gives the size of the GOP VBT Data buffer.
- [EFI_PHYSICAL_ADDRESS VbtAddress](#)
Offset 32 - 39: This field points to the GOP VBT data buffer.
- [UINT8 PlatformConfig](#)
Offset 40: This field gives the Platform Configuration Information (0=Platform is S0ix Capable for ULT SKUs only, 1=Platform is not S0ix Capable, 2=Force Platform is S0ix Capable for All SKUs)
- [UINT8 AlsEnable](#)
Offset 41: Ambient Light Sensor Enable: 0=Disable, 2=Enable.
- [UINT8 BacklightControlSupport](#)
Offset 42: Backlight Control Support: 0=PWM Inverted, 2=PWM Normal
- [UINT8 IgdBootTest](#)
Offset 43: IGD Boot Type CMOS option: 0=Default, 0x01=CRT, 0x04=EFP, 0x08=LFP, 0x20=EFP3, 0x40=EFP2, 0x80=LFP2.
- [UINT32 IuerStatusVal](#)

Offset 44 - 47: Offset 16 This field holds the current status of all the supported Ultrabook events (Intel(R) Ultrabook Event Status bits)

- CHAR16 [GopVersion](#) [0x10]

Offset 48 - 79: This field holds the GOP Driver Version. It is an Output Protocol and updated by the Silicon code.

- UINT8 [IgdPanelType](#)

Offset 80: IGD Panel Type CMOS option

0=Default, 1=640X480LVDS, 2=800X600LVDS, 3=1024X768LVDS, 4=1280X1024LVDS, 5=1400X1050LVDS1
6=1400X1050LVDS2, 7=1600X1200LVDS, 8=1280X768LVDS, 9=1680X1050LVDS, 10=1920X1200LVDS,
13=1600X900LVDS
14=1280X800LVDS, 15=1280X600LVDS, 16=2048X1536LVDS, 17=1366X768LVDS.

- UINT8 [IgdPanelScaling](#)

Offset 81: IGD Panel Scaling: **0=AUTO**, 1=OFF, 6=Force scaling.

- UINT8 [IgdBlcConfig](#)

Offset 82: Backlight Control Support: 0=PWM Inverted, **2=PWM Normal**

- UINT8 [IgdDvmtMemSize](#)

Offset 83:

- UINT8 [GfxTurboIMON](#)

Offset 84: IMON Current Value: 14=Minimal, **31=Maximum**

- UINT8 [Reserved](#) [3]

Offset 85: Reserved for DWORD alignment.

- UINT16 [BCLM](#) [MAX_BCLM_ENTRIES]

Offset 88: IGD Backlight Brightness Level Duty cycle Mapping Table.

13.74.1 Detailed Description

This configuration block is to configure IGD related variables used in DXE.

If Intel Gfx Device is not supported or disabled, all policies will be ignored. The data elements should be initialized by a Platform Module.

Revision 1:

- Initial version. **Revision 2:**
- Removed IgdDvmtMemSize.

Definition at line 239 of file GraphicsConfig.h.

13.74.2 Member Data Documentation

13.74.2.1 IgdDvmtMemSize

```
UINT8 GRAPHICS_DXE_CONFIG::IgdDvmtMemSize
```

Offset 83:

Deprecated IGD DVMT Memory Size: 1=128MB, **2=256MB**, 3=MAX

Definition at line 258 of file GraphicsConfig.h.

The documentation for this struct was generated from the following file:

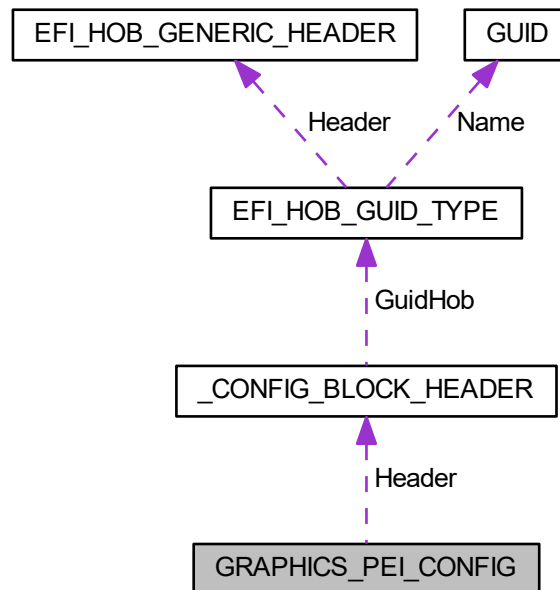
- [GraphicsConfig.h](#)

13.75 GRAPHICS_PEI_CONFIG Struct Reference

This configuration block is to configure IGD related variables used in PostMem PEI.

```
#include <GraphicsConfig.h>
```

Collaboration diagram for GRAPHICS_PEI_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Offset 0-27 Config Block Header.
- [UINT8 RenderStandby](#)
Offset 28 : (Test) This field is used to enable or disable RC6 (Render Standby): 0=FALSE, 1=TRUE
- [UINT8 PmSupport](#)
Offset 29 : (Test) Configure GT to use along with PM Support TRUE/FALSE: 0=FALSE, 1=TRUE
- [UINT16 CdClock](#)
Offset 30 CdClock Frequency select
0xFF = Auto.
- [UINT8 PeiGraphicsPeimInit](#)
Offset 32 : This policy is used to enable/disable Intel Gfx PEIM. 0- Disable, 1- Enable.
- [UINT8 CdynmaxClampEnable](#)
Offset 33 : This policy is used to enable/disable CDynmax Clamping Feature (CCF) 1- Enable, 0- Disable.
- [UINT16 GtFreqMax](#)
Offset 34 : (Test) Max GT frequency limited by user in multiples of 50MHz: Default value which indicates normal frequency is 0xFF
- [UINT8 DisableTurboGt](#)

- Offset 36 : This policy is used to enable/disable DisableTurboGt **0- Disable**, 1- Enable.

 - UINT8 [SkipCdClockInit](#)

Offset 37 : Skip full CD clock initialization. **0- Disable**, 1- Enable.
 - UINT8 [RC1pFreqEnable](#)

Offset 38 : This policy is used to enable/disable RC1p Frequency. **0- Disable**, 1- Enable.
 - UINT8 [PavpEnable](#)

Offset 39 :IGD PAVP TRUE/FALSE: 0=FALSE, 1=TRUE
 - VOID * [LogoPtr](#)

Offset 40 Address of Intel Gfx PEIM Logo to be displayed.
 - UINT32 [LogoSize](#)

Offset 44 Intel Gfx PEIM Logo Size.
 - VOID * [GraphicsConfigPtr](#)

Offset 48 Address of the Graphics Configuration Table.
 - VOID * [BltBufferAddress](#)

Offset 52 Address of Blt buffer for PEIM Logo use.
 - UINT32 [BltBufferSize](#)

Offset 56 The size for Blt Buffer, calculating by PixelWidth * PixelHeight * 4 bytes (the size of EFI_GRAPHICS_OUTPUT_BLT_PIXEL)
 - UINT8 [ProgramGtChickenBits](#)

Offset 60 Program GT Chicket bits in GTTMMADR + 0xD00 BITS [3:1].
 - UINT8 [SkipFspGop](#)

Offset 61 This policy is used to skip PEIM GOP in FSP.**0- Use FSP provided GOP driver**, 1- Skip FSP provided GOP driver.
 - UINT8 [Rsvd1](#) [2]

Offset 62 Reserved for 4 bytes alignment.
 - UINT32 [LogoPixelHeight](#)

Offset 64 Address of LogoPixelHeight for PEIM Logo use.
 - UINT32 [LogoPixelWidth](#)

Offset 68 Address of LogoPixelWidth for PEIM Logo use.
 - UINT32 [VerticalResolution](#)

Offset 72 Address of VerticalResolution for PEIM Logo use.
 - UINT32 [HorizontalResolution](#)

Offset 76 Address of HorizontalResolution for PEIM Logo use.

13.75.1 Detailed Description

This configuration block is to configure IGD related variables used in PostMem PEI.

If Intel Gfx Device is not supported, all policies can be ignored. **Revision 1:**

- Initial version. **Revision 2:**
- Removed DfdRestoreEnable. **Revision 3:**
- Removed DdiConfiguration. **Revision 4:**
- Added new CdClock frequency **Revision 5:**
- Added GT Chicket bits **Revision 6:**
- Added LogoPixelHeight and LogoPixelWidth **Revision 7:**
- Added SkipFspGop **Revision 8:**
- Added VerticalResolution and HorizontalResolution

Definition at line 197 of file GraphicsConfig.h.

13.75.2 Member Data Documentation

13.75.2.1 CdClock

UINT16 GRAPHICS_PEI_CONFIG::CdClock

Offset 30 CdClock Frequency select

0xFF = Auto.

Max CdClock freq based on Reference Clk

0: 192 Mhz, 1: 307.2 Mhz, 2: 312 Mhz, 3: 324 Mhz, 4: 326.4 Mhz, 5: 552 Mhz, 6: 556.8 Mhz, 7: 648 Mhz, 8: 652.8 Mhz

Definition at line 208 of file GraphicsConfig.h.

The documentation for this struct was generated from the following file:

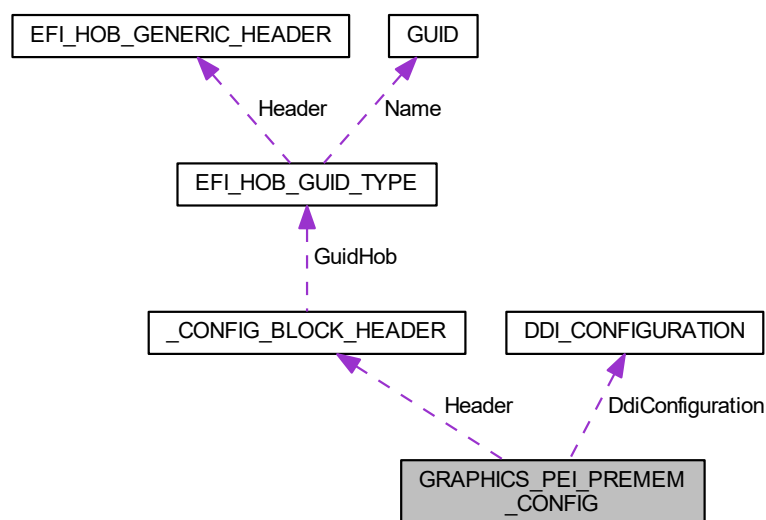
- [GraphicsConfig.h](#)

13.76 GRAPHICS_PEI_PREMEM_CONFIG Struct Reference

This Configuration block is to configure GT related PreMem data/variables.

```
#include <GraphicsConfig.h>
```

Collaboration diagram for GRAPHICS_PEI_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.
- [UINT8 PrimaryDisplay](#)
Offset 28 Selection of the primary display device: 0=iGFX, 1=PEG, 2=PCIe Graphics on PCH, **3=AUTO**, 4=Switchable Graphics
When AUTO mode selected, the priority of display devices is: PCIe Graphics on PCH > PEG > iGFX.
- [UINT8 InternalGraphics](#)
Offset 29 Intel Gfx Support.
- [UINT16 IgdDvmt50PreAlloc](#)
Offset 30 Pre-allocated memory for iGFX
0 = 0MB, 1 or 247 = 32MB,
2 = 64MB,
240 = 4MB, 241 = 8MB,
242 = 12MB, 243 = 16MB,
244 = 20MB, 245 = 24MB,
246 = 28MB, 248 = 36MB,
249 = 40MB, 250 = 44MB,
251 = 48MB, 252 = 52MB,
253 = 56MB, **254 = 60MB**,
Note: enlarging pre-allocated memory for iGFX may need to reduce MmioSize because of 4GB boundary limitation
- [UINT8 PanelPowerEnable](#)
Offset 32 :**(Test)** Control for enabling/disabling VDD force bit (Required only for early enabling of eDP panel): 0=F↔ALSE, 1=**TRUE**
- [UINT8 ApertureSize](#)
Offset 33 :Graphics aperture size (256MB is the recommended size as per BWG) : 0=128MB, **1=256MB**, 3=512MB, 7=1024MB, 15=2048MB.
- [UINT8 GtPsmiSupport](#)
Offset 34 :PSMI support On/Off: **0=FALSE**, 1=TRUE.
- [UINT8 PsmiRegionSize](#)
Offset 35 :Psmi region size: **0=32MB**, 1=288MB, 2=544MB, 3=800MB, 4=1056MB.
- [UINT8 DismSize](#)
Offset 36 :DiSM Size for 2LM Sku: **0=0GB**, 1=1GB, 2=2GB, 3=3GB, 4=4GB, 5=5GB, 6=6GB, 7=7GB.
- [UINT8 DfdRestoreEnable](#)
Offset 37 :**(Deprecated)** Display memory map programming for DFD Restore **0- Disable**, 1- Enable.
- [UINT16 GttSize](#)
Offset 38 :Selection of iGFX GTT Memory size: 1=2MB, 2=4MB, **3=8MB**
- [UINT32 GttMmAdr](#)
Offset 40 Temp Address of System Agent GTTMMADR: Default is **0xAF000000**
- [UINT32 GmAdr](#)
Offset 44 Obsolete not to be used, use GmAdr64.
- [DDI_CONFIGURATION](#) DdiConfiguration
Offset 48 DDI configuration, need to match with VBT settings.
- [UINT8 GtClosEnable](#)
Offset 50 Gt CLOS.
- [UINT8 Rsvd0](#) [7]
Offset 51 Reserved for 4 bytes of alignment.
- [UINT64 GmAdr64](#)
Offset 58 Temp Address of System Agent GMADR: Default is **0xB0000000**
- [UINT16 DeltaT12PowerCycleDelay](#)
Offset 66 Power Cycle Delay required for eDP as per VESA standard.0 - 0 ms, **0xFFFF - Auto calculate to max 500 ms**< >

- UINT8 [DisplayAudioLink](#)
Offset 68 Enable/Disable Display Audio Links in Pre-OS. 0- Disable, 1- Enable.
- UINT8 [OemT12DelayOverride](#)
*Offset 69 :Oem T12 Delay Override **Disable**< >,Enable-Enable T12 Time.*
- UINT8 [Rsvd2](#) [4]
Offset 70 Reserved for 4 bytes of alignment.

13.76.1 Detailed Description

This Configuration block is to configure GT related PreMem data/variables.

Revision 1:

- Initial version. **Revision 2:**
- Added DfdRestoreEnable. **Revision 3:**
- Added DdiConfiguration. **Revision 4:**
- Added GmAdr64 and made GmAdr obsolete **Revision 5:**
- Added DeltaT12PowerCycleDelay **Revision 6:**
- Added DisplayAudioLink **Revision 7:**
- Added OemT12DelayOverride

Definition at line 115 of file GraphicsConfig.h.

13.76.2 Member Data Documentation

13.76.2.1 InternalGraphics

```
UINT8 GRAPHICS_PEI_PREMEM_CONFIG::InternalGraphics
```

Offset 29 Intel Gfx Support.

It controls enabling/disabling iGfx device. When AUTO mode selected, iGFX will be turned off when external graphics detected. If FALSE, all other policies can be ignored. **2 = AUTO**; 0 = FALSE; 1 = TRUE.

Definition at line 132 of file GraphicsConfig.h.

The documentation for this struct was generated from the following file:

- [GraphicsConfig.h](#)

13.77 GUID Struct Reference

128 bit buffer containing a unique identifier value.

```
#include <Base.h>
```

13.77.1 Detailed Description

128 bit buffer containing a unique identifier value.

Unless otherwise specified, aligned on a 64 bit boundary.

Definition at line 221 of file Base.h.

The documentation for this struct was generated from the following file:

- [Base.h](#)

13.78 HDA_LINK_DMIC Struct Reference

HD Audio DMIC Interface Policies.

```
#include <HdAudioConfig.h>
```

Public Attributes

- UINT32 [Enable](#): 1
*HDA DMIC interface enable. When enabled related pins will be switched to native mode: **0: Disable**; 1: Enable.*
- UINT32 [DmicClockSelect](#): 2
*DMIC link clock select: **0: Both**, 1: ClkA, 2: ClkB; default is "Both".*
- HDA_DMIC_PIN_MUX [PinMux](#)
Pin mux configuration.

13.78.1 Detailed Description

HD Audio DMIC Interface Policies.

Definition at line 115 of file HdAudioConfig.h.

The documentation for this struct was generated from the following file:

- [HdAudioConfig.h](#)

13.79 HDA_LINK_HDA Struct Reference

HD Audio Link Policies.

```
#include <HdAudioConfig.h>
```

Public Attributes

- UINT32 [Enable](#): 1
*HDA interface enable. When enabled related pins will be switched to native mode: **0: Disable**; 1: Enable.*
- UINT8 [SdiEnable](#) [PCH_MAX_HDA_SDI]
*HDA SDI signal enable. When enabled related SDI pins will be switched to appropriate native mode: **0: Disable**; 1: Enable.*
- UINT8 [Reserved](#) [(4 -(PCH_MAX_HDA_SDI % 4)) % 4]
Padding for SDI enable table.

13.79.1 Detailed Description

HD Audio Link Policies.

Definition at line 105 of file HdAudioConfig.h.

The documentation for this struct was generated from the following file:

- [HdAudioConfig.h](#)

13.80 HDA_LINK_SNDW Struct Reference

HD Audio SNDW Interface Policies.

```
#include <HdAudioConfig.h>
```

Public Attributes

- UINT32 [Enable](#): 1
*HDA SNDW interface enable. When enabled related pins will be switched to native mode: **0: Disable**; 1: Enable.*

13.80.1 Detailed Description

HD Audio SNDW Interface Policies.

Definition at line 133 of file HdAudioConfig.h.

The documentation for this struct was generated from the following file:

- [HdAudioConfig.h](#)

13.81 HDA_LINK_SSP Struct Reference

HD Audio SSP Interface Policies.

```
#include <HdAudioConfig.h>
```

Public Attributes

- UINT32 [Enable](#): 1
*HDA SSP interface enable. When enabled related pins will be switched to native mode: 0: **Disable**; 1: Enable.*

13.81.1 Detailed Description

HD Audio SSP Interface Policies.

Definition at line 125 of file HdAudioConfig.h.

The documentation for this struct was generated from the following file:

- [HdAudioConfig.h](#)

13.82 HDA_VERB_TABLE_HEADER Struct Reference

Azalia verb table header Every verb table should contain this defined header and followed by azalia verb commands.

```
#include <HdAudioConfig.h>
```

Public Attributes

- UINT16 [VendorId](#)
Codec Vendor ID.
- UINT16 [Deviceld](#)
Codec Device ID.
- UINT8 [RevisionId](#)
Revision ID of the codec. 0xFF matches any revision.
- UINT8 [SdiNum](#)
SDI number, 0xFF matches any SDI.
- UINT16 [DataDwords](#)
Number of data DWORDs following the header.

13.82.1 Detailed Description

Azalia verb table header Every verb table should contain this defined header and followed by azalia verb commands.

Definition at line 72 of file HdAudioConfig.h.

The documentation for this struct was generated from the following file:

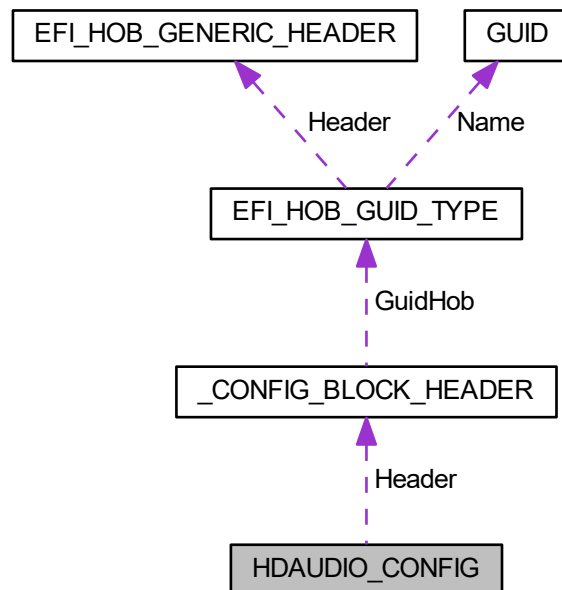
- [HdAudioConfig.h](#)

13.83 HDAUDIO_CONFIG Struct Reference

This structure contains the policies which are related to HD Audio device (cAVS).

```
#include <HdAudioConfig.h>
```

Collaboration diagram for HDAUDIO_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- UINT32 [Pme](#): 1
*Azalia wake-on-ring, 0: **Disable**; 1: Enable.*
- UINT32 [CodecSxWakeCapability](#): 1
*Capability to detect wake initiated by a codec in Sx (eg by modem codec), 0: **Disable**; 1: Enable.*
- UINT32 [HdAudioLinkFrequency](#): 4
*HDA-Link frequency (PCH_HDAUDIO_LINK_FREQUENCY enum): 2: **24MHz**, 1: 12MHz, 0: 6MHz.*
- UINT32 [RsvdBits0](#): 26
Reserved bits 0.
- UINT8 [VerbTableEntryNum](#)
Number of the verb table entry defined in VerbTablePtr.
- UINT8 [Rsvd0](#) [3]
Reserved bytes, align to multiple 4.
- UINT32 [VerbTablePtr](#)
Pointer to a verb table array.

13.83.1 Detailed Description

This structure contains the policies which are related to HD Audio device (cAVS).

Revision 1:

- Initial version.

Definition at line 146 of file HdAudioConfig.h.

13.83.2 Member Data Documentation

13.83.2.1 VerbTableEntryNum

```
UINT8 HDAUDIO_CONFIG::VerbTableEntryNum
```

Number of the verb table entry defined in VerbTablePtr.

Each entry points to a verb table which contains HDAUDIO_VERB_TABLE structure and verb command blocks.

Definition at line 156 of file HdAudioConfig.h.

13.83.2.2 VerbTablePtr

```
UINT32 HDAUDIO_CONFIG::VerbTablePtr
```

Pointer to a verb table array.

This pointer points to 32bits address, and is only eligible and consumed in post mem phase. Each entry points to a verb table which contains HDAUDIO_VERB_TABLE structure and verb command blocks. The prototype of this is: HDAUDIO_VERB_TABLE **VerbTablePtr;

Definition at line 165 of file HdAudioConfig.h.

The documentation for this struct was generated from the following file:

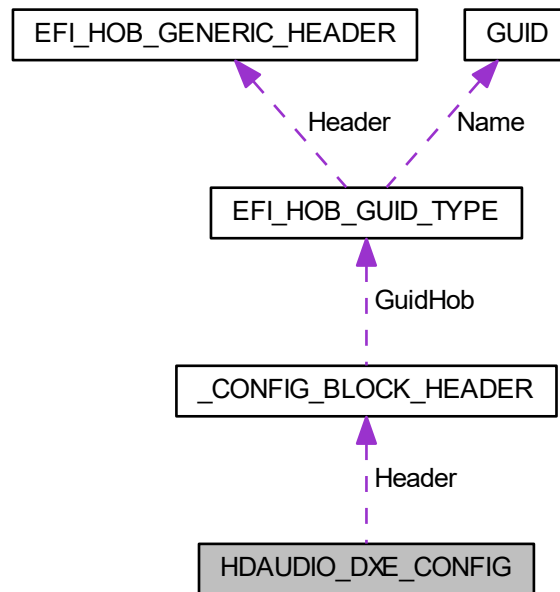
- [HdAudioConfig.h](#)

13.84 HDAUDIO_DXE_CONFIG Struct Reference

This structure contains the DXE policies which are related to HD Audio device (cAVS).

```
#include <HdAudioConfig.h>
```

Collaboration diagram for HDAUDIO_DXE_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- HDAUDIO_SNDW_CONFIG [SndwConfig](#) [PCH_MAX_HDA_SNDW_LINK_NUM]
SNDW configuration for exposed via SNDW ACPI tables:
- UINT32 [DspFeatureMask](#)
*Bitmask of supported DSP features: [BIT0] - WoV; [BIT1] - BT Sideband; [BIT2] - Codec VAD; [BIT5] - BT Intel HFP; [BIT6] - BT Intel A2DP [BIT7] - DSP based speech pre-processing disabled; [BIT8] - 0: Intel WoV, 1: Windows Voice Activation Default is **zero**.*

13.84.1 Detailed Description

This structure contains the DXE policies which are related to HD Audio device (cAVS).

Revision 1:

- Initial version. **Revision 2:**

- Add IoControlEnabled **Revision 3:**
- Remove IoControlEnabled

Definition at line 250 of file HdAudioConfig.h.

The documentation for this struct was generated from the following file:

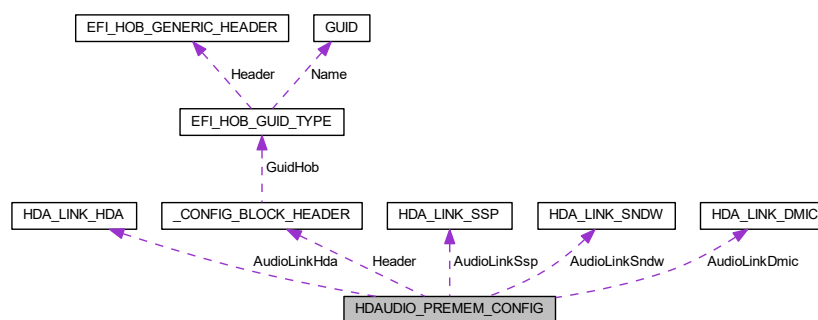
- [HdAudioConfig.h](#)

13.85 HDAUDIO_PREMEM_CONFIG Struct Reference

This structure contains the premem policies which are related to HD Audio device (cAVS).

```
#include <HdAudioConfig.h>
```

Collaboration diagram for HDAUDIO_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [Enable](#): 1
*Intel HD Audio (Azalia) enablement: 0: Disable, 1: **Enable***
- UINT32 [DspEnable](#): 1
*DSP enablement: 0: Disable; 1: **Enable***
- UINT32 [VcType](#): 1
*Virtual Channel Type Select: 0: **VC0**, 1: VC1.*
- UINT32 [DspUaaCompliance](#): 1
*Universal Audio Architecture compliance for DSP enabled system: 0: **Not-UAA Compliant (Intel SST driver supported only)**, 1: UAA Compliant (HDA Inbox driver or SST driver supported)*
- UINT32 [IDispLinkFrequency](#): 4
*iDisp-Link frequency (PCH_HDAUDIO_LINK_FREQUENCY enum): 4: **96MHz**, 3: 48MHz*
- UINT32 [IDispLinkTmode](#): 3
*iDisp-Link T-Mode (PCH_HDAUDIO_IDISP_TMODE enum): 0: 2T, 1: 1T, 2: 4T, 3: **8T**, 4: 16T*
- UINT32 [IDispCodecDisconnect](#): 1

- iDisplay Audio Codec disconnection, 0: **Not disconnected, enumerable**; 1: Disconnected SDI, not enumerable*
- UINT32 [PowerGatingSupported](#): 1
*Power Gating supported: 0: **Not supported**, 1: Supported.*
- UINT32 [RsvdBits](#): 19
Reserved bits 0.
- [HDA_LINK_HDA](#) [AudioLinkHda](#)
Audio Link Mode configuration bitmask.
- [HDA_LINK_DMIC](#) [AudioLinkDmic](#) [PCH_MAX_HDA_DMIC_LINK_NUM]
*DMIC link enablement: 0: Disable; 1: **Enable**.*
- [HDA_LINK_SSP](#) [AudioLinkSsp](#) [PCH_MAX_HDA_SSP_LINK_NUM]
*I2S/SSP link enablement: 0: **Disable**; 1: Enable.*
- [HDA_LINK_SNDW](#) [AudioLinkSndw](#) [PCH_MAX_HDA_SNDW_LINK_NUM]
*SoundWire link enablement: 0: **Disable**; 1: Enable.*
- UINT16 [ResetWaitTimer](#)
*(Test) The delay timer after Azalia reset, the value is number of microseconds. Default is **600**.*
- UINT8 [Rsvd0](#) [2]
Reserved bytes, align to multiple 4.
- UINT32 [SubSystemIds](#)
Value for SID and SVID. If its set to 0 then default value is used.

13.85.1 Detailed Description

This structure contains the premem policies which are related to HD Audio device (cAVS).

Revision 1:

- Initial version. **Revision 2:**
- Add DmicClockSelect **Revision 3:**
- Add AudioFpgaSocVer **Revision 4:**
- Remove AudioFpgaSocVer

Definition at line 180 of file HdAudioConfig.h.

13.85.2 Member Data Documentation

13.85.2.1 AudioLinkDmic

[HDA_LINK_DMIC](#) HDAUDIO_PREMEM_CONFIG::AudioLinkDmic [PCH_MAX_HDA_DMIC_LINK_NUM]

DMIC link enablement: 0: Disable; 1: **Enable**.

DMIC0 LKF: Muxed with SNDW2/SNDW4.

Definition at line 207 of file HdAudioConfig.h.

13.85.2.2 AudioLinkHda

[HDA_LINK_HDA](#) HDAUDIO_PREMEM_CONFIG::AudioLinkHda

Audio Link Mode configuration bitmask.

Allows to configure enablement of the following interfaces: HDA-Link, DMIC, SSP, SoundWire. HDA-Link enablement: 0: Disable; **1: Enable**.

Definition at line 202 of file HdAudioConfig.h.

13.85.2.3 AudioLinkSndw

[HDA_LINK_SNDW](#) HDAUDIO_PREMEM_CONFIG::AudioLinkSndw[PCH_MAX_HDA_SNDW_LINK_NUM]

SoundWire link enablement: **0: Disable**; 1: Enable.

SNDW2 LKF: Muxed with DMIC0/DMIC1. SNDW3 LKF: Muxed with DMIC1. SNDW4 LKF: Muxed with DMIC0.

Definition at line 221 of file HdAudioConfig.h.

13.85.2.4 AudioLinkSsp

[HDA_LINK_SSP](#) HDAUDIO_PREMEM_CONFIG::AudioLinkSsp[PCH_MAX_HDA_SSP_LINK_NUM]

I2S/SSP link enablement: **0: Disable**; 1: Enable.

SSP0/1 LKF: Muxed with HDA.

Note

Since the I2S/SSP2 pin set contains pads which are also used for CNVi purpose, enabling AudioLinkSsp2 is exclusive with CNVi is present.

Definition at line 214 of file HdAudioConfig.h.

The documentation for this struct was generated from the following file:

- [HdAudioConfig.h](#)

13.86 HSIO_PARAMETERS Struct Reference

This structure describes USB3 Port N configuration parameters.

```
#include <Usb3HsioConfig.h>
```

Public Attributes

- UINT8 [HsioTxDownscaleAmp](#)
USB 3.0 TX Output Downscale Amplitude Adjustment (orate01margin) HSIO_TX_DWORD8[21:16] **Default = 00h**
- UINT8 [HsioTxDeEmph](#)
USB 3.0 TX Output -3.5dB De-Emphasis Adjustment Setting (ow2tapgen2deemph3p5) HSIO_TX_DWORD5[21:16] **Default = 29h** (approximately -3.5dB De-Emphasis)
- UINT8 [HsioCtrlAdaptOffsetCfg](#)
Signed Magnatude number added to the CTLE code.
- UINT8 [HsioFilterSelN](#)
LFPS filter select for n (filter_sel_n_2_0) HSIO_RX_DWORD51 [29:27] 0h:1.6ns 1h:2.4ns 2h:3.2ns 3h:4.0ns 4h:4.8ns 5h:5.6ns 6h:6.4ns **Default = 0h**
- UINT8 [HsioFilterSelP](#)
LFPS filter select for p (filter_sel_p_2_0) HSIO_RX_DWORD51 [26:24] 0h:1.6ns 1h:2.4ns 2h:3.2ns 3h:4.0ns 4h:4.8ns 5h:5.6ns 6h:6.4ns **Default = 0h**
- UINT8 [HsioOlfpsCfgPullUpDwnRes](#)
Controls the input offset (olfpscfgpullupdwnres_sus_usb_2_0) HSIO_RX_DWORD51 [2:0] 000 Prohibited 001 45K 010 Prohibited 011 31K 100 36K 101 36K 110 36K 111 36K **Default = 3h**
- UINT8 [HsioTxDeEmphEnable](#)
Enable the write to USB 3.0 TX Output -3.5dB De-Emphasis Adjustment, **0: Disable**; 1: Enable.
- UINT8 [HsioTxDownscaleAmpEnable](#)
Enable the write to USB 3.0 TX Output Downscale Amplitude Adjustment, **0: Disable**; 1: Enable.
- UINT8 [HsioCtrlAdaptOffsetCfgEnable](#)
Enable the write to Signed Magnatude number added to the CTLE code, **0: Disable**; 1: Enable.
- UINT8 [HsioFilterSelNEnable](#)
Enable the write to LFPS filter select for n, **0: Disable**; 1: Enable.
- UINT8 [HsioFilterSelPEnable](#)
Enable the write to LFPS filter select for p, **0: Disable**; 1: Enable.
- UINT8 [HsioOlfpsCfgPullUpDwnResEnable](#)
Enable the write to olfpscfgpullupdwnres, **0: Disable**; 1: Enable.
- UINT8 [HsioTxRate3UniqTran](#)
USB 3.0 TX Output - Unique Transition Bit Scale for rate 3 (rate3UniqTranScale) HSIO_TX_DWORD9[6:0] **Default = 4Ch**
- UINT8 [HsioTxRate2UniqTran](#)
USB 3.0 TX Output -Unique Transition Bit Scale for rate 2 (rate2UniqTranScale) HSIO_TX_DWORD9[14:8] **Default = 4Ch**
- UINT8 [HsioTxRate1UniqTran](#)
USB 3.0 TX Output - Unique Transition Bit Scale for rate 1 (rate1UniqTranScale) HSIO_TX_DWORD9[22:16] **Default = 4Ch**
- UINT8 [HsioTxRate0UniqTran](#)
USB 3.0 TX Output - Unique Transition Bit Scale for rate 0 (rate0UniqTranScale) HSIO_TX_DWORD9[30:24] **Default = 4Ch**
- UINT8 [HsioTxRate3UniqTranEnable](#)
Enable the write to USB 3.0 TX Unique Transition Bit Mode for rate 3, **0: Disable**; 1: Enable.
- UINT8 [HsioTxRate2UniqTranEnable](#)
Enable the write to USB 3.0 TX Unique Transition Bit Mode for rate 2, **0: Disable**; 1: Enable.
- UINT8 [HsioTxRate1UniqTranEnable](#)
Enable the write to USB 3.0 TX Unique Transition Bit Mode for rate 1, **0: Disable**; 1: Enable.
- UINT8 [HsioTxRate0UniqTranEnable](#)
Enable the write to USB 3.0 TX Unique Transition Bit Mode for rate 0, **0: Disable**; 1: Enable.

13.86.1 Detailed Description

This structure describes USB3 Port N configuration parameters.

Definition at line 49 of file Usb3HsioConfig.h.

13.86.2 Member Data Documentation

13.86.2.1 HsioCtrlAdaptOffsetCfg

```
UINT8 HSIO_PARAMETERS::HsioCtrlAdaptOffsetCfg
```

Signed Magnatude number added to the CTLE code.

(ctle_adapt_offset_cfg_4_0) HSIO_RX_DWORD25 [20:16] Ex: -1 – 1_0001. +1: 0_0001 **Default = 0h**

Definition at line 68 of file Usb3HsioConfig.h.

The documentation for this struct was generated from the following file:

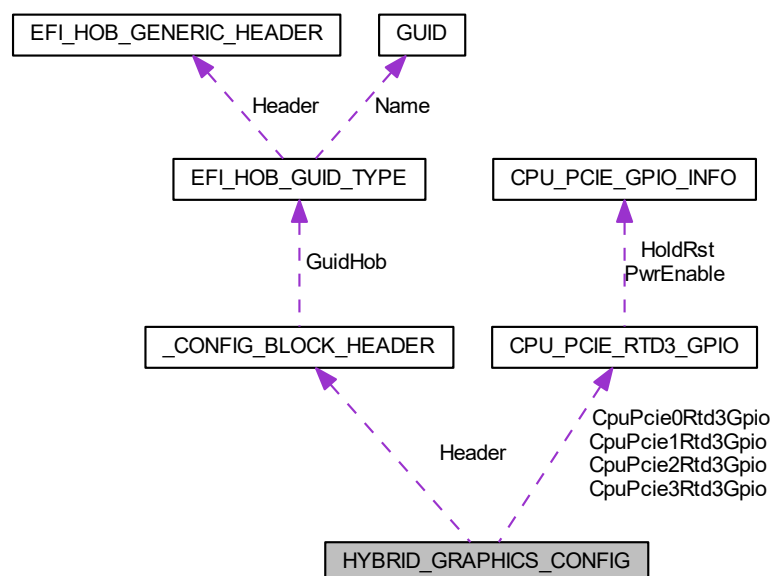
- [Usb3HsioConfig.h](#)

13.87 HYBRID_GRAPHICS_CONFIG Struct Reference

This Configuration block configures CPU PCI Express 0/1/2 RTD3 GPIOs & Root Port.

```
#include <HybridGraphicsConfig.h>
```

Collaboration diagram for HYBRID_GRAPHICS_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.
- [CPU_PCIE_RTD3_GPIO](#) CpuPcie0Rtd3Gpio
Offset 28 RTD3 GPIOs used for PCIe.
- UINT8 [RootPortIndex](#)
Offset 52 Root Port Index number used for HG.
- UINT8 [HgMode](#)
*Offset 53 HgMode: **0=Disabled**, 1=HG Muxed, 2=HG Muxless, 3=PEG.*
- UINT16 [HgSubSystemId](#)
*Offset 54 Hybrid Graphics Subsystem ID: **2212***
- UINT16 [HgDelayAfterPwrEn](#)
*Offset 56 Dgpu Delay after Power enable using Setup option: 0=Minimal, 1000=Maximum, **300=300 microseconds***
- UINT16 [HgDelayAfterHoldReset](#)
*Offset 58 Dgpu Delay after Hold Reset using Setup option: 0=Minimal, 1000=Maximum, **100=100 microseconds***
- [CPU_PCIE_RTD3_GPIO](#) CpuPcie1Rtd3Gpio
Offset 60 RTD3 GPIOs used for PCIe.
- [CPU_PCIE_RTD3_GPIO](#) CpuPcie2Rtd3Gpio
Offset 84 RTD3 GPIOs used for PCIe.
- [CPU_PCIE_RTD3_GPIO](#) CpuPcie3Rtd3Gpio
Offset 108 RTD3 GPIOs used for PCIe.
- UINT8 [HgSupport](#)
Offset 132 HgSupport on specific port.
- UINT8 [Rsvd0](#) [3]
Offset 133 Reserved bytes for future use.

13.87.1 Detailed Description

This Configuration block configures CPU PCI Express 0/1/2 RTD3 GPIOs & Root Port.

Hybrid Gfx uses the same GPIOs & Root port as PCI Express 0/1/2 RTD3. **Revision 1:**

- Initial version. **Revision 2:**
- Add HgSlot Policy: PEG or PCH Slot Slection for Hybrid Graphics **Revision 3:**
- Removed HgSlot Policy: As there is no selection between PEG or PCH Slot for Hybrid Graphics **Revision 4:**
- Added HgSupport Policy: To support HG support on other PEG ports

Definition at line 83 of file HybridGraphicsConfig.h.

The documentation for this struct was generated from the following file:

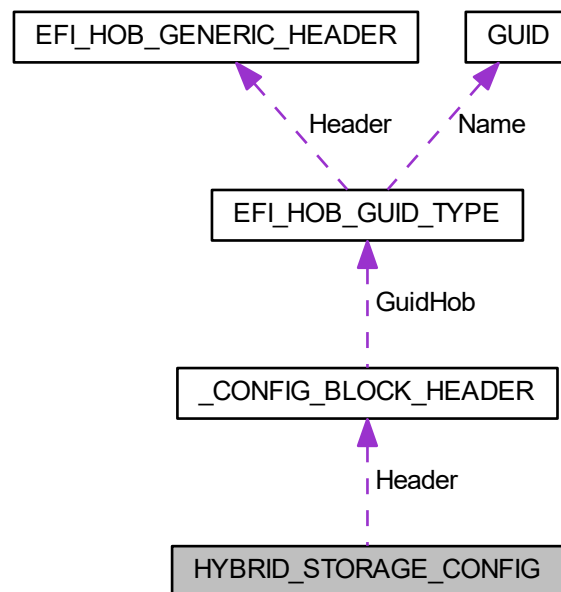
- [HybridGraphicsConfig.h](#)

13.88 HYBRID_STORAGE_CONFIG Struct Reference

The [HYBRID_STORAGE_CONFIG](#) block describes the expected configuration for Hybrid Storage device.

```
#include <HybridStorageConfig.h>
```

Collaboration diagram for HYBRID_STORAGE_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- `UINT8` [HybridStorageMode](#)
*Hybrid Storage Mode **0**: **Disable**, 1: Enable Dynamic Configuration.*

13.88.1 Detailed Description

The [HYBRID_STORAGE_CONFIG](#) block describes the expected configuration for Hybrid Storage device.

Revision 1:

- Init version **Revision 2**:
- Added support for cpu attached hybrid storage

Definition at line 54 of file `HybridStorageConfig.h`.

The documentation for this struct was generated from the following file:

- [HybridStorageConfig.h](#)

13.89 I2C_PIN_MUX Struct Reference

I2C signals pin muxing settings.

```
#include <SerialIoDevices.h>
```

Public Attributes

- [UINT32 Sda](#)
*SDA Pin mux configuration. Refer to GPIO_*_MUXING_SERIALIO_I2Cx_SDA_*.*
- [UINT32 Scl](#)
*SCL Pin mux configuration. Refer to GPIO_*_MUXING_SERIALIO_I2Cx_SCL_*.*

13.89.1 Detailed Description

I2C signals pin muxing settings.

If signal can be enable only on a single pin then this parameter is ignored by RC. Refer to GPIO_*_MUXING_SERIALIO_I2Cx_* in GpioPins*.h for supported settings on a given platform

Definition at line 228 of file SerialIoDevices.h.

The documentation for this struct was generated from the following file:

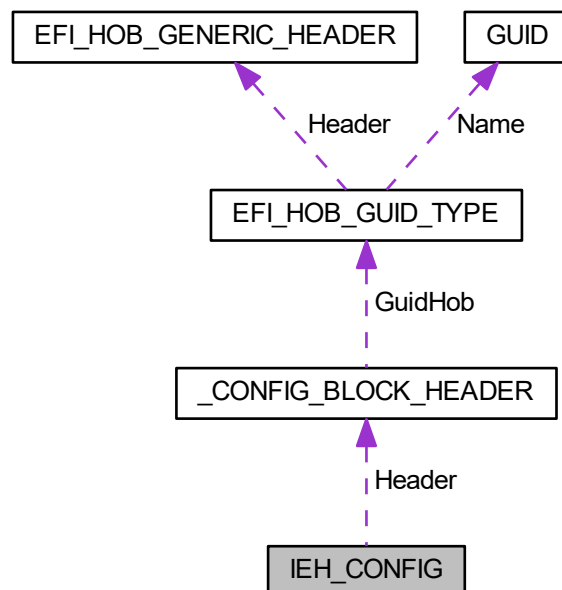
- [SerialIoDevices.h](#)

13.90 IEH_CONFIG Struct Reference

The [IEH_CONFIG](#) block describes the expected configuration of the PCH Integrated Error Handler.

```
#include <IehConfig.h>
```

Collaboration diagram for IEH_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [Mode](#): 1
*IEH mode 0: **Bypass Mode**; 1: Enable.*
- UINT32 [RsvdBits0](#): 31
Reserved bits.

13.90.1 Detailed Description

The [IEH_CONFIG](#) block describes the expected configuration of the PCH Integrated Error Handler.

Definition at line 53 of file `lehConfig.h`.

The documentation for this struct was generated from the following file:

- [lehConfig.h](#)

13.91 IOM_AUX_ORI_PAD_CONFIG Struct Reference

The [IOM_AUX_ORI_PAD_CONFIG](#) describes IOM TypeC port map GPIO pin.

```
#include <TcssPeiConfig.h>
```

Public Attributes

- UINT32 [GpioPullN](#)
GPIO Pull Up Ping number that is for IOM indicate the pull up pin from TypeC port.
- UINT32 [GpioPullP](#)
GPIO Pull Down Ping number that is for IOM indicate the pull down pin from TypeC port.

13.91.1 Detailed Description

The [IOM_AUX_ORI_PAD_CONFIG](#) describes IOM TypeC port map GPIO pin.

Those GPIO setting for DP Aux Orientation Bias Control when the TypeC port didn't have re-timer. IOM needs know Pull-Up and Pull-Down pin for Bias control

Definition at line 55 of file `TcssPeiConfig.h`.

The documentation for this struct was generated from the following file:

- [TcssPeiConfig.h](#)

13.92 IOM_INTERFACE_CONFIG Struct Reference

The IOM_EC_INTERFACE_CONFIG block describes interaction between BIOS and IOM-EC.

```
#include <TcssPeiConfig.h>
```

Public Attributes

- UINT32 [VccSt](#)
IOM VCCST request. (Not equal to actual VCCST value)
- UINT32 [UsbOverride](#)
IOM to override USB connection.
- UINT32 [D3ColdEnable](#)
Enable/disable D3 Cold support in TCSS.
- UINT32 [D3HotEnable](#)
Enable/disable D3 Hot support in TCSS.

13.92.1 Detailed Description

The IOM_EC_INTERFACE_CONFIG block describes interaction between BIOS and IOM-EC.

Definition at line 64 of file TcssPeiConfig.h.

The documentation for this struct was generated from the following file:

- [TcssPeiConfig.h](#)

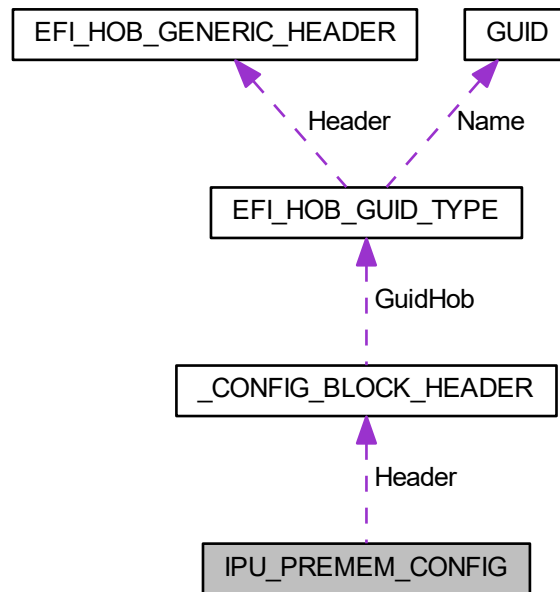
13.93 IPU_PREMEM_CONFIG Struct Reference

IPU PreMem configuration

Revision 1:

```
#include <IpuPreMemConfig.h>
```

Collaboration diagram for IPU_PREMEM_CONFIG:



Public Attributes

- `UINT8 IpuEnable`
Config Block Header.
- `UINT8 IpuImrConfiguration`
Deprecate.
- `UINT8 ImgUclkOutEn` [`GPIO_IMGUCLK_NUMBER_OF_PINS`]
It enable the IMGU CLKOUT.
- `UINT8 LaneUsed` [`MAX_CSI_PORT`]
*Indicate laneUsed of each CSI port 0: **Not configured**; 1: x1; 2:x2; 3:x3; 4:x4.*
- `UINT8 CsiSpeed` [`MAX_CSI_PORT`]
*Indicate speed of each CSI port 0: **Sensor default**; 1: <416Mbps; 2:<1.5Gbps; 3:<2Gbps; 4:<2.5Gbps; 5:<4Gbps; 6>4Gbps.*

13.93.1 Detailed Description

IPU PreMem configuration

Revision 1:

- Initial version. **Revision 2:**
- Change Bit-wise to Byte-wise. **Revision 3:**
- Add LaneUsed and Speed of each Port configuration for Mcd10 support. **Revision 4:**
- Change `GPIO_IMGUCLK_NUMBER_OF_PINS` to 6 **Revision 5:**
- Deprecate `IpuImrConfiguration`.

Definition at line 70 of file `IpuPreMemConfig.h`.

13.93.2 Member Data Documentation

13.93.2.1 ImgUClkOutEn

```
UINT8 IPU_PREMEM_CONFIG::ImgUClkOutEn[GPIO_IMGUCLK_NUMBER_OF_PINS]
```

It enable the IMGU CLKOUT.

TRUE **FALSE**

Definition at line 88 of file IpuPreMemConfig.h.

13.93.2.2 IpuEnable

```
UINT8 IPU_PREMEM_CONFIG::IpuEnable
```

Config Block Header.

(Test) It enables the SA IPU Device if supported and not fused off. If **FALSE**, all other policies in this config block will be ignored. **1=TRUE**; **0=FALSE**.

Definition at line 78 of file IpuPreMemConfig.h.

The documentation for this struct was generated from the following file:

- [IpuPreMemConfig.h](#)

13.94 IPv4_ADDRESS Struct Reference

4-byte buffer.

```
#include <Base.h>
```

13.94.1 Detailed Description

4-byte buffer.

An IPv4 internet protocol address.

Definition at line 231 of file Base.h.

The documentation for this struct was generated from the following file:

- [Base.h](#)

13.95 IPv6_ADDRESS Struct Reference

16-byte buffer.

```
#include <Base.h>
```

13.95.1 Detailed Description

16-byte buffer.

An IPv6 internet protocol address.

Definition at line 238 of file Base.h.

The documentation for this struct was generated from the following file:

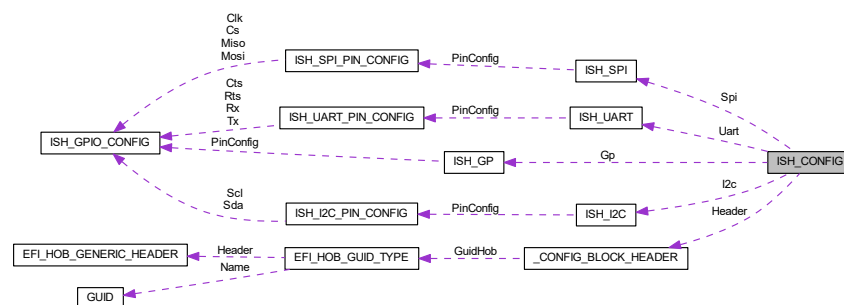
- [Base.h](#)

13.96 ISH_CONFIG Struct Reference

The [ISH_CONFIG](#) block describes Integrated Sensor Hub device.

```
#include <IshConfig.h>
```

Collaboration diagram for ISH_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [PdtUnlock](#): 1
ISH PDT Unlock Msg: 0: False 1: True.
- UINT32 [MsiInterrupt](#): 1
ISH MSI Interrupts: 0: False 1: True.
- UINT32 [RsvdBits0](#): 30
Reserved Bits.

13.96.1 Detailed Description

The [ISH_CONFIG](#) block describes Integrated Sensor Hub device.

Definition at line 137 of file `IshConfig.h`.

The documentation for this struct was generated from the following file:

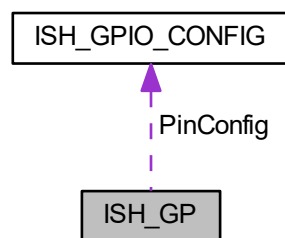
- [IshConfig.h](#)

13.97 ISH_GP Struct Reference

Struct contains GPIO pins assigned and signal settings of GP.

```
#include <IshConfig.h>
```

Collaboration diagram for ISH_GP:



Public Attributes

- `UINT32` [Enable](#): 1
*ISH GP GPIO pins assigned: **0: False** 1: True.*
- `UINT32` [RsvdBits0](#): 31
Reserved Bits.

13.97.1 Detailed Description

Struct contains GPIO pins assigned and signal settings of GP.

Definition at line 128 of file `IshConfig.h`.

The documentation for this struct was generated from the following file:

- [IshConfig.h](#)

13.98 ISH_GPIO_CONFIG Struct Reference

ISH GPIO settings.

```
#include <IshConfig.h>
```

Public Attributes

- UINT32 [PinMux](#)
GPIO signals pin muxing settings.
- UINT32 [PadTermination](#)
GPIO Pads Internal Termination.

13.98.1 Detailed Description

ISH GPIO settings.

Definition at line 50 of file IshConfig.h.

13.98.2 Member Data Documentation

13.98.2.1 PadTermination

```
UINT32 ISH_GPIO_CONFIG::PadTermination
```

GPIO Pads Internal Termination.

For more information please see Platform Design Guide. Check GPIO_ELECTRICAL_CONFIG for reference

Definition at line 62 of file IshConfig.h.

13.98.2.2 PinMux

```
UINT32 ISH_GPIO_CONFIG::PinMux
```

GPIO signals pin muxing settings.

If signal can be enable only on a single pin then this parameter should be set to 0. Refer to GPIO_*_MUXI↔NG_ISH_*_* in GpioPins*.h for supported settings on a given platform GPIO Pin mux configuration. Refer to GPIO_*_MUXING_ISH_*_*_MOSI_*

Definition at line 56 of file IshConfig.h.

The documentation for this struct was generated from the following file:

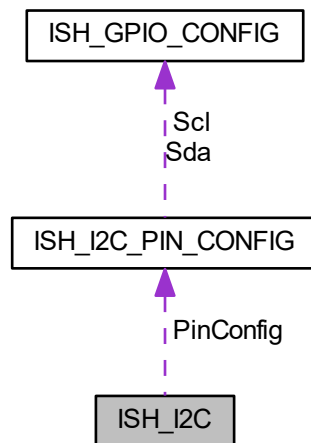
- [IshConfig.h](#)

13.99 ISH_I2C Struct Reference

Struct contains GPIO pins assigned and signal settings of I2C.

```
#include <IshConfig.h>
```

Collaboration diagram for ISH_I2C:



Public Attributes

- UINT32 [Enable](#): 1
*ISH I2C GPIO pins assigned: 0: **False** 1: **True**.*
- UINT32 [RsvdBits0](#): 31
Reserved Bits.

13.99.1 Detailed Description

Struct contains GPIO pins assigned and signal settings of I2C.

Definition at line 119 of file IshConfig.h.

The documentation for this struct was generated from the following file:

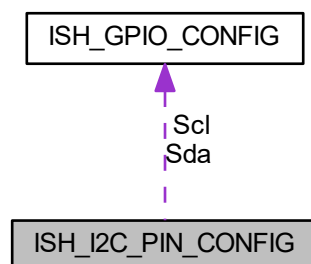
- [IshConfig.h](#)

13.100 ISH_I2C_PIN_CONFIG Struct Reference

I2C signals settings.

```
#include <IshConfig.h>
```

Collaboration diagram for ISH_I2C_PIN_CONFIG:



Public Attributes

- [ISH_GPIO_CONFIG Sda](#)
SDA Pin configuration.
- [ISH_GPIO_CONFIG Scl](#)
SCL Pin configuration.

13.100.1 Detailed Description

I2C signals settings.

Definition at line 90 of file `IshConfig.h`.

The documentation for this struct was generated from the following file:

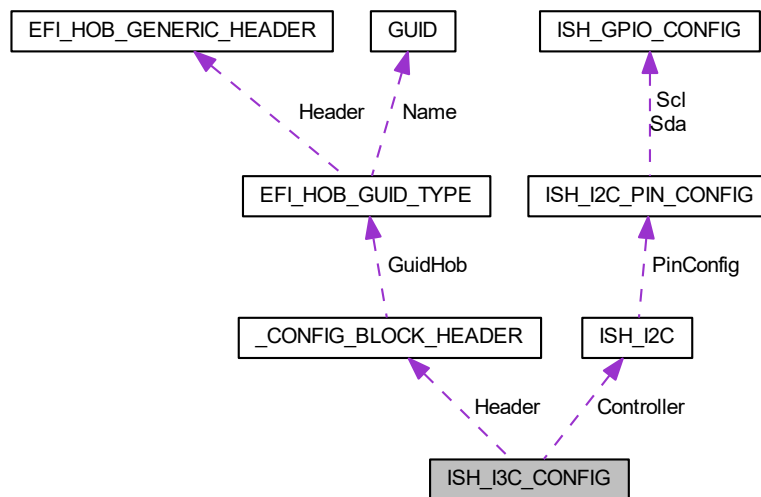
- [IshConfig.h](#)

13.101 ISH_I3C_CONFIG Struct Reference

The [ISH_I3C_CONFIG](#) block describes I3C in ISH device.

```
#include <IshConfig.h>
```

Collaboration diagram for ISH_I3C_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [ISH_I3C](#) Controller
ISH I3C Controller.

13.101.1 Detailed Description

The [ISH_I3C_CONFIG](#) block describes I3C in ISH device.

Definition at line 154 of file `IshConfig.h`.

The documentation for this struct was generated from the following file:

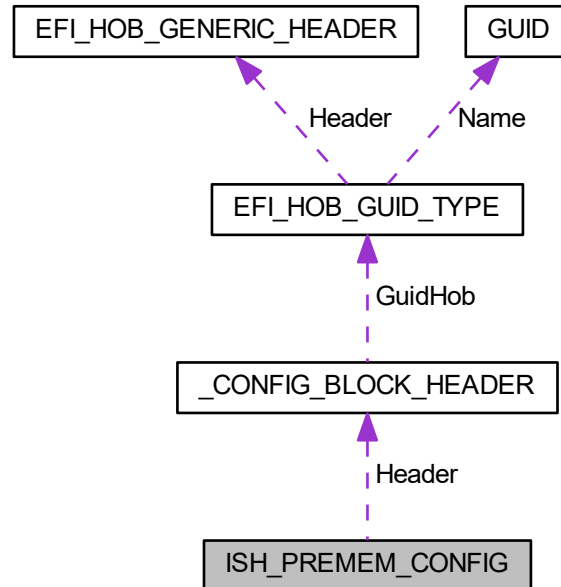
- [IshConfig.h](#)

13.102 ISH_PREMEM_CONFIG Struct Reference

Premem Policy for Integrated Sensor Hub device.

```
#include <IshConfig.h>
```

Collaboration diagram for ISH_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- UINT32 [Enable](#): 1
*ISH Controler 0: Disable; **1: Enable**.*
- UINT32 [RsvdBits0](#): 31
Reserved Bits.

13.102.1 Detailed Description

Premem Policy for Integrated Sensor Hub device.

Definition at line 163 of file IshConfig.h.

13.102.2 Member Data Documentation

13.102.2.1 Enable

```
UINT32 ISH_PREMEM_CONFIG::Enable
```

ISH Controler 0: Disable; **1: Enable**.

For Desktop sku, the ISH POR should be disabled. **0:Disable** .

Definition at line 169 of file IshConfig.h.

The documentation for this struct was generated from the following file:

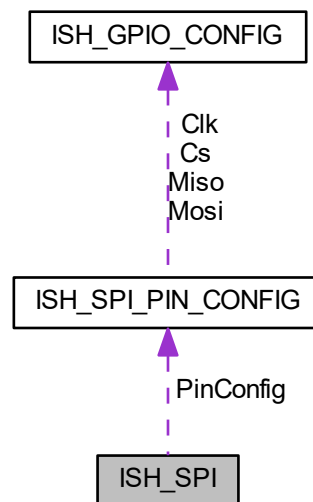
- [IshConfig.h](#)

13.103 ISH_SPI Struct Reference

Struct contains GPIO pins assigned and signal settings of SPI.

```
#include <IshConfig.h>
```

Collaboration diagram for ISH_SPI:



Public Attributes

- UINT8 [Enable](#)
*ISH SPI GPIO pins assigned: **0: False** 1: True.*
- UINT8 [CsEnable](#) [PCH_MAX_ISH_SPI_CS_PINS]
*ISH SPI CS pins assigned: **0: False** 1: True.*
- UINT16 [RsvdField0](#)
Reserved field.

13.103.1 Detailed Description

Struct contains GPIO pins assigned and signal settings of SPI.

Definition at line 99 of file IshConfig.h.

The documentation for this struct was generated from the following file:

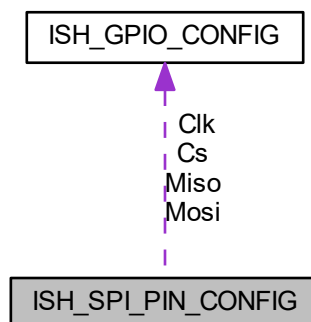
- [IshConfig.h](#)

13.104 ISH_SPI_PIN_CONFIG Struct Reference

SPI signals settings.

```
#include <IshConfig.h>
```

Collaboration diagram for ISH_SPI_PIN_CONFIG:



Public Attributes

- [ISH_GPIO_CONFIG Mosi](#)
MOSI Pin configuration.
- [ISH_GPIO_CONFIG Miso](#)
MISO Pin configuration.
- [ISH_GPIO_CONFIG Clk](#)
CLK Pin configuration.
- [ISH_GPIO_CONFIG Cs](#) [PCH_MAX_ISH_SPI_CS_PINS]
CS Pin configuration.

13.104.1 Detailed Description

SPI signals settings.

Definition at line 68 of file IshConfig.h.

The documentation for this struct was generated from the following file:

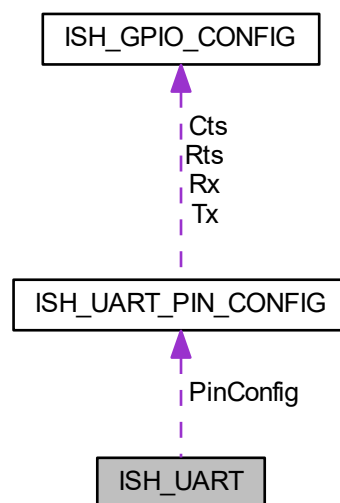
- [IshConfig.h](#)

13.105 ISH_UART Struct Reference

Struct contains GPIO pins assigned and signal settings of UART.

```
#include <IshConfig.h>
```

Collaboration diagram for ISH_UART:



Public Attributes

- UINT32 [Enable](#): 1
*ISH UART GPIO pins assigned: 0: **False** 1: **True**.*
- UINT32 [RsvdBits0](#): 31
Reserved Bits.

13.105.1 Detailed Description

Struct contains GPIO pins assigned and signal settings of UART.

Definition at line 110 of file IshConfig.h.

The documentation for this struct was generated from the following file:

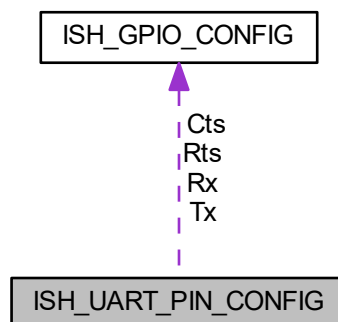
- [IshConfig.h](#)

13.106 ISH_UART_PIN_CONFIG Struct Reference

UART signals settings.

```
#include <IshConfig.h>
```

Collaboration diagram for ISH_UART_PIN_CONFIG:



Public Attributes

- [ISH_GPIO_CONFIG Rx](#)
RXD Pin configuration.
- [ISH_GPIO_CONFIG Tx](#)
TXD Pin configuration.
- [ISH_GPIO_CONFIG Rts](#)
RTS Pin configuration.
- [ISH_GPIO_CONFIG Cts](#)
CTS Pin configuration.

13.106.1 Detailed Description

UART signals settings.

Definition at line 79 of file `IshConfig.h`.

The documentation for this struct was generated from the following file:

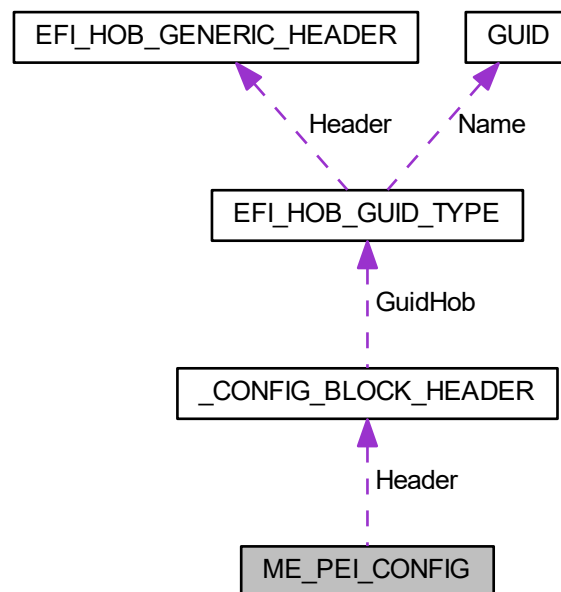
- [IshConfig.h](#)

13.107 ME_PEI_CONFIG Struct Reference

ME Pei Post-Memory Configuration Structure.

```
#include <MePeiConfig.h>
```

Collaboration diagram for ME_PEI_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [EndOfPostMessage](#): 2
*0: Disabled; 1: Send in PEI; 2: **Send in DXE** - Send EOP at specific phase.*
- UINT32 [Heci3Enabled](#): 1

- UINT32 [DisableD0I3SettingForHeci](#): 1
(Test) 0: Disable; 1: Enable - Enable/Disable D0i3 for HECI.
- UINT32 [MeUnconfigOnRtcClear](#): 2
Enable/Disable Me Unconfig On Rtc Clear.
- UINT32 [MctpBroadcastCycle](#): 1
(Test) 0: Disable; 1: Enable - Program registers for MCTP Cycle.
- UINT32 [EnforceEDebugMode](#): 1
0: Disable; 1: Enable - Enforces ME to enter Enhanced Debug Mode
- UINT32 [CseDataResilience](#): 1
*0: Disable; 1: **Enable** - CSE data resilience support*
- UINT32 [RsvdBits](#): 23
Reserved for future use & Config block alignment.

13.107.1 Detailed Description

ME Pei Post-Memory Configuration Structure.

Revision 1:

- Initial version. **Revision 2:**
- Deprecated Heci3Enabled. **Revision 3**
- Added EnforceEDebugMode. **Revision 4**
- Added CseDataResilience.

Definition at line 128 of file MePeiConfig.h.

13.107.2 Member Data Documentation

13.107.2.1 Heci3Enabled

```
UINT32 ME_PEI_CONFIG::Heci3Enabled
```

Deprecated

Definition at line 132 of file MePeiConfig.h.

13.107.2.2 MeUnconfigOnRtcClear

```
UINT32 ME_PEI_CONFIG::MeUnconfigOnRtcClear
```

Enable/Disable Me Unconfig On Rtc Clear.

If enabled, BIOS will send MeUnconfigOnRtcClearDisable Msg with parameter 0. It will cause ME to unconfig if RTC is cleared.

- 0: Disable
- **1: Enable**
- 2: Cmos is clear, status unknown
- 3: Reserved

Definition at line 142 of file MePeiConfig.h.

The documentation for this struct was generated from the following file:

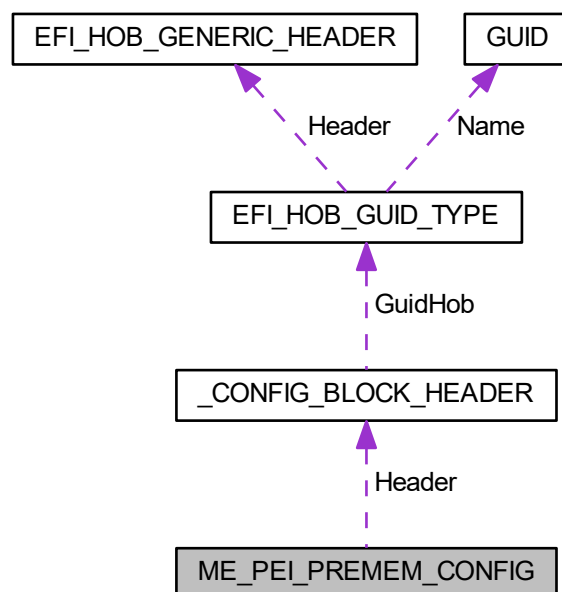
- [MePeiConfig.h](#)

13.108 ME_PEI_PREMEM_CONFIG Struct Reference

ME Pei Pre-Memory Configuration Structure.

```
#include <MePeiConfig.h>
```

Collaboration diagram for ME_PEI_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [HeciTimeouts](#): 1
*0: Disable; 1: **Enable** - HECI Send/Receive Timeouts.*
- UINT32 [DidInitStat](#): 2
*(Test) 0: **Disabled** 1: ME DID init stat 0 - Success 2: ME DID init stat 1 - No Memory in Channels 3: ME DID init stat 2 - Memory Init Error*
- UINT32 [DisableCpuReplacedPolling](#): 1
(Test) 0: Set to 0 to enable polling for CPU replacement 1: Set to 1 will disable polling for CPU replacement
- UINT32 [SendDidMsg](#): 1
*(Deprecated) 0: Disable; 1: **Enable** - Enable/Disable to send DID message.*
- UINT32 [DisableMessageCheck](#): 1
(Test) 0: ME BIOS will check each messages before sending 1: ME BIOS always sends messages without checking
- UINT32 [SkipMbpHob](#): 1
(Test) The SkipMbpHob policy determines whether ME BIOS Payload data will be requested during boot in a MBP message.
- UINT32 [HeciCommunication2](#): 1
*(Test) 0: **Disable**; 1: **Enable** - Enable/Disable HECI2.*
- UINT32 [KtDeviceEnable](#): 1
*(Test) 0: Disable; 1: **Enable** - Enable/Disable Kt Device.*
- UINT32 [SkipCpuReplacementCheck](#): 1
*(Test) 0: **Disable**; 1: **Enable** - Enable/Disable to skip CPU replacement check.*
- UINT32 [HeciFullTrace](#): 1
*(Test) 0: Disable; 1: **Enable** - Enable/Disable Full HECI communication information logged to serial*
- UINT32 [RsvdBits](#): 21
Reserved for future use & Config block alignment.
- UINT32 [Heci1BarAddress](#)
HECI1 BAR address.
- UINT32 [Heci2BarAddress](#)
HECI2 BAR address.
- UINT32 [Heci3BarAddress](#)
HECI3 BAR address.

13.108.1 Detailed Description

ME Pei Pre-Memory Configuration Structure.

Revision 1:

- Initial version. **Revision 2:**
- Add SkipCpuReplacementCheck Option. **Revision 3:**
- Deprecate SendDidMsg. **Revision 4:**
- Add HeciFullTrace Option.

Definition at line 65 of file MePeiConfig.h.

13.108.2 Member Data Documentation

13.108.2.1 SkipMbpHob

```
UINT32 ME_PEI_PREMEM_CONFIG::SkipMbpHob
```

(Test) The SkipMbpHob policy determines whether ME BIOS Payload data will be requested during boot in a MBP message.

If set to 1, BIOS will send the MBP message with SkipMbp flag set causing CSME to respond with MKHI header only and no MBP data

0: ME BIOS will keep MBP and create HOB for MBP data	1: ME BIOS will skip MBP data
<input type="checkbox"/>	<input type="checkbox"/>

Definition at line 97 of file MePeiConfig.h.

The documentation for this struct was generated from the following file:

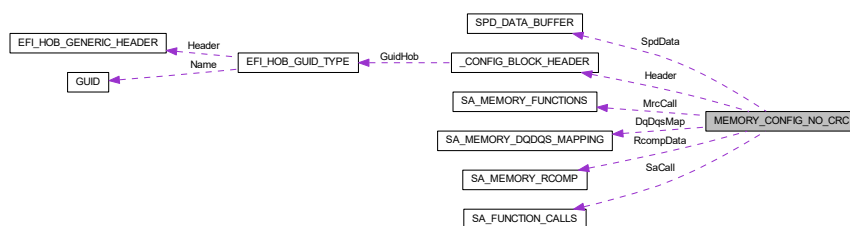
- MePeiConfig.h

13.109 MEMORY_CONFIG_NO_CRC Struct Reference

Memory Configuration The contents of this structure are not CRC'd by the MRC for option change detection.

```
#include <MemoryConfig.h>
```

Collaboration diagram for MEMORY_CONFIG_NO_CRC:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.
- [SA_FUNCTION_CALLS](#) SaCall
Offset 28 Function calls into the SA.
- [SA_MEMORY_FUNCTIONS](#) MrcCall
Offset 212 Function calls into the MRC.
- [SPD_DATA_BUFFER](#) * SpdData
Offset 236 Memory SPD data, will be used by the MRC when SPD SmBus address is zero.
- [SA_MEMORY_DQDQS_MAPPING](#) * DqDqsMap
Offset 240 LPDDR DQ bit and DQS byte swizzling between CPU and DRAM.
- [SA_MEMORY_RCOMP](#) * RcompData
Offset 244 DDR RCOMP resistors and target values.
- UINT64 [PlatformMemorySize](#)
Offset 248 The minimum platform memory size required to pass control into DXE.
- UINT32 [CleanMemory](#):1
*Offset 256 Ask MRC to clear memory content: **FALSE=Do not Clear Memory**; **TRUE=Clear Memory**.*
- UINT8 [SerialDebugLevel](#)
Sets the serial debug message level
0x00 = Disabled
0x01 = Errors only
0x02 = Errors and Warnings
0x03 = Errors, Warnings, and Info
0x04 = Errors, Warnings, Info, and Events
0x05 = Displays Memory Init Execution Time Summary only
.
- UINT8 [MemTestOnWarmBoot](#)
*Offset 261 Run Base Memory Test On WarmBoot: 0=Disabled, **1=Enabled***
- UINT8 [Reserved11](#) [2]
Offset 262 - 263 Reserved.

13.109.1 Detailed Description

Memory Configuration The contents of this structure are not CRC'd by the MRC for option change detection.

Revision 1: - Initial version. **Revision 2:** - Added MemTestOnWarmBoot

Definition at line 579 of file MemoryConfig.h.

13.109.2 Member Data Documentation

13.109.2.1 SerialDebugLevel

```
UINT8 MEMORY_CONFIG_NO_CRC::SerialDebugLevel
```

Sets the serial debug message level

0x00 = Disabled

0x01 = Errors only

0x02 = Errors and Warnings

0x03 = Errors, Warnings, and Info

0x04 = Errors, Warnings, Info, and Events

0x05 = Displays Memory Init Execution Time Summary only

.

Offset 260

Definition at line 598 of file MemoryConfig.h.

The documentation for this struct was generated from the following file:

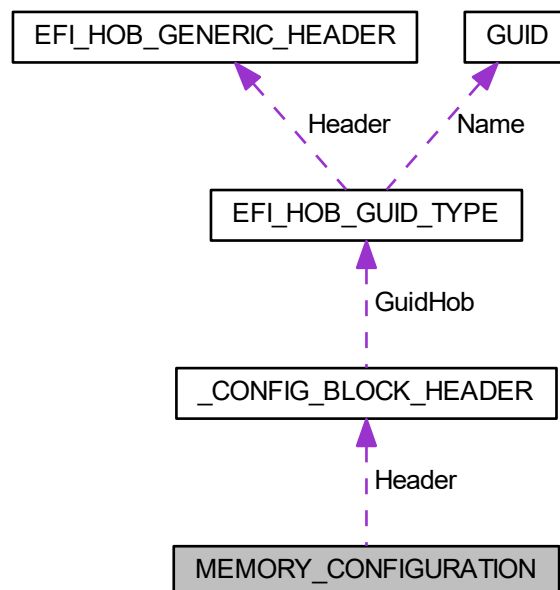
- [MemoryConfig.h](#)

13.110 MEMORY_CONFIGURATION Struct Reference

Memory Configuration The contents of this structure are CRC'd by the MRC for option change detection.

```
#include <MemoryConfig.h>
```

Collaboration diagram for MEMORY_CONFIGURATION:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Offset 0-27 Config Block Header.
- [UINT16 Size](#)
Offset 28 The size of this structure, in bytes. Must be the first entry in this structure.
- [UINT8 HobBufferSize](#)
Offset 30 Size of HOB buffer for MRC.
- [UINT8 SpdProfileSelected](#)
*Offset 31 SPD XMP profile selection - for XMP supported DIMM: **0=Default DIMM profile**, 1=Custom Profile, 2=XMP Profile 1, 3=XMP Profile 2, 4=XMP Profile 3, 5=XMP User Profile 4, 6=XMP User Profile 5.*
- [UINT16 tCL](#)
*Offset 32 User defined Memory Timing tCL value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 31=Maximum.*
- [UINT16 tRCDtRP](#)
*Offset 34 User defined Memory Timing tRCD value (same as tRP), valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 63=Maximum.*
- [UINT16 tRAS](#)
*Offset 36 User defined Memory Timing tRAS value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 64=Maximum.*
- [UINT16 tWR](#)
*Offset 38 User defined Memory Timing tWR value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, legal values: 5, 6, 7, 8, 10, 12, 14, 16, 18, 20, 24.*
- [UINT16 tRFC](#)
*Offset 40 User defined Memory Timing tRFC value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 1023=Maximum.*
- [UINT16 tRRD](#)
*Offset 42 User defined Memory Timing tRRD value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 15=Maximum.*
- [UINT16 tWTR](#)
*Offset 44 User defined Memory Timing tWTR value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 28=Maximum.*
- [UINT16 tRTP](#)
*Offset 46 User defined Memory Timing tRTP value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 15=Maximum. DDR4 legal values: 5, 6, 7, 8, 9, 10, 12.*
- [UINT16 tFAW](#)
*Offset 48 User defined Memory Timing tFAW value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 63=Maximum.*
- [UINT16 tCWL](#)
*Offset 50 User defined Memory Timing tCWL value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 20=Maximum.*
- [UINT16 tREFI](#)
*Offset 52 User defined Memory Timing tREFI value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 65535=Maximum.*
- [UINT16 PciIndex](#)
*Offset 54 Pci index register address: **0xCF8=Default***
- [UINT16 PciData](#)
*Offset 56 Pci data register address: **0xCFC=Default***
- [UINT16 VddVoltage](#)
*Offset 58 DRAM voltage (Vdd) in millivolts: **0=Platform Default (no override)**, 1200=1.2V, 1350=1.35V etc.*
- [UINT16 Idd3n](#)
Offset 60 EPG Active standby current (Idd3N) in milliamps from DIMM datasheet.
- [UINT16 Idd3p](#)

- Offset 62 EPG Active power-down current (Idd3P) in milliamps from DIMM datasheet.
- UINT32 [EccSupport](#):1
 - Offset 64 Bit 0 - DIMM Ecc Support option - for Desktop only: 0=Disable, 1=Enable
- UINT32 [MrcSafeConfig](#):1
 - Bit 1 - MRC Safe Mode: 0=Disable, 1=Enable.
- UINT32 [RemapEnable](#):1
 - Bit 2 - This option is used to control whether to enable/disable memory remap above 4GB: 0=Disable, 1=Enable.
- UINT32 [ScramblerSupport](#):1
 - Bit 3 - Memory scrambler support: 0=Disable, 1=Enable
- UINT32 [Vc1ReadMeter](#):1
 - Bit 4 - VC1 Read Metering Enable: 0=Disable, 1=Enable
- UINT32 [ForceSingleSubchannel](#):1
 - Bit 5 - TRUE means use SubChannel0 only (for LPDDR4): 0=Disable, 1=Enable.
- UINT32 [SimicsFlag](#):1
 - Bit 6 - Option to Enable SIMICS: 0=Disable, 1=Enable
- UINT32 [Ddr4DqpSharedClock](#):1
 - Bit 7 - Select if CLK0 is shared between Rank0 and Rank1 in DDR4 DDP package. 0=Not shared, 1=Shared.
- UINT32 [SharedZqPin](#):1
 - Bit 8 - Select if the ZQ resistor is shared between Ranks in DDR4/LPDDR4 DRAM Packages 0=Not Shared, 1=Shared.
- UINT32 [LpDqsOscEn](#):1
 - Bit 9 - LPDDR Write DQ/DQS Retraining: 0=Disable, 1=Enable
- UINT32 [RmtPerTask](#):1
 - Bit 10 - Rank Margin Tool Per Task. 0 = Disabled, 1 = Enabled.
- UINT32 [TrainTrace](#):1
 - Bit 11 - Trained state tracing debug. 0 = Disabled, 1 = Enabled.
- UINT32 [SafeMode](#):1
 - Bit 12 - Define if safe mode is enabled for MC/IO.
- UINT32 [MsHashEnable](#):1
 - Bit 13 - Controller Hash Enable: 0=Disable, 1=Enable
- UINT32 [DisPgCloseIdleTimeout](#):1
 - Bit 14 - Disable Page Close Idle Timeout: 0=Enable, 1=Disable
- UINT32 [lbecc](#):1
 - Bit 15 - In-band ECC: 0=Disable, 1=Enable.
- UINT32 [lbeccOperationMode](#):2
 - Bits 16:17 - In-band ECC Operation Mode: 0=Functional Mode protects requests based on the address range, 1=Makes all requests non protected and ignore range checks, 2=Makes all requests protected and ignore range checks
- UINT32 [ChHashOverride](#):1
 - Bit 18 - Select if Channel Hash setting values will be taken from input parameters or automatically taken from POR values depending on DRAM type detected.
- UINT32 [EccDftEn](#):1
 - Bit 19 - ECC DFT support option.
- UINT32 [Write0](#):1
 - Bit 20 - Write0 feature (LP5/DDR5): 0=Disable, 1=Enable
- UINT32 [ReuseAdlSDdr5Board](#):1
 - Bit 21 - Indicate whether adl ddr5 board is reused.
- UINT32 [PeriodicDcc](#):1
 - Bit 22 - Periodic DCC: 0=Disable, 1=Enable.
- UINT32 [LpMode](#):2
 - Bit 23:24 - LPMode: 0=Auto, 1=Enabled, 2=Disabled, 3=Reserved.

- UINT32 [RsvdO64B25to31](#):7
Bits 25:31 reserved.
- UINT8 [DisableChannel](#) [[MEM_CFG_MAX_CONTROLLERS](#)][[MEM_CFG_MAX_CHANNELS](#)]
Offset 68-75.
- UINT8 [Ratio](#)
*Offset 76 DDR Frequency ratio, to multiply by 133 or 100 MHz depending on RefClk. **0 = Auto***
- UINT8 [ProbelessTrace](#)
*Offset 77 Probeless Trace: **0=Disabled, 1=Enabled***
- UINT8 [ChHashInterleaveBit](#)
Channel Hash Enable.
- UINT8 [SmramMask](#)
Offset 79 Reserved memory ranges for SMRAM.
- UINT32 [BClkFrequency](#)
*Offset 80 Base reference clock value, in Hertz: **100000000 = 100Hz**, 125000000=125Hz, 167000000=167Hz, 250000000=250Hz.*
- UINT32 [ECT](#):1
Training Algorithms 1 Offset 84.
- UINT32 [SOT](#):1
*Bit 1 - Enable/Disable Sense Amp Offset Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [ERDMPRTC2D](#):1
*Bit 2 - Enable/Disable Early ReadMPR Timing Centering 2D. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [RDMPRT](#):1
*Bit 3 - Enable/Disable Read MPR Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [RCVET](#):1
*Bit 4 - Enable/Disable Receive Enable Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [JWRL](#):1
*Bit 5 - Enable/Disable JEDEC Write Leveling Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [EWRTC2D](#):1
*Bit 6 - Enable/Disable Early Write Time Centering 2D Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [ERDTC2D](#):1
*Bit 7 - Enable/Disable Early Read Time Centering 2D Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [WRTC1D](#):1
*Bit 8 - Enable/Disable 1D Write Timing Centering Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [WRVC1D](#):1
*Bit 9 - Enable/Disable 1D Write Voltage Centering Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [RDTC1D](#):1
*Bit 10 - Enable/Disable 1D Read Timing Centering Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [DIMMODTT](#):1
*Bit 11 - Enable/Disable DIMM ODT Training. Note it is not recommended to change this setting from the default value: **0=Disable**, 1=Enable.*
- UINT32 [DIMMRONT](#):1
*Bit 12 - Enable/Disable DIMM RON training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*

- UINT32 [WRDSEQ](#):1
*Bit 13 - Enable/Disable Write Drive Strength / Equalization Training 2D. Note it is not recommended to change this setting from the default value: **0=Disable**, 1=Enable.*
- UINT32 [WRSRT](#):1
*Bit 14 - Enable/Disable Write Slew Rate training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [RDODTT](#):1
*Bit 15 - Enable/Disable Read ODT Training. Note it is not recommended to change this setting from the default value: **0=Disable**, 1=Enable.*
- UINT32 [RDEQT](#):1
*Bit 16 - Enable/Disable Read Equalization Training. Note it is not recommended to change this setting from the default value: **0=Disable**, 1=Enable.*
- UINT32 [RDAPT](#):1
*Bit 17 - Enable/Disable Read Amplifier Power Training. Note it is not recommended to change this setting from the default value: **0=Disable**, 1=Enable.*
- UINT32 [WRTC2D](#):1
*Bit 18 - Enable/Disable 2D Write Timing Centering Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [RDTC2D](#):1
*Bit 19 - Enable/Disable 2D Read Timing Centering Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [WRVC2D](#):1
*Bit 20 - Enable/Disable 2D Write Voltage Centering Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [RDVC2D](#):1
*Bit 21 - Enable/Disable 2D Read Voltage Centering Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [CMDVC](#):1
*Bit 22 - Enable/Disable Command Vref Centering Training. Note it is not recommended to change this setting from the default value 0=Disable, **1=Enable**.*
- UINT32 [LCT](#):1
*Bit 23 - Enable/Disable Late Command Training. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [RTL](#):1
*Bit 24 - Enable/Disable Round Trip Latency function. Note it is not recommended to change this setting from the default value: 0=Disable, **1=Enable**.*
- UINT32 [TAT](#):1
*Bit 25 - Enable/Disable Turn Around Time function. Note it is not recommended to change this setting from the default value: **0=Disable**, 1=Enable.*
- UINT32 [RMT](#):1
*Bit 26 - Enable/Disable Rank Margin Tool function: **0=Disable**, 1=Enable.*
- UINT32 [MEMTST](#):1
*Bit 27 - Enable/Disable Memory Test function: **0=Disable**, 1=Enable.*
- UINT32 [ALIASCHK](#):1
*Bit 28 - Enable/Disable DIMM SPD Alias Check: 0=Disable, **1=Enable***
- UINT32 [RCVENC1D](#):1
*Bit 29 - Enable/Disable Receive Enable Centering Training (LPDDR Only). Note it is not recommended to change this setting from the default value: **0=Disable**, 1=Enable.*
- UINT32 [RMC](#):1
*Bit 30 - Enable/Disable Retrain Margin Check. Note it is not recommended to change this setting from the default value: **0=Disable**, 1=Enable.*
- UINT32 [WRDSUDT](#):1
Bit 31 - Enable/Disable Write Drive Strength Up/Dn independently.
- UINT32 [DCC](#): 1

Training Algorithms 2 Offset 88.

- UINT32 **RDVC1D**: 1
Bit 1 - Enable/Disable Read Voltage Centering 1D: 0=Disable, 1=Enable.
- UINT32 **TXTCO**: 1
Bit 2 - Enable/Disable Write TCO Comp Training: 0=Disable, 1=Enable.
- UINT32 **CLKTCO**: 1
Bit 3 - Enable/Disable Clock TCO Comp Training: 0=Disable, 1=Enable.
- UINT32 **CMDSR**: 1
Bit 4 - Enable/Disable CMD Slew Rate Training: 0=Disable, 1=Enable.
- UINT32 **CMDDSEQ**: 1
Bit 5 - Enable/Disable CMD Drive Strength and Tx Equalization: 0=Disable, 1=Enable.
- UINT32 **DIMMODTCA**: 1
Bit 6 - Enable/Disable Dimm ODT CA Training: 0=Disable, 1=Enable.
- UINT32 **TXTCODQS**: 1
Bit 7 - Enable/Disable Write TCO Dqs Training: 0=Disable, 1=Enable.
- UINT32 **CMDDRUD**: 1
Bit 8 - Enable/Disable CMD/CTL Drive Strength Up/Dn 2D: 0=Disable, 1=Enable.
- UINT32 **VCCDLLBP**: 1
Bit 9 - Enable/Disable VccDLL bypass to VccLOG training: 0=Disable, 1=Enable.
- UINT32 **PVTTDNLp**: 1
Bit 10 - Enable/Disable PanicVttDnLp Training: 0=Disable, 1=Enable.
- UINT32 **RDVREFDC**: 1
Bit 11 - Enable/Disable Read Vref Decap Training: 0=Disable, 1=Enable.
- UINT32 **VDDQT**: 1
Bit 12 - Enable/Disable Vddq Training: 0=Disable, 1=Enable.
- UINT32 **RMTBIT**: 1
Bit 13 - Enable/Disable Rank Margin Tool Per Bit: 0=Disable, 1=Enable.
- UINT32 **PDA**: 1
BIT 14 - Enable/Disable PDA Enumeration Training. Note it is not recommended to change this setting from the default value: 0=Disable, 1=Enable.
- UINT32 **DIMMDFE**: 1
BIT 15 - Enable/Disable DIMM DFE Training: 0=Disable, 1=Enable.
- UINT32 **TXDQSDCC**: 1
BIT 16 - Enable/Disable TX DQS DCC Training: 0=Disable, 1=Enable.
- UINT32 **DRAMDCA**: 1
BIT 17 - Enable/Disable DRAM DCA Training: 0=Disable, 1=Enable.
- UINT32 **WRDS**: 1
BIT 18 - Enable/Disable Write Driver Strength Training: 0=Disable, 1=Enable.
- UINT32 **EARLYDIMMDFE**: 1
BIT 19 - Enable/Disable EARLY DIMM DFE Training: 0=Disable, 1=Enable.
- UINT32 **DQSOFFSETADJUST**: 1
BIT 20 - Enable/Disable DQS OFFSET ADJUST Training: 0=Disable, 1=Enable.
- UINT32 **DRAMEMPHASIS**: 1
BIT 21 - Enable/Disable DRAMEMPHASIS Training: 0=Disable, 1=Enable.
- UINT32 **PPR**: 1
BIT 22 - Enable/Disable Post Package Repair Training: 0=Disable, 1=Enable.
- UINT32 **ReservedBits2**: 9
Bits 23:31 - Reserved.
- UINT32 **MrcTimeMeasure**: 1
Offset 92 Bit 0 - Enables serial debug level to display the MRC execution times only: 0=Disable, 1=Enable.
- UINT32 **MrcFastBoot**: 1

- Bit 1 - Enables the MRC fast boot path for faster cold boot execution: 0=Disable, 1=Enable*
- UINT32 [DqPinsInterleaved](#):1
 - Bit 2 - Interleaving mode of DQ/DQS pins which depends on board routing: 0=Disable, 1=Enable.*
- UINT32 [RankInterleave](#):1
 - Bit 3 - Rank Interleave Mode: 0=Disable, 1=Enable*
- UINT32 [EnhancedInterleave](#):1
 - Bit 4 - Enhanced Interleave Mode: 0=Disable, 1=Enable*
- UINT32 [WeaklockEn](#):1
 - Bit 5 - Weak Lock Enable: 0=Disable, 1=Enable*
- UINT32 [ChHashEnable](#):1
 - Bit 6 - Channel Hash Enable: 0=Disable, 1=Enable*
- UINT32 [EnablePwrDn](#):1
 - Bit 7 - Enable Power Down control for DDR: 0=PCODE control, 1=BIOS control*
- UINT32 [EnablePwrDnLpddr](#):1
 - Bit 8 - Enable Power Down for LPDDR: 0=PCODE control, 1=BIOS control*
- UINT32 [SrefCfgEna](#):1
 - Bit 9 - Enable Self Refresh: 0=Disable, 1=Enable*
- UINT32 [ThrtCkeMinDefeatLpddr](#):1
 - Bit 10 - Throttler CKE min defeature for LPDDR: 0=Disable, 1=Enable*
- UINT32 [ThrtCkeMinDefeat](#):1
 - Bit 11 - Throttler CKE min defeature: 0=Disable, 1=Enable.*
- UINT32 [AutoSelfRefreshSupport](#):1
 - Bit 12 - FALSE = No auto self refresh support, TRUE = auto self refresh support*
- UINT32 [ExtTemperatureSupport](#):1
 - Bit 13 - FALSE = No extended temperature support, TRUE = extended temperature support*
- UINT32 [MobilePlatform](#):1
 - Bit 14 - Memory controller device id indicates: TRUE if mobile, FALSE if not. Note: This will be auto-detected and updated.*
- UINT32 [Force1Dpc](#):1
 - Bit 15 - TRUE means force one DIMM per channel, FALSE means no limit*
- UINT32 [ForceSingleRank](#):1
 - Bit 16 - TRUE means use Rank0 only (in each DIMM): 0=Disable, 1=Enable.*
- UINT32 [VttTermination](#):1
 - Bit 17 - Vtt Termination for Data ODT: 0=Disable, 1=Enable.*
- UINT32 [VttCompForVsshi](#):1
 - Bit 18 - Enable/Disable Vtt Comparator For Vsshi: 0=Disable, 1=Enable.*
- UINT32 [ExitOnFailure](#):1
 - Bit 19 - MRC option for exit on failure or continue on failure: 0=Disable, 1=Enable*
- UINT32 [NewFeatureEnable1](#):1
 - Bit 20 - Generic enable knob for new feature set 1 0: Disable ; 1: Enable.*
- UINT32 [NewFeatureEnable2](#):1
 - Bit 21 - Generic enable knob for new feature set 2 0: Disable ; 1: Enable.*
- UINT32 [RhSelect](#):2
 - Bit 22-23 - RH Select: 0=Disable, 1=RFM, 2=pTRR.*
- UINT32 [RefreshPanicWm](#):4
 - Bit 24-27 - Deprecated from revision 17. Use RefreshWm instead.*
- UINT32 [RefreshHpWm](#):4
 - Bit 28-31 - Deprecated from revision 17. Use RefreshWm instead.*
- UINT32 [VddSettleWaitTime](#)
 - Offset 96 Amount of time in microseconds to wait for Vdd to settle on top of 200us required by JEDEC spec: Default=0*
- UINT16 [SrefCfgIdleTmr](#)

- Offset 100 Self Refresh idle timer: **512=Minimal**, 65535=Maximum.
- UINT16 [ChHashMask](#)
Offset 102 Channel Hash Mask: 0x0001=BIT6 set(Minimal), 0x3FFF=BIT[19:6] set(Maximum), **0x30CE= BIT[19:18, 13:12 ,9:7] set**
 - UINT16 [DdrFreqLimit](#)
Offset 104 Memory Frequency limit: **0 = Auto**, or memory speed value in MT/s: 1067, 1333, 1600 etc. See the possible values in MrcInterface.h.
 - UINT8 [MaxRttWr](#)
Offset 106 Maximum DIMM RTT_WR to use in power training: **0=ODT Off**, 1 = 120 ohms.
 - UINT8 [ThrtCkeMinTmr](#)
Offset 107 Throttler CKE min timer: 0=Minimal, 0xFF=Maximum, **0x00=Default**
 - UINT8 [ThrtCkeMinTmrLpddr](#)
Offset 108 Throttler CKE min timer for LPDDR: 0=Minimal, 0xFF=Maximum, **0x00=Default**
 - BOOLEAN [PerBankRefresh](#)
Offset 109 Enables and Disables the per bank refresh. This only impacts memory technologies that support PBR: LPDDR4, LPDDR5 and DDR5. FALSE=Disabled, **TRUE=Enabled**
 - UINT8 [SaGv](#)
Offset 110 SA GV: **0=Disabled**, 1=Point1, 2=Point2, 3=Point3, 4=Point4, 5=Enabled.
 - UINT8 [NModeSupport](#)
Offset 111 Memory N Mode Support - Enable user to select Auto, 1N or 2N: **0=AUTO**, 1=1N, 2=2N.
 - UINT8 [RefClk](#)
Offset 112 Selects the DDR base reference clock. 0x01 = 100MHz, **0x00 = 133MHz**
 - UINT8 [EnCmdRate](#)
Offset 113 CMD Rate Enable: 0=Disable, 5=2 CMDs, **7=3 CMDs**, 9=4 CMDs, 11=5 CMDs, 13=6 CMDs, 15=7 CMDs.
 - UINT8 [Refresh2X](#)
Offset 114 Refresh 2x: **0=Disable**, 1=Enable for WARM or HOT, 2=Enable for HOT only.
 - UINT8 [EpgEnable](#)
Offset 115 Enable Energy Performance Gain.
 - UINT8 [UserThresholdEnable](#)
Offset 116 Flag to manually select the DIMM CLTM Thermal Threshold, 0=Disable, 1=Enable, **0=Default**
 - UINT8 [UserBudgetEnable](#)
Offset 117 Flag to manually select the Budget Registers for CLTM Memory Dimms , 0=Disable, 1=Enable, **0=Default**
 - UINT8 [RetrainOnFastFail](#)
Offset 118 Restart MRC in Cold mode if SW MemTest fails during Fast flow. 0 = Disabled, **1 = Enabled**
 - UINT8 [PowerDownMode](#)
Offset 119 CKE Power Down Mode: **0xFF=AUTO**, 0=No Power Down, 1= APD mode, 6=PPD-DLL Off mode.
 - UINT8 [PwdownIdleCounter](#)
Offset 120 CKE Power Down Mode Idle Counter: 0=Minimal, 255=Maximum, **0x80=0x80 DCLK**
 - UINT8 [CmdRanksTerminated](#)
Offset 121 LPDDR: Bitmask of ranks that have CA bus terminated. **0x01=Default, Rank0 is terminating and Rank1 is non-terminating**
 - UINT16 [MsHashMask](#)
Offset 122 Controller Hash Mask: 0x0001=BIT6 set(Minimal), 0x3FFF=BIT[19:6] set(Maximum), **0x30CE= BIT[19:18, 13:12 ,9:7] set**
 - UINT32 [Lp5CccConfig](#)
Offset 124 BitMask where bits [3:0] are controller 0 Channel [3:0] and [7:4] are Controller 1 Channel [3:0]. 0 selects Ascending mapping and 1 selects Descending mapping.
 - UINT8 [RMTLoopCount](#)
Offset 128 Indicates the Loop Count to be used for Rank Margin Tool Testing: 1=Minimal, 32=Maximum, 0=AUTO, **0=Default**
 - UINT8 [MsHashInterleaveBit](#)
Offset 129 Option to select interleave Address bit. Valid values are 0 - 3 for BITS 6 - 9.

- [UINT8 GearRatio](#)
Offset 130 Gear mode when SAGV is disabled: **0=Auto**, 1 - Gear1, 2 - Gear2, 4 - Gear4.
- [UINT8 Ddr4OneDpc](#)
Offset 131 DDR4 1DPC performance feature: 0 - Disabled; 1 - Enabled on DIMM0 only, 2 - Enabled on DIMM1 only; 3 - Enabled on both DIMMs. (bit [0] - DIMM0, bit [1] - DIMM1)
- [UINT32 BclkRfiFreq](#) [MEM_MAX_SAGV_POINTS]
Offset 132 Bclk RFI Frequency for each SAGV point in Hz units. 98000000Hz = 98MHz **0 - No RFI Tuning**. Range is 98Mhz-100Mhz.
- [UINT16 SaGvFreq](#) [MEM_MAX_SAGV_POINTS]
Offset 148 Frequency per SAGV point. 0 is Auto, otherwise holds the frequency value expressed as an integer: **0=Default**, 1067, 1333, 1600, 1800, 1867, etc.
- [UINT8 SaGvGear](#) [MEM_MAX_SAGV_POINTS]
Offset 156 Gear ratio per SAGV point. 0 is Auto, otherwise holds the Gear ratio expressed as an integer: **0=Auto**, 1, 2, 4.
- [UINT8 IbeccProtectedRangeEnable](#) [MEM_MAX_IBECC_REGIONS]
Offset 160 Enable use of address range for ECC Protection: **0=Disabled**, 1=Enabled.
- [UINT32 IbeccProtectedRangeBase](#) [MEM_MAX_IBECC_REGIONS]
Offset 168 Base address for address range of ECC Protection: [0..0x3FFFFFFF].
- [UINT32 IbeccProtectedRangeMask](#) [MEM_MAX_IBECC_REGIONS]
Offset 200 Mask address for address range of ECC Protection: [1..0x3FFFFFFF].
- [UINT8 McRefresh2X](#)
Offset 232 Deprecated from revision 19, use McRefreshRate instead.
- [UINT8 Lfsr0Mask](#)
Offset 233 RH pTRR LFSR0 Mask.
- [UINT8 Lfsr1Mask](#)
Offset 234 RH pTRR LFSR1 Mask.
- [UINT8 CmdMirror](#)
Offset 235 BitMask where bits [3:0] are controller 0 Channel [3:0] and [7:4] are Controller 1 Channel [3:0]. 0 = No Command Mirror and 1 = Command Mirror.
- [UINT8 AllowOppRefBelowWriteThreshold](#)
Offset 236 Option to allow opportunistic refreshes while we don't exit power down.
- [UINT8 WriteThreshold](#)
Offset 237 Option to set number of writes that can be accumulated while CKE is low before CKE is asserted.
- [UINT8 PprEnable](#)
Offset 238 Post-Package Repair: 0: Disabled, 2: Hard PPR.
- [UINT8 MarginLimitCheck](#)
Offset 239 Margin limit check enable: **0=Disable**, 1=L1 only, 2=L2 only, 3=Both L1 and L2.
- [UINT16 MarginLimitL2](#)
Offset 240 Margin limit check L2 threshold: **100=Default**
- [UINT8 LpddrRttWr](#)
Offset 242 Initial RttWr for LP4/5 in Ohms, 0 means Auto.
- [UINT8 LpddrRttCa](#)
Offset 243 Initial RttCa for LP4/5 in Ohms, 0 means Auto.
- [UINT16 VddqVoltage](#)
Offset 244 DRAM voltage (Vddq) in millivolts: **0=Platform Default (no override)**, 1200=1.2V, 1350=1.35V etc.
- [UINT16 VppVoltage](#)
Offset 246 DRAM voltage (Vpp) in millivolts: **0=Platform Default (no override)**, 1800=1.8V, 2050=2.05V etc.
- [UINT8 ExtendedBankHashing](#)
Offset 248 Enable EBH (Extended Bank Hashing): 0=Disabled; **1 = Enabled**
- [UINT8 DynamicMemoryBoost](#)
Offset 249 Enable/Disable Dynamic Memory Boost Feature. Only valid if SpdProfileSelected is an XMP Profile; otherwise ignored. **0=Disabled**, 1=Enabled.

- [UINT8 lbeccErrInjControl](#)
Offset 250 In-band ECC: Error Injection Control 0: No Error Injection, 1:Inject Correctable Error Address match, 3:Inject Correctable Error on insertion counter, 5: Inject Uncorrectable Error Address match, 7:Inject Uncorrectable Error on insertion counter.
- [UINT8 RealtimeMemoryFrequency](#)
Offset 251 Enable/Disable Realtime Memory Frequency feature. Only valid if SpdProfileSelected is an XMP Profile; otherwise ignored. **0=Disabled**, 1=Enabled.
- [UINT32 lbeccErrInjCount](#)
Offset 252 Number of transactions between ECC error injection.
- [UINT64 lbeccErrInjAddress](#)
Offset 256 Address to match against for ECC error injection.
- [UINT64 lbeccErrInjMask](#)
Offset 264 Mask to match against for ECC error injection.
- [UINT8 RefreshWm](#)
Offset 272 Refresh Watermarks, 0 = Low, **1 = High**
- [UINT8 McRefreshRate](#)
Offset 273 Type of solution to be used for RHP - 0/1/2/3 = RefreshNORMAL/Refresh1x/Refresh2x/Refresh4x.
- [UINT8 DisableSagvReorder](#)
Offset 274 Disable sagv reordering on warm reboot.
- [UINT8 PprRunOnce](#)
Offset 275 Post-Package Repair Just Run Once: 0: Disabled, 1: Enabled.
- [UINT32 DebugValue](#)
Offset 276 Debug Value for general use.
- [UINT16 tRFCpb](#)
Offset 280 User defined Memory Timing tRFCpb value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 28=Maximum.
- [UINT16 tRFC2](#)
Offset 282 User defined Memory Timing tRFC2 value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 28=Maximum.
- [UINT16 tRFC4](#)
Offset 284 User defined Memory Timing tRFC4 value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 28=Maximum.
- [UINT16 tRRD_L](#)
Offset 286 User defined Memory Timing tRRD_L value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 28=Maximum.
- [UINT16 tRRD_S](#)
Offset 288 User defined Memory Timing tRRD_S value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 28=Maximum.
- [UINT16 tWTR_L](#)
Offset 290 User defined Memory Timing tWTR_L value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 28=Maximum.
- [UINT16 tCCD_L](#)
Offset 292 User defined Memory Timing tCCD_L value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 28=Maximum.
- [UINT16 tWTR_S](#)
Offset 294 User defined Memory Timing tWTR_S value, valid when SpdProfileSelected is CUSTOM_PROFILE: **0=AUTO**, 28=Maximum.
- [UINT64 EccErrInjAddress](#)
Offset 296 Address to match against for ECC error injection.
- [UINT64 EccErrInjMask](#)
Offset 304 Mask to match against for ECC error injection.
- [UINT32 EccErrInjCount](#)
Offset 312 Number of transactions between ECC error injection.

- UINT16 [FreqLimitMixedConfig](#)
Offset 316 Override the reduced speed in mixed 2DPC config or non-POR 2DPC config. **0=Auto**, otherwise speed in MT/s: 1067, 1333 etc. See the possible values in MrcInterface.h.
- UINT8 [FirstDimmBitMask](#)
Offset 318 Defines which DIMM should be populated first on a 2DPC board.
- UINT8 [UserBd](#)
Offset 319 MrcBoardType 1 - Desktop 1DPC 2 - Desktop 2DPC, Daisy Chain, far DIMM should be populated first 3 - Desktop 2DPC, T-topology with asymmetrical DIMM0/1 routing, has particular DIMM population order 4 - Desktop 2DPC, T-topology with symmetrical DIMM0/1 routing, no particular DIMM population order 5 - ULT Mobile.
- UINT8 [SagvSwitchFactorIA](#)
Offset 320 Sagv Switch Factor: **1** , 50=Maximum, default 30.
- UINT8 [SagvSwitchFactorGT](#)
Offset 321 Sagv Switch Factor: **1** , 50=Maximum, default 30.
- UINT8 [SagvSwitchFactorIO](#)
Offset 322 Sagv Switch Factor: **1** , 50=Maximum, default 30.
- UINT8 [SagvSwitchFactorStall](#)
Offset 323 Sagv Switch Factor: **1** , 50=Maximum, default 30.
- UINT8 [SagvHeuristicsUpControl](#)
Offset 324 Sagv duration in ms before heuristics up: **1** , 50=Maximum, default 1ms.
- UINT8 [SagvHeuristicsDownControl](#)
Offset 325 Sagv duration in ms before heuristics down: **1** , 50=Maximum, default 1ms.
- UINT16 [FreqLimitMixedConfig_1R1R_8GB](#)
Offset 326 The frequency limit for mixed DIMM configuration 1R1R_8GB (1R 8GB and 1R 8GB in one channel).
- UINT16 [FreqLimitMixedConfig_1R1R_16GB](#)
Offset 328 The frequency limit for mixed DIMM configuration 1R1R_16GB (1R 16GB and 1R 16GB in one channel).
- UINT16 [FreqLimitMixedConfig_1R1R_8GB_16GB](#)
Offset 330 The frequency limit for mixed DIMM configuration 1R1R_8GB_16GB (1R 8GB and 1R 16GB in one channel).
- UINT16 [FreqLimitMixedConfig_2R2R](#)
Offset 332 The frequency limit for mixed DIMM configuration 2R_2R (2R 32GB and 2R 32GB in one channel).
- UINT16 [LctCmdEyeWidth](#)
Offset 334 LCT Command eyewidth.
- UINT8 [FirstDimmBitMaskEcc](#)
Offset 336 Defines which ECC DIMM should be populated first on a 2DPC board.
- UINT8 [Lp5BankMode](#)
Offset 337 LP5 Bank Mode.
- UINT8 [OverloadSAM](#)
Offset 338 Overload SAM to copy the sagv frequency point.
- UINT8 [OCSafeMode](#)
Offset 339 OverClocking Safe Mode.
- UINT8 [DfeGain](#)
Offset 340 DfeGain.
- UINT8 [CsPiStartHighinEct](#)
Offset 341 CsPiStartHighinEct.
- UINT8 [DisableFGRAndPBRWA](#)
Offset 342 DisableFGRAndPBRWA.
- UINT8 [LowerBasicMemTestSize](#)
Offset 343 LowerBasicMemTestSize.

13.110.1 Detailed Description

Memory Configuration The contents of this structure are CRC'd by the MRC for option change detection.

This structure is copied en mass to the MrclInput structure. If you add fields here, you must update the MrclInput structure. Revision history is at the top of this file

Definition at line 318 of file MemoryConfig.h.

13.110.2 Member Data Documentation

13.110.2.1 ChHashInterleaveBit

```
UINT8 MEMORY_CONFIGURATION::ChHashInterleaveBit
```

Channel Hash Enable.

NOTE: BIT7 will interleave the channels at a 2 cache-line granularity, BIT8 at 4 and BIT9 at 8
0=BIT6, **1=BIT7**, 2=BIT8, 3=BIT9 Offset 78 Option to select interleave Address bit. Valid values are 0 - 3 for BITS
6 - 9 (Valid values for BDW are 0-7 for BITS 6 - 13)

Definition at line 375 of file MemoryConfig.h.

13.110.2.2 DCC

```
UINT32 MEMORY_CONFIGURATION::DCC
```

Training Algorithms 2 Offset 88.

Bit 0 - Enable/Disable Duty Cycle Correction: 0=Disable, 1=Enable.

Definition at line 413 of file MemoryConfig.h.

13.110.2.3 ECT

```
UINT32 MEMORY_CONFIGURATION::ECT
```

Training Algorithms 1 Offset 84.

Bit 0 - Enable/Disable Early Command Training. Note it is not recommended to change this setting from the default value: **0=Disable**, 1=Enable.

Definition at line 380 of file MemoryConfig.h.

13.110.2.4 FirstDimmBitMask

```
UINT8 MEMORY_CONFIGURATION::FirstDimmBitMask
```

Offset 318 Defines which DIMM should be populated first on a 2DPC board.

4 bit mask: Bit[0]: MC0 DIMM0, Bit[1]: MC0 DIMM1, Bit[2]: MC1 DIMM0, Bit[3]: MC1 DIMM1. For each MC, the first DIMM to be populated should be set to '1'. Note: this mask is only for non-ECC DIMM.

Definition at line 535 of file MemoryConfig.h.

13.110.2.5 FirstDimmBitMaskEcc

```
UINT8 MEMORY_CONFIGURATION::FirstDimmBitMaskEcc
```

Offset 336 Defines which ECC DIMM should be populated first on a 2DPC board.

4 bit mask: Bit[0]: MC0 DIMM0, Bit[1]: MC0 DIMM1, Bit[2]: MC1 DIMM0, Bit[3]: MC1 DIMM1. For each MC, the first DIMM to be populated should be set to '1'. For example, if one MC is T-topology, there is no special population rule, can put it as 11 for this MC and it means either D0 or D1 can be be populated firstly. Note: this mask is only for ECC DIMM, not for non-ECC DIMM.

Definition at line 560 of file MemoryConfig.h.

13.110.2.6 FreqLimitMixedConfig_1R1R_16GB

```
UINT16 MEMORY_CONFIGURATION::FreqLimitMixedConfig_1R1R_16GB
```

Offset 328 The frequency limit for mixed DIMM configuration 1R1R_16GB (1R 16GB and 1R 16GB in one channel).

It's to be set by customer. Customer can connect it to a setup option, or an UPD, etc. By default, the value will be 2000.

Definition at line 553 of file MemoryConfig.h.

13.110.2.7 FreqLimitMixedConfig_1R1R_8GB

```
UINT16 MEMORY_CONFIGURATION::FreqLimitMixedConfig_1R1R_8GB
```

Offset 326 The frequency limit for mixed DIMM configuration 1R1R_8GB (1R 8GB and 1R 8GB in one channel).

It's to be set by customer. Customer can connect it to a setup option, or an UPD, etc. By default, the value will be 2000.

Definition at line 551 of file MemoryConfig.h.

13.110.2.8 FreqLimitMixedConfig_1R1R_8GB_16GB

```
UINT16 MEMORY_CONFIGURATION::FreqLimitMixedConfig_1R1R_8GB_16GB
```

Offset 330 The frequency limit for mixed DIMM configuration 1R1R_8GB_16GB (1R 8GB and 1R 16GB in one channel).

It's to be set by customer. Customer can connect it to a setup option, or an UPD, etc. By default, the value will be 2000.

Definition at line 555 of file MemoryConfig.h.

13.110.2.9 FreqLimitMixedConfig_2R2R

```
UINT16 MEMORY_CONFIGURATION::FreqLimitMixedConfig_2R2R
```

Offset 332 The frequency limit for mixed DIMM configuration 2R_2R (2R 32GB and 2R 32GB in one channel).

It's to be set by customer. Customer can connect it to a setup option, or an UPD, etc. By default, the value will be 2000.

Definition at line 557 of file MemoryConfig.h.

13.110.2.10 WRDSUDT

```
UINT32 MEMORY_CONFIGURATION::WRDSUDT
```

Bit 31 - Enable/Disable Write Drive Strength Up/Dn independently.

Note it is not recommended to change this setting from the default value: **0=Disable**, 1=Enable

Definition at line 411 of file MemoryConfig.h.

The documentation for this struct was generated from the following file:

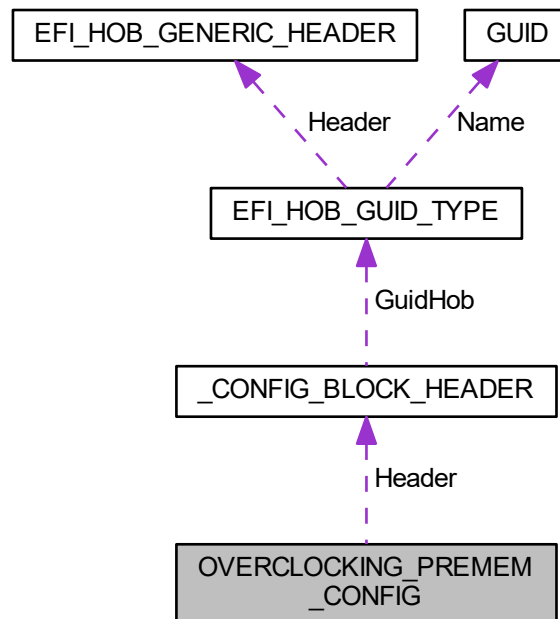
- [MemoryConfig.h](#)

13.111 OVERCLOCKING_PREMEM_CONFIG Struct Reference

Overclocking Configuration Structure.

```
#include <OverclockingConfig.h>
```

Collaboration diagram for OVERCLOCKING_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- UINT32 [OcSupport](#): 1
Overclocking support.
- UINT32 [OcLock](#): 1
*If enabled, sets OC lock bit in MSR 0x194[20], locking the OC mailbox and other OC configuration settings.; 0: Disable; 1: **Enable (Lock)**.*
- UINT32 [CoreVoltageMode](#): 1
Core voltage mode, specifies which voltage mode the processor will be operating.
- UINT32 [CorePllVoltageOffset](#): 6
*Core PLL voltage offset. 0: **No offset**. Range 0-15 in 17.5mv units.*
- UINT32 [Avx2RatioOffset](#): 5
*AVX2 Ratio Offset. 0: **No offset**. Range is 0-31. Used to lower the AVX ratio to maximize possible ratio for SSE workload.*
- UINT32 [Avx3RatioOffset](#): 5
deprecated from version 28

- UINT32 [BclkAdaptiveVoltage](#): 1
*Bclk Adaptive Voltage enable/disable. 0: **Disabled**, 1: Enabled. When enabled, the CPU V/F curves are aware of BCLK frequency when calculated.*
- UINT32 [RingDownBin](#): 1
Ring Downbin enable/disable.
- UINT32 [RingVoltageMode](#): 1
Ring voltage mode, specifies which voltage mode the processor will be operating.
- UINT32 [GtVoltageMode](#): 1
*Specifies whether GT voltage is operating in Adaptive or Override mode: 0=**Adaptive**, 1=Override.*
- UINT32 [RealtimeMemoryTiming](#): 1
*Enable/Disable the message sent to the CPU to allow realtime memory timing changes after MRC_DONE. 0=**Disable**, 1=Enable.*
- UINT32 [FivrEfficiency](#): 1
*Deprecated < Fivr Faults. Enable or Disable FIVR Faults. 0: Disabled, 1: **Enabled**.*
- UINT32 [CoreVfPointOffsetMode](#): 1
*Deprecated < Fivr Efficiency Management. 0: Disabled, 1: **Enabled**.*
- UINT32 [CoreVfConfigScope](#): 1
Core VF Configuration Scope.
- UINT32 [PerCoreRatioOverride](#): 1
*Enable or disable Per Core PState OC supported by writing OCMB 0x1D to program new favored core ratio to each Core. 0: **Disable**, 1: enable.*
- UINT32 [UnlimitedIccMax](#): 1
deprecated from version 32
- UINT32 [SaVoltageMode](#): 1
SA/Uncore voltage mode, specifies which voltage mode the processor will be operating.
- UINT32 [CoreRatioExtensionMode](#): 1
*Enable or disable Core Ratio above 85 Extension Mode by writing BIOS MB 0x37 to enable FULL_RANGE_MULT↔ IPLIER_UNLOCK_EN. 0: **Disable**, 1: enable.*
- UINT8 [CoreMaxOcRatio](#)
Maximum core turbo ratio override allows to increase CPU core frequency beyond the fused max turbo ratio limit (P0).
- UINT8 [GtMaxOcRatio](#)
*Maximum GT turbo ratio override: 0=Minimal, 60=Maximum, 0=**AUTO***
- UINT8 [RingMaxOcRatio](#)
Maximum ring ratio override allows to increase CPU ring frequency beyond the fused max ring ratio limit.
- UINT8 [AtomL2MaxOcRatio](#)
Maximum ATOM L2 turbo ratio override allows to increase CPU core frequency beyond the fused max turbo ratio limit (P0).
- UINT16 [CoreVoltageOverride](#)
The core voltage override which is applied to the entire range of cpu core frequencies.
- UINT16 [CoreVoltageAdaptive](#)
Adaptive Turbo voltage target used to define the interpolation voltage point when the cpu is operating in turbo mode range.
- INT16 [CoreVoltageOffset](#)
The core voltage offset applied on top of all other voltage modes.
- UINT16 [RingVoltageOverride](#)
The ring voltage override which is applied to the entire range of cpu ring frequencies.
- UINT16 [RingVoltageAdaptive](#)
Adaptive Turbo voltage target used to define the interpolation voltage point when the ring is operating in turbo mode range.
- INT16 [RingVoltageOffset](#)
The ring voltage offset applied on top of all other voltage modes.
- INT16 [GtVoltageOffset](#)

- The voltage offset applied to GT slice. Valid range from -1000mv to 1000mv: **0=Minimal**, 1000=Maximum.

 - UINT16 [GtVoltageOverride](#)

The GT voltage override which is applied to the entire range of GT frequencies **0=Default**
 - UINT16 [GtExtraTurboVoltage](#)

The adaptive voltage applied during turbo frequencies. Valid range from 0 to 2000mV: **0=Minimal**, 2000=Maximum.
 - INT16 [SaVoltageOffset](#)

The voltage offset applied to the SA. Valid range from -1000mv to 1000mv: **0=Default**
 - UINT32 [GtPllVoltageOffset](#): 6

GT PLL voltage offset. **0: No offset**. Range 0-15 in 17.5mv units.
 - UINT32 [RingPllVoltageOffset](#): 6

Ring PLL voltage offset. **0: No offset**. Range 0-15 in 17.5mv units.
 - UINT32 [SaPllVoltageOffset](#): 6

System Agent PLL voltage offset. **0: No offset**. Range 0-15 in 17.5mv units.
 - UINT32 [McPllVoltageOffset](#): 6

Memory Controller PLL voltage offset. **0: No offset**. Range 0-15 in 17.5mv units.
 - UINT32 [IaCepEnable](#): 1

Enable IA Current Excursion Protection. **1: Enable**. 0: Disable.
 - UINT32 [GtCepEnable](#): 1

Enable GT Current Excursion Protection. **1: Enable**. 0: Disable.
 - UINT32 [SaCepEnable](#): 1

Enable SA Current Excursion Protection. **1: Enable**. 0: Disable.
 - UINT32 [IaSirpEnable](#): 1

Enable IA SoC Iccmax Reactive Protector. **1: Enable**. 0: Disable.
 - UINT32 [ItdThrottleEnable](#): 1

Enable Inverse Temperature Dependency Throttle Support. **1: Enable**. 0: Disable.
 - UINT32 [DlvrBypassModeEnable](#): 1

Deprecated. Enable CPU DLVR bypass mode support, **0: Disable**. 0: Enable.
 - UINT32 [SaPllFreqOverride](#): 1

SA PLL Frequency Override. **0: 3200MHz**. 1: 1600MHz.
 - UINT32 [TscDisableHwFixup](#): 1

Core HW Fixup disable during TSC copy from PMA to APIC, **0: Enable**. 1: Disable.
 - UINT8 [TjMaxOffset](#)

TjMax Offset.
 - UINT8 [FastThrottleThreshold](#)

FastThrottleThreshold.
 - UINT8 [RsvdByte2](#) [2]

Reserved for dword alignment.
 - UINT32 [TvbRatioClipping](#): 1

This service controls Core frequency reduction caused by high package temperatures for processors that implement the Intel Thermal Velocity Boost (TVB) feature.
 - UINT32 [TvbVoltageOptimization](#): 1

This service controls thermal based voltage optimizations for processors that implement the Intel Thermal Velocity Boost (TVB) feature.
 - UINT32 [IalccUnlimitedMode](#): 1

Support IA Unlimited ICCMAX up to maximum value 512A., **0: Disable**. 1: Enable.
 - UINT32 [GtlccUnlimitedMode](#): 1

Support GT Unlimited ICCMAX up to maximum value 512A., **0: Disable**. 1: Enable.
 - UINT32 [FIIOcModeEn](#): 1

Enable FLL Mode override with the value in FIIOcMode. **0: Disable**. 1: Enable.
 - UINT32 [Etvb](#): 1

Enhanced Thermal Turbo Mode (eTVB). **0:eTVB is disabled on supporting products** 1:eTVB is enabled on supporting products.

- UINT32 [UnderVoltProtection](#): 2
When UnderVolt Protection is enabled, user will be not be able to program under voltage in OS runtime. 0: protection is disabled, **1:protection is enabled**
- UINT16 [PerCoreHtDisable](#)
Defines the per-core HT disable mask where: 1 - Disable selected logical core HT, 0 - is ignored.
- UINT8 [Avx2VoltageScaleFactor](#)
Avx2 Voltage Guardband Scale Factor This controls the AVX2 Voltage Guardband Scale factor applied to AVX2 workloads.
- UINT8 [Avx512VoltageScaleFactor](#)
Avx512 Voltage Guardband Scale Factor This controls the AVX512 Voltage Guardband Scale factor applied to AVX512 workloads.
- INT16 [CoreVfPointOffset](#) [CPU_OC_MAX_VF_POINTS]
Array used to specifies the Core Voltage Offset applied to the each selected VF Point.
- UINT8 [RsvdByte3](#) [2]
Reserved for dword alignment.
- UINT8 [CoreVfPointRatio](#) [CPU_OC_MAX_VF_POINTS]
Array for the each selected VF Point to display the Core Ration.
- UINT8 [CoreVfPointCount](#)
Number of supported Core Voltage & Frequency Point.
- INT16 [PerCoreVoltageOffset](#) [CPU_MAX_BIG_CORES]
Array used to specifies the selected Core Offset Voltage.
- UINT8 [AtomClusterRatio](#) [CPU_MAX_ATOM_CLUSTERS]
The Atom cluster max ratio overrides.
- UINT8 [AtomL2VoltageMode](#)
Atom L2 voltage mode, specifies which voltage mode the atom processor will be operating.
- UINT8 [RsvdByte4](#)
Reserved for dword alignment.
- UINT16 [AtomL2VoltageOverride](#)
The atom L2 voltage override which is applied to the entire range of atom core frequencies.
- UINT16 [AtomL2VoltageAdaptive](#)
Adaptive Turbo voltage target used to define the interpolation voltage point when the atom is operating in turbo mode range.
- INT16 [AtomL2VoltageOffset](#)
The atom L2 voltage offset applied on top of all other voltage modes.
- UINT16 [SaVoltageOverride](#)
The SA/Uncore voltage override which is applied to the entire range of uncore frequencies.
- UINT16 [SaExtraTurboVoltage](#)
Adaptive Turbo voltage target used to define the interpolation voltage point when the SA/Uncore is operating in turbo mode range.
- UINT8 [RingVfPointOffsetMode](#)
Selects Ring Voltage & Frequency Point Offset between Legacy and Selection modes.
- UINT8 [RingVfPointCount](#)
Number of supported Ring Voltage & Frequency Point.
- UINT8 [RsvdByte5](#) [2]
Reserved for dword alignment.
- INT16 [RingVfPointOffset](#) [CPU_OC_MAX_VF_POINTS]
Array used to specifies the Ring Voltage Offset applied to the each selected VF Point.
- UINT8 [RsvdByte6](#) [2]
Reserved for dword alignment.
- UINT8 [RingVfPointRatio](#) [CPU_OC_MAX_VF_POINTS]

- Array for the each selected VF Point to display the Ring Ration.*
- UINT8 [AtomPllVoltageOffset](#)
*Atom PLL voltage offset. **0: No offset.** Range 0-15 in 17.5mv units.*
 - UINT32 [CpuBclkOcFrequency](#)
*CPU BCLK OC Frequency in 10KHz units increasing. Value 9800 (10KHz) = 98MHz **0 - Auto.** Range is 8000-50000 (10KHz)*
 - UINT32 [DisablePerCoreMask](#)
Core disable mask is a bitwise indication of which core should be disabled.
 - UINT32 [DisablePerAtomMask](#)
Small Core disable mask is a bitwise indication of which atom should be disabled, while ActiveSmallCoreCount is to set the number of active atoms.
 - UINT8 [PvdRatioThreshold](#)
The Pvd Ratio Threshold is the threshold value for input ratio (P0 to Pn) to select the multiplier so that the output is within the DCO frequency range.
 - UINT8 [PvdMode](#)
PVD Mode selects Static PVD ratio (when PVD THRESHOLD is 0).
 - UINT8 [FllOverclockMode](#)
FLL_OVERCLOCK_MODE: 0x0 = no overclocking, 0x1 = ratio overclocking with nominal (0.5-1x) reference clock frequency, 0x2 = BCLK overclocking with elevated (1-3x) reference clock frequency, 0x3 = BCLK overclocking with extreme elevated (3-5x) reference clock frequency and ratio limited to 63.
 - UINT8 [PllMaxBandingRatio](#)
The PLL Max Banding Ratio is to allow setting a new PLL Max Banding Ratio for IA domains.
 - INT16 [PerAtomClusterVoltageOffset](#) [CPU_MAX_ATOM_CLUSTERS]
Deprecated from revision 34.
 - UINT32 [SocBclkOcFrequency](#)
*SOC BCLK OC Frequency in 10KHz units increasing. Value 9800 (10KHz) = 98MHz **0 - Auto.** Range is 4000-100000 (10KHz)*
 - UINT16 [IalccMax](#)
Voltage Regulator Current Limit (Icc Max) for IA and GT.
 - UINT8 [TvbDownBinsTempThreshold0](#)
Down Bins (delta) for Temperature Threshold 0.
 - UINT8 [TvbTempThreshold0](#)
TVB Temp (degrees C) - Temperature Threshold 0.
 - UINT8 [TvbTempThreshold1](#)
TVB Temp (degrees C) - Temperature Threshold 1.
 - UINT8 [TvbDownBinsTempThreshold1](#)
Down Bins (delta) for Temperature Threshold 1.

13.111.1 Detailed Description

Overclocking Configuration Structure.

Revision 1:

- Initial version. **Revision 2**
- Add PerCoreHiDisable **Revision 3**
- Add Avx2VoltageScaleFactor and Avx512VoltageScaleFactor **Revision 4**
- Add CoreVfPointOffsetMode & CoreVfPointOffset & CoreVfPointRatio & CoreVfPointCount **Revision 5**
- Change OcLock default to 'Enabled' **Revision 6**

- Add CoreVfScope & PerCoreVoltageOffset **Revision 7**
- Add AtomMaxOcRatio **Revision 8**
- Add PerCoreRatioOverride and PerCoreRatio for Per Core PState overlocking. **Revision 9** Add UnlimitedIccMax **Revision 10** Add Atom Voltage configurations **Revision 11** Add SA Voltage adaptive and override configurations **Revision 12** Add RingVfPointOffsetMode & RingVfPointOffset & RingVfPointRatio & RingVfPointCount **Revision 13** Add CpuBclkOcFrequency **Revision 14**:
- Add DisablePerCoreMask & DisablePerAtomMask. **Revision 15** Add CoreRatioExtensionMode **Revision 16** Add PvdRatioThreshold **Revision 17** Add IaCepEnable & GtCepEnable & SaCepEnable & IaSirpEnable **Revision 18** Add Per Atom Cluster VF support Corrected the policy name of AtomMaxOcRatio to be AtomL2MaxOcRatio **Revision 19** Modify PllVoltageOffset of each Domain Range in 0 - 15 instead of 0 - 63 **Revision 20** Add ltdThrottleEnable **Revision 21** Add AtomClusterRatio & Correct max big core number to be 8 **Revision 22** Add SocBclkOcFrequency **Revision 23** Add AtomPllVoltageOffset **Revision 24** Deprecated FivrFaults, FivrEfficiency. **Revision 25** Add DlvrBypassModeEnable. **Revision 26** Add PvdMode and FllOverclockMode **Revision 27** Deprecated AtomL2MaxOcRatio **Revision 28** Deprecated Avx512VoltageScaleFactor and Avx3RatioOffset **Revision 29** Add SaPllFreqOverride **Revision 30** Add PllMaxBandingRatio **Revision 31** Add TscDisableHwFixup **Revision 32** Add lalccUnlimitedMode & lalccMax & GtlccUnlimitedMode & GtlccMax and deprecated UnlimitedIccMax **Revision 33** Add TvbDownBinsTempThreshold0, TvbDownBinsTempThreshold1, TvbTempThreshold0, TvbTempThreshold1 **Revision 34** Deprecated PllMaxBandingRatio **Revision 35** Add FllOcModeEn **Revision 36** Deprecated DisablePerAtomMask (DisablePerCoreMask is used for both the cores) **Revision 37** Add Etvb **Revision 38** Add UnderVoltProtection **Revision 39** Deprecated DlvrBypassModeEnable **Revision 40** Add FastThrottleThreshold **Revision 41** Default values of lalccMax and GtlccMax are changed to 0 (no override) instead of 4.

Definition at line 140 of file OverclockingConfig.h.

13.111.2 Member Data Documentation

13.111.2.1 AtomClusterRatio

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::AtomClusterRatio[CPU_MAX_ATOM_CLUSTERS]
```

The Atom cluster max ratio overrides.

Default: 0 Range: 0 to 120.

Definition at line 356 of file OverclockingConfig.h.

13.111.2.2 AtomL2MaxOcRatio

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::AtomL2MaxOcRatio
```

Maximum ATOM L2 turbo ratio override allows to increase CPU core frequency beyond the fused max turbo ratio limit (P0).

0. no override/HW defaults.. non-turbo max - 120. deprecated from version 27

Definition at line 218 of file OverclockingConfig.h.

13.111.2.3 AtomL2VoltageAdaptive

UINT16 OVERCLOCKING_PREMEM_CONFIG::AtomL2VoltageAdaptive

Adaptive Turbo voltage target used to define the interpolation voltage point when the atom is operating in turbo mode range.

Used when AtomL2VoltageMode = Adaptive. **0. no override.** Range 0-2000mV.

Definition at line 375 of file OverclockingConfig.h.

13.111.2.4 AtomL2VoltageMode

UINT8 OVERCLOCKING_PREMEM_CONFIG::AtomL2VoltageMode

Atom L2 voltage mode, specifies which voltage mode the atom processor will be operating.

0: Adaptive Mode allows the atom processor to interpolate a voltage curve when beyond fused P0 range; **1: Override**, sets one voltage for for the entire frequency range, Pn-P0.

Definition at line 362 of file OverclockingConfig.h.

13.111.2.5 AtomL2VoltageOffset

INT16 OVERCLOCKING_PREMEM_CONFIG::AtomL2VoltageOffset

The atom L2 voltage offset applied on top of all other voltage modes.

This offset is applied over the entire frequency range. This is a 2's complement number in mV units. **Default: 0** Range: -1000 to 1000.

Definition at line 380 of file OverclockingConfig.h.

13.111.2.6 AtomL2VoltageOverride

UINT16 OVERCLOCKING_PREMEM_CONFIG::AtomL2VoltageOverride

The atom L2 voltage override which is applied to the entire range of atom core frequencies.

Used when AtomL2VoltageMode = Override. **0. no override.** Range 0-2000 mV.

Definition at line 369 of file OverclockingConfig.h.

13.111.2.7 Avx2VoltageScaleFactor

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::Avx2VoltageScaleFactor
```

Avx2 Voltage Guardband Scale Factor This controls the AVX2 Voltage Guardband Scale factor applied to AVX2 workloads.

Valid range is 0-200 in 1/100 units, where a value of 125 would apply a 1.25 scale factor. A value of 0 means no scale factor applied (no change to voltage on AVX commands) A value of 100 applies the default voltage guardband values (1.0 factor). A value > 100 will increase the voltage guardband on AVX2 workloads. A value < 100 will decrease the voltage guardband on AVX2 workloads.

0. No scale factor applied

Definition at line 318 of file OverclockingConfig.h.

13.111.2.8 Avx512VoltageScaleFactor

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::Avx512VoltageScaleFactor
```

Avx512 Voltage Guardband Scale Factor This controls the AVX512 Voltage Guardband Scale factor applied to AVX512 workloads.

Valid range is 0-200 in 1/100 units, where a value of 125 would apply a 1.25 scale factor. A value of 0 means no scale factor applied (no change to voltage on AVX commands) A value of 100 applies the default voltage guardband values (1.0 factor). A value > 100 will increase the voltage guardband on AVX512 workloads. A value < 100 will decrease the voltage guardband on AVX512 workloads.

0. No scale factor applied deprecated from version 28

Definition at line 330 of file OverclockingConfig.h.

13.111.2.9 CoreMaxOcRatio

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::CoreMaxOcRatio
```

Maximum core turbo ratio override allows to increase CPU core frequency beyond the fused max turbo ratio limit (P0).

0. no override/HW defaults.. Range non-turbo max - 120 (Previously max turbo is 83).

Definition at line 207 of file OverclockingConfig.h.

13.111.2.10 CoreVfConfigScope

UINT32 OVERCLOCKING_PREMEM_CONFIG::CoreVfConfigScope

Core VF Configuration Scope.

Allows both all-core VF curve or per-core VF curve configuration. **0: All-core Scope, setting the VF curve for all cores.** 1: Per-core Scope, setting the VF curve per-core.

Definition at line 192 of file OverclockingConfig.h.

13.111.2.11 CoreVfPointOffset

INT16 OVERCLOCKING_PREMEM_CONFIG::CoreVfPointOffset[CPU_OC_MAX_VF_POINTS]

Array used to specifies the Core Voltage Offset applied to the each selected VF Point.

This voltage is specified in millivolts.

Definition at line 335 of file OverclockingConfig.h.

13.111.2.12 CoreVfPointOffsetMode

UINT32 OVERCLOCKING_PREMEM_CONFIG::CoreVfPointOffsetMode

Deprecated < Fivr Efficiency Management. 0: Disabled, **1: Enabled.**

Selects Core Voltage & Frequency Offset between Legacy and Selection modes. Need Reset System after enabling OverClocking Feature to Initialize the default value. **0: In Legacy Mode, setting a global offset for the entire VF curve.** 1: In Selection Mode, setting a selected VF point.

Definition at line 185 of file OverclockingConfig.h.

13.111.2.13 CoreVoltageAdaptive

UINT16 OVERCLOCKING_PREMEM_CONFIG::CoreVoltageAdaptive

Adaptive Turbo voltage target used to define the interpolation voltage point when the cpu is operating in turbo mode range.

Used when CoreVoltageMode = Adaptive. **0. no override.** Range 0-2000mV.

Definition at line 230 of file OverclockingConfig.h.

13.111.2.14 CoreVoltageMode

```
UINT32 OVERCLOCKING_PREMEM_CONFIG::CoreVoltageMode
```

Core voltage mode, specifies which voltage mode the processor will be operating.

0: Adaptive Mode allows the processor to interpolate a voltage curve when beyond fused P0 range; 1: Override, sets one voltage for for the entire frequency range, Pn-P0.

Definition at line 156 of file OverclockingConfig.h.

13.111.2.15 CoreVoltageOffset

```
INT16 OVERCLOCKING_PREMEM_CONFIG::CoreVoltageOffset
```

The core voltage offset applied on top of all other voltage modes.

This offset is applied over the entire frequency range. This is a 2's complement number in mV units. **Default: 0**
Range: -1000 to 1000.

Definition at line 235 of file OverclockingConfig.h.

13.111.2.16 CoreVoltageOverride

```
UINT16 OVERCLOCKING_PREMEM_CONFIG::CoreVoltageOverride
```

The core voltage override which is applied to the entire range of cpu core frequencies.

Used when CoreVoltageMode = Override. **0. no override.** Range 0-2000 mV.

Definition at line 224 of file OverclockingConfig.h.

13.111.2.17 DisablePerAtomMask

```
UINT32 OVERCLOCKING_PREMEM_CONFIG::DisablePerAtomMask
```

Small Core disable mask is a bitwise indication of which atom should be disabled, while ActiveSmallCoreCount is to set the number of active atoms.

Each bit represents logical core id: 1 - Disable 0 - Ignored Defalut is 0, means no Small Core disable mask setting.

Definition at line 432 of file OverclockingConfig.h.

13.111.2.18 DisablePerCoreMask

```
UINT32 OVERCLOCKING_PREMEM_CONFIG::DisablePerCoreMask
```

Core disable mask is a bitwise indication of which core should be disabled.

Each bit represents physical core id for both P-core and E-core: 1 - Disable 0 - Ignored Default is 0, means no Core disable mask setting.

Definition at line 424 of file OverclockingConfig.h.

13.111.2.19 FastThrottleThreshold

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::FastThrottleThreshold
```

FastThrottleThreshold.

Specified value for max allowed temperature when cores throttle. Support FastThrottleThreshold in the range of 63 to 115 deg Celsius. **Default: 0 Hardware Defaults** Range 63 to 115. 0 = Keep HW default.

Definition at line 280 of file OverclockingConfig.h.

13.111.2.20 IaIccMax

```
UINT16 OVERCLOCKING_PREMEM_CONFIG::IaIccMax
```

Voltage Regulator Current Limit (Icc Max) for IA and GT.

This value represents the Maximum instantaneous current allowed at any given time. The value is represented in 1/4 A increments. A value of 400 = 100A. **0(no override)**. Range is 4-2047.

Definition at line 464 of file OverclockingConfig.h.

13.111.2.21 OcSupport

```
UINT32 OVERCLOCKING_PREMEM_CONFIG::OcSupport
```

Overclocking support.

This controls whether OC mailbox transactions are sent. If disabled, all policies in this config block besides Oc↔Support and OcLock will be ignored. **0: Disable**; 1: Enable.

Note

If PcdOverclockEnable is disabled, this should also be disabled.

Definition at line 149 of file OverclockingConfig.h.

13.111.2.22 PerCoreHtDisable

```
UINT16 OVERCLOCKING_PREMEM_CONFIG::PerCoreHtDisable
```

Defines the per-core HT disable mask where: 1 - Disable selected logical core HT, 0 - is ignored.

Input is in HEX and each bit maps to a logical core. Ex. A value of '1F' would disable HT for cores 4,3,2,1 and 0. **Default is 0**, all cores have HT enabled. Range is 0 - 0x1FF. You can only disable up to MAX_CORE_COUNT - 1.

Definition at line 306 of file OverclockingConfig.h.

13.111.2.23 PerCoreVoltageOffset

```
INT16 OVERCLOCKING_PREMEM_CONFIG::PerCoreVoltageOffset[CPU_MAX_BIG_CORES]
```

Array used to specifies the selected Core Offset Voltage.

This voltage is specified in millivolts.

Definition at line 349 of file OverclockingConfig.h.

13.111.2.24 PllMaxBandingRatio

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::PllMaxBandingRatio
```

The PLL Max Banding Ratio is to allow setting a new PLL Max Banding Ratio for IA domains.

0. Hardware Default/No Override. Minimum must be greater than P0 and maximum is 120.

Definition at line 457 of file OverclockingConfig.h.

13.111.2.25 PvdMode

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::PvdMode
```

PVD Mode selects Static PVD ratio (when PVD THRESHOLD is 0).

0x0 = div-1 (VCO = Output clock), 0x1 = div-2 (VCO = 2x Output clock) 0x2 = div-4 (VCO = 4x Output clock), 0x3 = div-8 (VCO = 8x Output clock)

Definition at line 445 of file OverclockingConfig.h.

13.111.2.26 PvdRatioThreshold

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::PvdRatioThreshold
```

The Pvd Ratio Threshold is the threshold value for input ratio (P0 to Pn) to select the multiplier so that the output is within the DCO frequency range.

This threshold is applied to all the domains (Cores, Ring, SA, Memory and GT). **0. Default/No Override.** Range 1-63.

Definition at line 439 of file OverclockingConfig.h.

13.111.2.27 RingDownBin

```
UINT32 OVERCLOCKING_PREMEM_CONFIG::RingDownBin
```

Ring Downbin enable/disable.

When enabled, the CPU will force the ring ratio to be lower than the core ratio. Disabling will allow the ring and core ratios to run at the same frequency. Uses OC Mailbox command 0x19. **0: Disables Ring Downbin feature. 1: Enables Ring downbin feature.**

Definition at line 168 of file OverclockingConfig.h.

13.111.2.28 RingMaxOcRatio

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::RingMaxOcRatio
```

Maximum ring ratio override allows to increase CPU ring frequency beyond the fused max ring ratio limit.

0. no override/HW defaults.. Range non-turbo max - 85.

Definition at line 213 of file OverclockingConfig.h.

13.111.2.29 RingVfPointOffset

```
INT16 OVERCLOCKING_PREMEM_CONFIG::RingVfPointOffset[CPU_OC_MAX_VF_POINTS]
```

Array used to specifies the Ring Voltage Offset applied to the each selected VF Point.

This voltage is specified in millivolts.

Definition at line 409 of file OverclockingConfig.h.

13.111.2.30 RingVfPointOffsetMode

UINT8 OVERCLOCKING_PREMEM_CONFIG::RingVfPointOffsetMode

Selects Ring Voltage & Frequency Point Offset between Legacy and Selection modes.

Need Reset System after enabling OverClocking Feature to Initialize the default value. **0: In Legacy Mode, setting a global offset for the entire VF curve.** 1: In Selection modes, setting a selected VF point.

Definition at line 399 of file OverclockingConfig.h.

13.111.2.31 RingVoltageAdaptive

UINT16 OVERCLOCKING_PREMEM_CONFIG::RingVoltageAdaptive

Adaptive Turbo voltage target used to define the interpolation voltage point when the ring is operating in turbo mode range.

Used when RingVoltageMode = Adaptive. **0. no override.** Range 0-2000mV.

Definition at line 247 of file OverclockingConfig.h.

13.111.2.32 RingVoltageMode

UINT32 OVERCLOCKING_PREMEM_CONFIG::RingVoltageMode

Ring voltage mode, specifies which voltage mode the processor will be operating.

0: Adaptive Mode allows the processor to interpolate a voltage curve when beyond fused P0 range; 1: Override, sets one voltage for for the entire frequency range, Pn-P0.

Definition at line 174 of file OverclockingConfig.h.

13.111.2.33 RingVoltageOffset

INT16 OVERCLOCKING_PREMEM_CONFIG::RingVoltageOffset

The ring voltage offset applied on top of all other voltage modes.

This offset is applied over the entire frequency range. This is a 2's complement number in mV units. **Default: 0** Range: -1000 to 1000.

Definition at line 252 of file OverclockingConfig.h.

13.111.2.34 RingVoltageOverride

UINT16 OVERCLOCKING_PREMEM_CONFIG::RingVoltageOverride

The ring voltage override which is applied to the entire range of cpu ring frequencies.

Used when RingVoltageMode = Override. **0. no override.** Range 0-2000 mV.

Definition at line 241 of file OverclockingConfig.h.

13.111.2.35 SaExtraTurboVoltage

UINT16 OVERCLOCKING_PREMEM_CONFIG::SaExtraTurboVoltage

Adaptive Turbo voltage target used to define the interpolation voltage point when the SA/Uncore is operating in turbo mode range.

Used when SaVoltageMode = Adaptive. **0. no override.** Range 0-2000mV.

Definition at line 392 of file OverclockingConfig.h.

13.111.2.36 SaVoltageMode

UINT32 OVERCLOCKING_PREMEM_CONFIG::SaVoltageMode

SA/Uncore voltage mode, specifies which voltage mode the processor will be operating.

0: Adaptive Mode allows the processor to interpolate a voltage curve when beyond fused P0 range; 1: Override, sets one voltage for for the entire frequency range, Pn-P0.

Definition at line 200 of file OverclockingConfig.h.

13.111.2.37 SaVoltageOverride

UINT16 OVERCLOCKING_PREMEM_CONFIG::SaVoltageOverride

The SA/Uncore voltage override which is applied to the entire range of uncore frequencies.

Used when SaVoltageMode = Override. **0. no override.** Range 0-2000 mV.

Definition at line 386 of file OverclockingConfig.h.

13.111.2.38 TjMaxOffset

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::TjMaxOffset
```

TjMax Offset.

Specified value here is clipped by pCode (125 - TjMax Offset) to support TjMax in the range of 62 to 115 deg Celsius. **Default: 0 Hardware Defaults** Range 10 to 63. 0 = No offset / Keep HW default.

Definition at line 275 of file OverclockingConfig.h.

13.111.2.39 TvbDownBinsTempThreshold0

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::TvbDownBinsTempThreshold0
```

Down Bins (delta) for Temperature Threshold 0.

When running above Temperature Threshold 0, the ratio will be clipped by MAX_RATIO[n]-This value, when Thermal turbo is enabled from OCMB 0x18 Bit 2 = 0 (Enabled). **1** . Range 0-10.

Definition at line 471 of file OverclockingConfig.h.

13.111.2.40 TvbDownBinsTempThreshold1

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::TvbDownBinsTempThreshold1
```

Down Bins (delta) for Temperature Threshold 1.

When running above Temperature Threshold 1, the ratio will be clipped by MAX_RATIO[n]-Down Bin Threshold 0-This value, when Thermal turbo is enabled from OCMB 0x18 Bit 2 = 0 (Enabled). **2** . Range 0-10.

Definition at line 489 of file OverclockingConfig.h.

13.111.2.41 TvbRatioClipping

```
UINT32 OVERCLOCKING_PREMEM_CONFIG::TvbRatioClipping
```

This service controls Core frequency reduction caused by high package temperatures for processors that implement the Intel Thermal Velocity Boost (TVB) feature.

It is required to be disabled for supporting overclocking at frequencies higher than the default max turbo frequency. **0: Disables TVB ratio clipping.** 1: Enables TVB ratio clipping.

Definition at line 288 of file OverclockingConfig.h.

13.111.2.42 TvbTempThreshold0

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::TvbTempThreshold0
```

TVB Temp (degrees C) - Temperature Threshold 0.

Running ABOVE this temperature will clip delta Down Bins for Threshold 0 from the resolved OC Ratio, when Thermal turbo is enabled from OCMB 0x18 Bit 2 = 0 (Enabled). **70** . Range 0-100.

Definition at line 477 of file OverclockingConfig.h.

13.111.2.43 TvbTempThreshold1

```
UINT8 OVERCLOCKING_PREMEM_CONFIG::TvbTempThreshold1
```

TVB Temp (degrees C) - Temperature Threshold 1.

Running ABOVE this temperature, clip delta Down Bins for Threshold 1 from the resolved OC Ratio, when Thermal turbo is enabled from OCMB 0x18 Bit 2 = 0 (Enabled). **100** . Range 0-100.

Definition at line 483 of file OverclockingConfig.h.

13.111.2.44 TvbVoltageOptimization

```
UINT32 OVERCLOCKING_PREMEM_CONFIG::TvbVoltageOptimization
```

This service controls thermal based voltage optimizations for processors that implement the Intel Thermal Velocity Boost (TVB) feature.

0: Disables TVB voltage optimization. **1: Enables TVB voltage optimization.**

Definition at line 294 of file OverclockingConfig.h.

The documentation for this struct was generated from the following file:

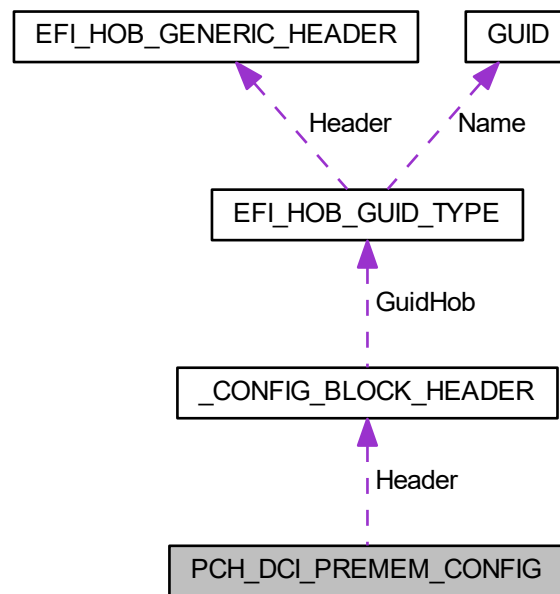
- [OverclockingConfig.h](#)

13.112 PCH_DCI_PREMEM_CONFIG Struct Reference

The [PCH_DCI_PREMEM_CONFIG](#) block describes policies related to Direct Connection Interface (DCI)

```
#include <DciConfig.h>
```

Collaboration diagram for PCH_DCI_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- `UINT8` [DciEn](#)
DCI enable.
- `UINT8` [DciDbcMode](#)
USB DbC enable mode.
- `UINT8` [DciModphyPg](#)
Deprecated.
- `UINT8` [DciUsb3TypecUfpDbg](#)
USB3 Type-C UFP2DFP kenel / platform debug support.
- `UINT8` [DciClkEnable](#)
Determine if to enable or disable DCI clock request in lowest power state.
- `UINT8` [KeepEarlyTrace](#)
Early trace is activated by default.

13.112.1 Detailed Description

The [PCH_DCI_PREMEM_CONFIG](#) block describes policies related to Direct Connection Interface (DCI)

Revision 1:

- Initial version. **Revision 2:**
- Added DciModphyPg
- change to use data in byte unit rather than bit-field **Revision 3:**
- Deprecated DciModphyPg
- Added DciClkEnable, KeepEarlyTrace

Definition at line 71 of file DciConfig.h.

13.112.2 Member Data Documentation

13.112.2.1 DciClkEnable

```
UINT8 PCH_DCI_PREMEM_CONFIG::DciClkEnable
```

Determine if to enable or disable DCI clock request in lowest power state.

0:Disabled; **1:Enabled**

Definition at line 99 of file DciConfig.h.

13.112.2.2 DciDbcMode

```
UINT8 PCH_DCI_PREMEM_CONFIG::DciDbcMode
```

USB DbC enable mode.

Disabled: Clear both USB2/3DBCEN; USB2: Set USB2DBCEN; USB3: Set USB3DBCEN; Both: Set both USB2/3DBCEN; No Change: Comply with HW value Refer to definition of DCI_USB_DBC_MODE for supported settings. 0:Disabled; 1:USB2; 2:USB3; 3:Both; **4:No Change**

Definition at line 85 of file DciConfig.h.

13.112.2.3 DciEn

```
UINT8 PCH_DCI_PREMEM_CONFIG::DciEn
```

DCI enable.

Determine if to enable DCI debug from host. **0:Disabled**; 1:Enabled

Definition at line 78 of file DciConfig.h.

13.112.2.4 DciUsb3TypecUfpDbg

```
UINT8 PCH_DCI_PREMEM_CONFIG::DciUsb3TypecUfpDbg
```

USB3 Type-C UFP2DFP kenel / platform debug support.

No change will do nothing to UFP2DFP configuration. When enabled, USB3 Type C UFP (upstream-facing port) may switch to DFP (downstream-facing port) for first connection. It must be enabled for USB3 kernel(kernel mode debug) and platform debug(DFx, DMA, Trace) over UFP Type-C receptacle. Refer to definition of DCI_USB_TYP↔E_C_DEBUG_MODE for supported settings. 0:Disabled; 1:Enabled; **2:No Change**

Definition at line 94 of file DciConfig.h.

13.112.2.5 KeepEarlyTrace

```
UINT8 PCH_DCI_PREMEM_CONFIG::KeepEarlyTrace
```

Early trace is activated by default.

Enable to keep early trace data and keep tracing; disable to stop tracing; When debug probe is connected and granted, force to no change, let host tool to set it. **0: Disable**; 1: Enable early trace; 2:No Change;

Definition at line 106 of file DciConfig.h.

The documentation for this struct was generated from the following file:

- [DciConfig.h](#)

13.113 PCH_DEVICE_INTERRUPT_CONFIG Struct Reference

The [PCH_DEVICE_INTERRUPT_CONFIG](#) block describes interrupt pin, IRQ and interrupt mode for PCH device.

```
#include <InterruptConfig.h>
```

Public Attributes

- [UINT8 Device](#)
Device number.
- [UINT8 Function](#)
Device function.
- [UINT8 IntX](#)
Interrupt pin: INTA-INTD (see PCH_INT_PIN)
- [UINT8 Irq](#)
IRQ to be set for device.

13.113.1 Detailed Description

The [PCH_DEVICE_INTERRUPT_CONFIG](#) block describes interrupt pin, IRQ and interrupt mode for PCH device.

Definition at line 57 of file `InterruptConfig.h`.

The documentation for this struct was generated from the following file:

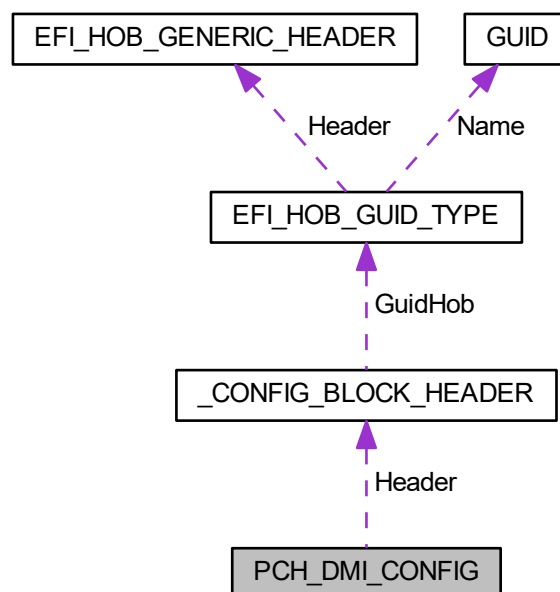
- [InterruptConfig.h](#)

13.114 PCH_DMI_CONFIG Struct Reference

The [PCH_DMI_CONFIG](#) block describes the expected configuration of the PCH for DMI.

```
#include <PchDmiConfig.h>
```

Collaboration diagram for PCH_DMI_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [PwrOptEnable](#): 1
*0: **Disable**; 1: Enable DMI Power Optimizer on PCH side.*
- UINT32 [DmiAspmCtrl](#): 8
*ASPM configuration on the PCH side of the DMI/OPI Link. Default is **PchPcieAspmAutoConfig***
- UINT32 [CwbEnable](#): 1
*0: Disable; 1: **Enable** Central Write Buffer feature configurable and enabled by default*
- UINT32 [L1RpCtl](#): 1
*0: Disable; 1: **Enable** Allow DMI enter L1 when all root ports are in L1, L0s or link down. Disabled by default.*
- UINT32 [DmiPowerReduction](#): 1
When set to TRUE turns on:
- UINT32 [ClockGating](#): 1
0: Disable; 1: Enable clock gating.
- UINT32 [Rsvdbits](#): 19
Reserved bits.

13.114.1 Detailed Description

The [PCH_DMI_CONFIG](#) block describes the expected configuration of the PCH for DMI.

Revision 1:

- Initial version.

Definition at line 73 of file PchDmiConfig.h.

13.114.2 Member Data Documentation

13.114.2.1 DmiPowerReduction

```
UINT32 PCH_DMI_CONFIG::DmiPowerReduction
```

When set to TRUE turns on:

- L1 State Controller Power Gating
- L1 State PHY Data Lane Power Gating
- PHY Common Lane Power Gating
- Hardware Autonomous Enable
- PMC Request Enable and Sleep Enable

Definition at line 88 of file PchDmiConfig.h.

The documentation for this struct was generated from the following file:

- [PchDmiConfig.h](#)

13.115 PCH_DMI_EQ_PARAM Struct Reference

Lane specific Dmi Gen3, Gen4 equalization parameters.

```
#include <PchDmiConfig.h>
```

Public Attributes

- [UINT8 Cm](#)
Coefficient C-1.
- [UINT8 Cp](#)
Coefficient C+1.
- [UINT8 Rsvd0](#) [2]
Reserved bytes.

13.115.1 Detailed Description

Lane specific Dmi Gen3, Gen4 equalization parameters.

Definition at line 54 of file PchDmiConfig.h.

The documentation for this struct was generated from the following file:

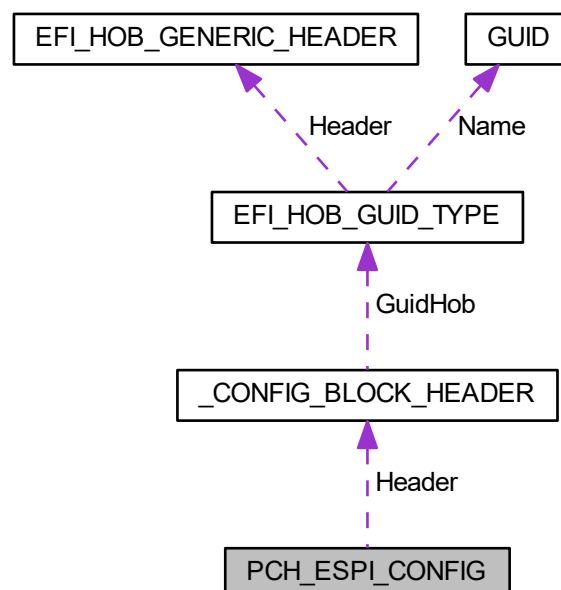
- [PchDmiConfig.h](#)

13.116 PCH_ESPI_CONFIG Struct Reference

This structure contains the policies which are related to ESPI.

```
#include <EspiConfig.h>
```

Collaboration diagram for PCH_ESPI_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [LgmrEnable](#): 1
LPC (eSPI) Memory Range Decode Enable.
- UINT32 [BmeMasterSlaveEnabled](#): 1
eSPI host and device BME settings.
- UINT32 [HostC10ReportEnable](#): 1
*Host HOST_C10 (Virtual Wire) to device Enable (VWHC10OE) 0b: **Disable HOST_C10 reporting (HOST_C10 indication from PMC is ignored)** 1b: Enable HOST_C10 reporting to device via eSPI Virtual Wire (upon receiving a HOST_C10 indication from PMC)*
- UINT32 [LockLinkConfiguration](#): 1
*eSPI Link Configuration Lock (SBLCL) If set to TRUE then communication through SET_CONFIG/GET_CONFIG to eSPI device addresses from range 0x0 - 0x7FF 1: **TRUE**, 0: **FALSE***
- UINT32 [EspPmHAE](#): 1
Hardware Autonomous Enable (HAE) If set to TRUE, then the IP may request a PG whenever it is idle.
- UINT32 [RsvdBits](#): 27
Reserved bits.

13.116.1 Detailed Description

This structure contains the policies which are related to ESPI.

Revision 1:

- Initial revision **Revision 2:**
- Added LockLinkConfiguration field to config block

Definition at line 51 of file EspiConfig.h.

13.116.2 Member Data Documentation

13.116.2.1 BmeMasterSlaveEnabled

```
UINT32 PCH_ESPI_CONFIG::BmeMasterSlaveEnabled
```

eSPI host and device BME settings.

When TRUE, then the BME bit enabled in eSPI host and device. 0: FALSE, 1: **TRUE**

Definition at line 64 of file EspiConfig.h.

13.116.2.2 LgmrEnable

UINT32 PCH_ESPI_CONFIG::LgmrEnable

LPC (eSPI) Memory Range Decode Enable.

When TRUE, then the range specified in PCLGMR[31:16] is enabled for decoding to LPC (eSPI). **0: FALSE**, 1: TRUE

Definition at line 58 of file EspiConfig.h.

The documentation for this struct was generated from the following file:

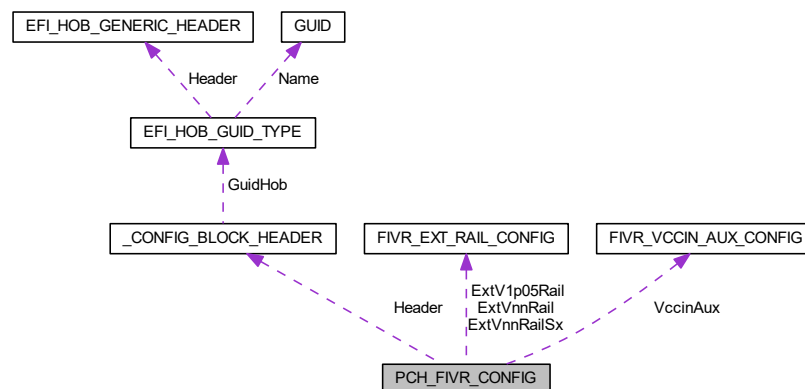
- [EspiConfig.h](#)

13.117 PCH_FIVR_CONFIG Struct Reference

The [PCH_FIVR_CONFIG](#) block describes FIVR settings.

```
#include <FivrConfig.h>
```

Collaboration diagram for PCH_FIVR_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [FIVR_EXT_RAIL_CONFIG](#) ExtV1p05Rail
External V1P05 VR rail configuration.
- [FIVR_EXT_RAIL_CONFIG](#) ExtVnnRail
External Vnn VR rail configuration.
- [FIVR_EXT_RAIL_CONFIG](#) ExtVnnRailSx
Additional External Vnn VR rail configuration that will get applied in Sx entry SMI callback.
- [FIVR_VCCIN_AUX_CONFIG](#) VccinAux
VCCIN_AUX voltage rail configuration.
- UINT32 [FivrDynPm](#): 1
Enable/Disable FIVR Dynamic Power Management Default is 1 .
- UINT32 [FivrVccstlccMaxControl](#): 1
Enable/Disable FIVR VccSt lccMax Control Default is 1 .
- UINT32 [BypassFivrConfig](#): 1
Enable/Disable Bypass of FIVR configuration Default is 0 .

13.117.1 Detailed Description

The [PCH_FIVR_CONFIG](#) block describes FIVR settings.

Definition at line 177 of file FivrConfig.h.

13.117.2 Member Data Documentation

13.117.2.1 ExtVnnRailSx

[FIVR_EXT_RAIL_CONFIG](#) `PCH_FIVR_CONFIG::ExtVnnRailSx`

Additional External Vnn VR rail configuration that will get applied in Sx entry SMI callback.

Required only if External Vnn VR needs different settings for Sx than those specified in ExtVnnRail.

Definition at line 192 of file FivrConfig.h.

The documentation for this struct was generated from the following file:

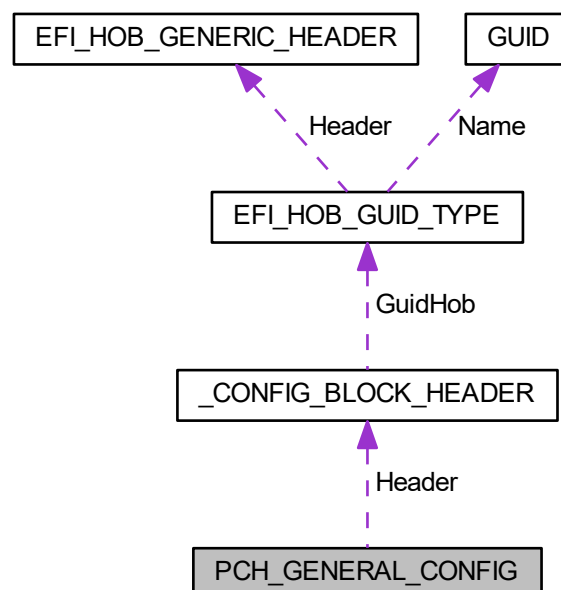
- [FivrConfig.h](#)

13.118 PCH_GENERAL_CONFIG Struct Reference

PCH General Configuration **Revision 1**: - Initial version.

```
#include <PchGeneralConfig.h>
```

Collaboration diagram for PCH_GENERAL_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [Crid](#): 1
This member describes whether or not the Compatibility Revision ID (CRID) feature of PCH should be enabled.
- UINT32 [LegacyIoLowLatency](#): 1
Set to enable low latency of legacy IO.
- UINT32 [AcpiL6dPmeHandling](#): 1
Enables _L6D ACPI handler.
- UINT32 [RsvdBits0](#): 29
Reserved bits.

13.118.1 Detailed Description

PCH General Configuration **Revision 1**: - Initial version.

Revision 2: - Added AcpiL6dPmeHandling

Definition at line 56 of file PchGeneralConfig.h.

13.118.2 Member Data Documentation

13.118.2.1 AcpiL6dPmeHandling

```
UINT32 PCH_GENERAL_CONFIG::AcpiL6dPmeHandling
```

Enables _L6D ACPI handler.

PME GPE is shared by multiple devices So BIOS must verify the same in the ASL handler by reading offset for PMEENABLE and PMESTATUS bit **0: Disable**, 1: Enable

Definition at line 77 of file PchGeneralConfig.h.

13.118.2.2 Crid

```
UINT32 PCH_GENERAL_CONFIG::Crid
```

This member describes whether or not the Compatibility Revision ID (CRID) feature of PCH should be enabled.

0: Disable; 1: Enable

Definition at line 62 of file PchGeneralConfig.h.

13.118.2.3 LegacyIoLowLatency

UINT32 PCH_GENERAL_CONFIG::LegacyIoLowLatency

Set to enable low latency of legacy IO.

Some systems require lower IO latency irrespective of power. This is a tradeoff between power and IO latency.

Note

: Once this is enabled, DmiAspm, Pcie DmiAspm in SystemAgent and ITSS Clock Gating are forced to disabled. **0: Disable**, 1: Enable

Definition at line 71 of file PchGeneralConfig.h.

The documentation for this struct was generated from the following file:

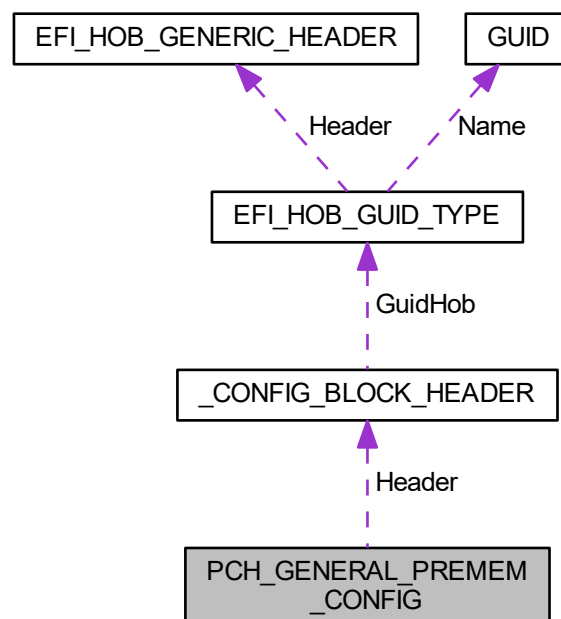
- [PchGeneralConfig.h](#)

13.119 PCH_GENERAL_PREMEM_CONFIG Struct Reference

PCH General Pre-Memory Configuration **Revision 1:** - Initial version.

```
#include <PchGeneralConfig.h>
```

Collaboration diagram for PCH_GENERAL_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [Port80Route](#): 1
Control where the Port 80h cycles are sent, 0: LPC; 1: PCI.
- UINT32 [IotgPllSscEn](#): 1
Need to disable CPU Side SSC for A0 PO.
- UINT32 [GpioOverride](#): 3
Gpio override Level – 0: Disable;.
- UINT32 [IoeDebugEn](#): 1
Enable/Disable IOE Debug.
- UINT32 [PmodeClkEn](#): 1
Enable/Disable PMODE clock.
- UINT32 [DisableDynamicTccoldHandshake](#): 1
TCSS TCCOLD disable Disable dynamic Tc-cold handshake by setting command 0x2b , subcommand 0xb to 1.
- UINT32 [RsvdBits0](#): 24
Reserved bits.

13.119.1 Detailed Description

PCH General Pre-Memory Configuration **Revision 1**: - Initial version.

Revision 2: - Added GpioOverride. **Revision 3**: - Added IoeDebugEn, PmodeClkEn **Revision 4**: - Add Disable↔ DynamicTccoldHandshake

Definition at line 88 of file PchGeneralConfig.h.

13.119.2 Member Data Documentation

13.119.2.1 DisableDynamicTccoldHandshake

```
UINT32 PCH_GENERAL_PREMEM_CONFIG::DisableDynamicTccoldHandshake
```

TCSS TCCOLD disable Disable dynamic Tc-cold handshake by setting command 0x2b , subcommand 0xb to 1.

This will disable the Tc-cold handshake. 0: **Do Nothing**; 1: Disable;

Definition at line 118 of file PchGeneralConfig.h.

13.119.2.2 GpioOverride

UINT32 PCH_GENERAL_PREMEM_CONFIG::GpioOverride

Gpio override Level – **0: Disable**;

- 1: Override Level 1 - Skips GPIO configuration in PEI/FSPM/FSPT phase
- 2: Override Level 2 - Reserved for future use

Definition at line 101 of file PchGeneralConfig.h.

13.119.2.3 IoeDebugEn

UINT32 PCH_GENERAL_PREMEM_CONFIG::IoeDebugEn

Enable/Disable IOE Debug.

When enabled, IOE D2D Dfx link and clock will keep up for debug **0: Disable**; 1: Enable

Definition at line 106 of file PchGeneralConfig.h.

13.119.2.4 PmodeClkEn

UINT32 PCH_GENERAL_PREMEM_CONFIG::PmodeClkEn

Enable/Disable PMODE clock.

When enabled, Pmode clock will toggle for XDP use **0: Disable**; 1: Enable

Definition at line 111 of file PchGeneralConfig.h.

The documentation for this struct was generated from the following file:

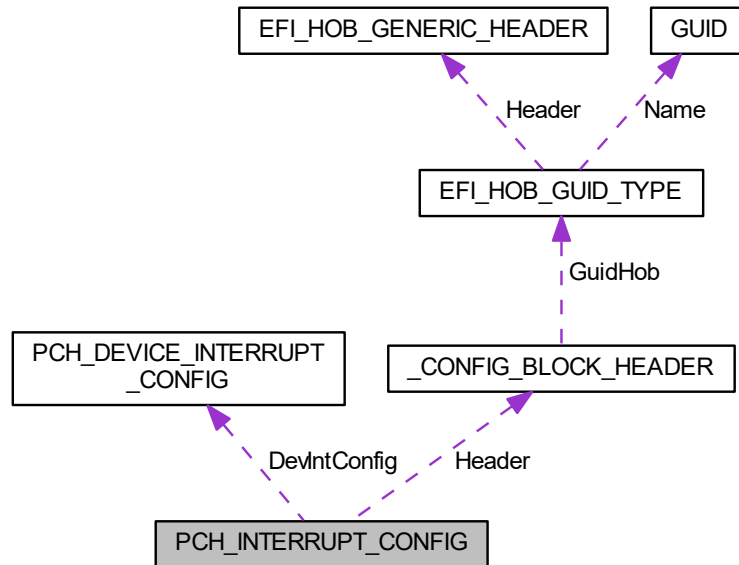
- [PchGeneralConfig.h](#)

13.120 PCH_INTERRUPT_CONFIG Struct Reference

The [PCH_INTERRUPT_CONFIG](#) block describes interrupt settings for PCH.

```
#include <InterruptConfig.h>
```

Collaboration diagram for PCH_INTERRUPT_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- `UINT8` [NumOfDevIntConfig](#)
Number of entries in DevIntConfig table.
- `UINT8` [Rsvd0](#) [3]
Reserved bytes, align to multiple 4.
- [PCH_DEVICE_INTERRUPT_CONFIG DevIntConfig](#) [`PCH_MAX_DEVICE_INTERRUPT_CONFIG`]
Array which stores PCH devices interrupts settings.
- `UINT8` [GpioIrqRoute](#)
Interrupt routing for GPIO. Default is 14.
- `UINT8` [ScIrqSelect](#)
Interrupt select for SCI. Default is 9.
- `UINT8` [TcolrqSelect](#)
Interrupt select for TCO. Default is 9.
- `UINT8` [TcolrqEnable](#)
*Enable IRQ generation for TCO. 0: **Disable**; 1: Enable.*

13.120.1 Detailed Description

The [PCH_INTERRUPT_CONFIG](#) block describes interrupt settings for PCH.

Definition at line 73 of file InterruptConfig.h.

The documentation for this struct was generated from the following file:

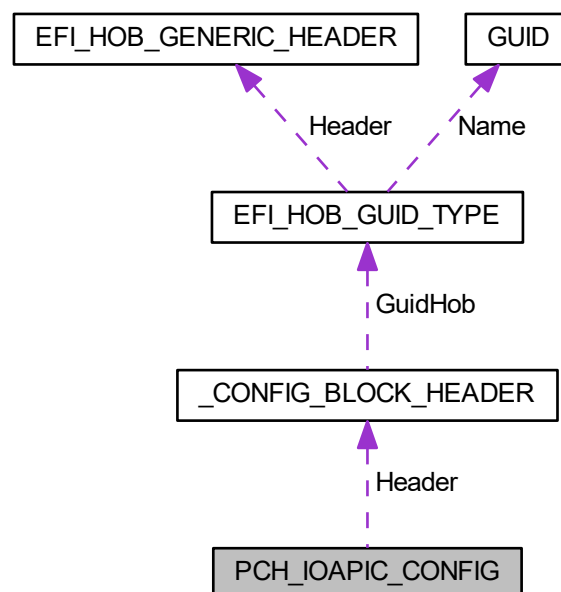
- [InterruptConfig.h](#)

13.121 PCH_IOAPIC_CONFIG Struct Reference

The [PCH_IOAPIC_CONFIG](#) block describes the expected configuration of the PCH IO APIC, it's optional and PCH code would ignore it if the BdfValid bit is not TRUE.

```
#include <IoApicConfig.h>
```

Collaboration diagram for PCH_IOAPIC_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [IoApicEntry24_119](#): 1
*0: Disable; 1: **Enable** IOAPIC Entry 24-119*
- UINT32 [Enable8254ClockGating](#): 1
Enable 8254 Static Clock Gating during early POST time.
- UINT32 [Enable8254ClockGatingOnS3](#): 1
Enable 8254 Static Clock Gating on S3 resume path.
- UINT32 [RsvdBits1](#): 29
Reserved bits.
- UINT8 [IoApicId](#)
*This member determines IOAPIC ID. Default is **0x02**.*
- UINT8 [Rsvd0](#) [3]
Reserved bytes.

13.121.1 Detailed Description

The [PCH_IOAPIC_CONFIG](#) block describes the expected configuration of the PCH IO APIC, it's optional and PCH code would ignore it if the BdfValid bit is not TRUE.

Bus:device:function fields will be programmed to the register P2SB IBDF(P2SB PCI offset R6Ch-6Dh), it's using for the following purpose: As the Requester ID when initiating Interrupt Messages to the processor. As the Completer ID when responding to the reads targeting the IOxAPI's Memory-Mapped I/O registers. This field defaults to Bus 0: Device 31: Function 0 after reset. BIOS can program this field to provide a unique Bus:Device:Function number for the internal IOxAPIC. The address resource range of IOAPIC must be reserved in E820 and ACPI as system resource.

Definition at line 57 of file IoApicConfig.h.

13.121.2 Member Data Documentation

13.121.2.1 Enable8254ClockGating

```
UINT32 PCH_IOAPIC_CONFIG::Enable8254ClockGating
```

Enable 8254 Static Clock Gating during early POST time.

0: Disable, 1: **Enable** Setting 8254CGE is required to support SLP_S0. Enable this if 8254 timer is not used. However, set 8254CGE=1 in POST time might fail to boot legacy OS using 8254 timer. Make sure it is disabled to support legacy OS using 8254 timer.

Note

: For some OS environment that it needs to set 8254CGE in late state it should set this policy to FALSE and use ItssSet8254ClockGateState (TRUE) in SMM later. This is also required during S3 resume. To avoid SMI requirement in S3 resume path, it can enable the Enable8254ClockGatingOnS3 and RC will do 8254 CGE programming in PEI during S3 resume with BOOT_SAI.

Definition at line 73 of file IoApicConfig.h.

13.121.2.2 Enable8254ClockGatingOnS3

```
UINT32 PCH_IOAPIC_CONFIG::Enable8254ClockGatingOnS3
```

Enable 8254 Static Clock Gating on S3 resume path.

0: Disable, **1: Enable** This is only applicable when Enable8254ClockGating is disabled. If Enable8254ClockGating is enabled, RC will do the 8254 CGE programming on S3 resume path as well.

Definition at line 80 of file IoApicConfig.h.

The documentation for this struct was generated from the following file:

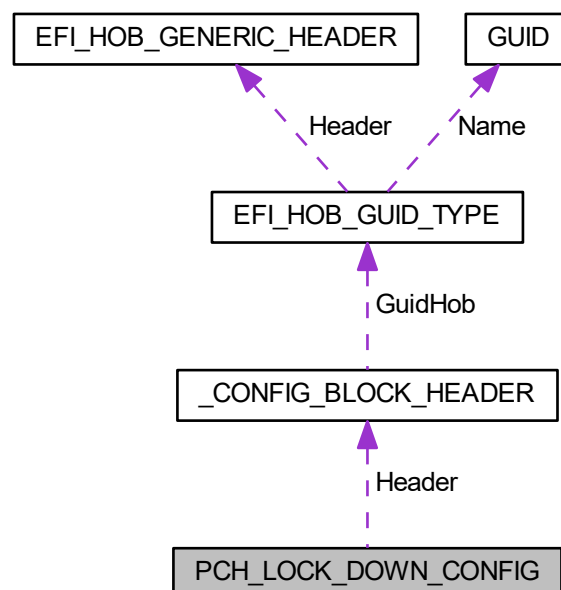
- [IoApicConfig.h](#)

13.122 PCH_LOCK_DOWN_CONFIG Struct Reference

The [PCH_LOCK_DOWN_CONFIG](#) block describes the expected configuration of the PCH for security requirement.

```
#include <LockDownConfig.h>
```

Collaboration diagram for PCH_LOCK_DOWN_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [GlobalSmi](#): 1
(Test) Enable SMI_LOCK bit to prevent writes to the Global SMI Enable bit.
- UINT32 [BiosInterface](#): 1
(Test) Enable BIOS Interface Lock Down bit to prevent writes to the Backup Control Register Top Swap bit and the General Control and Status Registers Boot BIOS Straps.
- UINT32 [BiosLock](#): 1
Enable the BIOS Lock Enable (BLE) feature and set EISS bit (D31:F5:RegDCh[5]) for the BIOS region protection.
- UINT32 [SpiEiss](#): 1
Separate EISS policy to be handled separately from BiosLock.
- UINT32 [UnlockGpioPads](#): 1
(Test) This test option when set will force all GPIO pads to be unlocked before BIOS transitions to POSTBOOT_SAI.
- UINT32 [RsvdBits0](#): 27
Reserved bits.

13.122.1 Detailed Description

The [PCH_LOCK_DOWN_CONFIG](#) block describes the expected configuration of the PCH for security requirement.

Definition at line 47 of file LockDownConfig.h.

13.122.2 Member Data Documentation

13.122.2.1 BiosInterface

```
UINT32 PCH_LOCK_DOWN_CONFIG::BiosInterface
```

(Test) Enable BIOS Interface Lock Down bit to prevent writes to the Backup Control Register Top Swap bit and the General Control and Status Registers Boot BIOS Straps.

Intel strongly recommends that BIOS sets the BIOS Interface Lock Down bit. Enabling this bit will mitigate malicious software attempts to replace the system BIOS with its own code. 0: Disable; **1: Enable.**

Definition at line 60 of file LockDownConfig.h.

13.122.2.2 BiosLock

```
UINT32 PCH_LOCK_DOWN_CONFIG::BiosLock
```

Enable the BIOS Lock Enable (BLE) feature and set EISS bit (D31:F5:RegDCh[5]) for the BIOS region protection.

When it is enabled, the BIOS Region can only be modified from SMM. If this EISS bit is set, then WPD must be a '1' and InSMM.STS must be '1' also in order to write to BIOS regions of SPI Flash. If this EISS bit is clear, then the InSMM.STS is a don't care. The BIOS must set the EISS bit while BIOS Guard support is enabled. In recovery path, platform can temporary disable EISS for SPI programming in PEI phase or early DXE phase. When PcdSmm↔VariableEnable is FALSE, to support BIOS regions update outside of SMM, the BiosLock must be set to Disabled by platform. 0: Disable; 1: **Enable**.

Definition at line 75 of file LockDownConfig.h.

13.122.2.3 GlobalSmi

```
UINT32 PCH_LOCK_DOWN_CONFIG::GlobalSmi
```

(Test) Enable SMI_LOCK bit to prevent writes to the Global SMI Enable bit.

0: Disable; 1: **Enable**.

Definition at line 52 of file LockDownConfig.h.

13.122.2.4 UnlockGpioPads

```
UINT32 PCH_LOCK_DOWN_CONFIG::UnlockGpioPads
```

(Test) This test option when set will force all GPIO pads to be unlocked before BIOS transitions to POSTBOOT_SAI.

This option should not be enabled in production configuration and used only for debug purpose when free runtime reconfiguration of GPIO pads is needed. 0: **Disable**; 1: Enable.

Definition at line 84 of file LockDownConfig.h.

The documentation for this struct was generated from the following file:

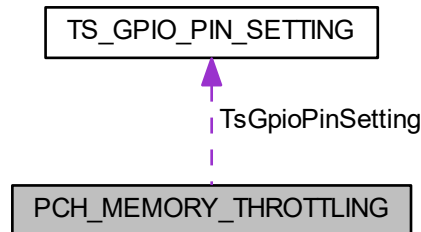
- [LockDownConfig.h](#)

13.123 PCH_MEMORY_THROTTLING Struct Reference

This structure supports an external memory thermal sensor (TS-on-DIMM or TS-on-Board).

```
#include <ThermalConfig.h>
```

Collaboration diagram for PCH_MEMORY_THROTTLING:



Public Attributes

- `UINT32 Enable`: 1
This will enable PCH memory throttling.
- `TS_GPIO_PIN_SETTING TsGpioPinSetting` [2]
GPIO_C and GPIO_D selection for memory throttling.

13.123.1 Detailed Description

This structure supports an external memory thermal sensor (TS-on-DIMM or TS-on-Board).

Definition at line 126 of file ThermalConfig.h.

13.123.2 Member Data Documentation

13.123.2.1 Enable

```
UINT32 PCH_MEMORY_THROTTLING::Enable
```

This will enable PCH memory throttling.

While this policy is enabled, must also enable EnableExtts in SA policy. **0: Disable**; 1: Enable

Definition at line 132 of file ThermalConfig.h.

13.123.2.2 TsGpioPinSetting

`TS_GPIO_PIN_SETTING` `PCH_MEMORY_THROTTLING::TsGpioPinSetting[2]`

GPIO_C and GPIO_D selection for memory throttling.

It's strongly recommended to choose GPIO_C and GPIO_D for memory throttling feature, and route EXTTS# accordingly.

Definition at line 139 of file ThermalConfig.h.

The documentation for this struct was generated from the following file:

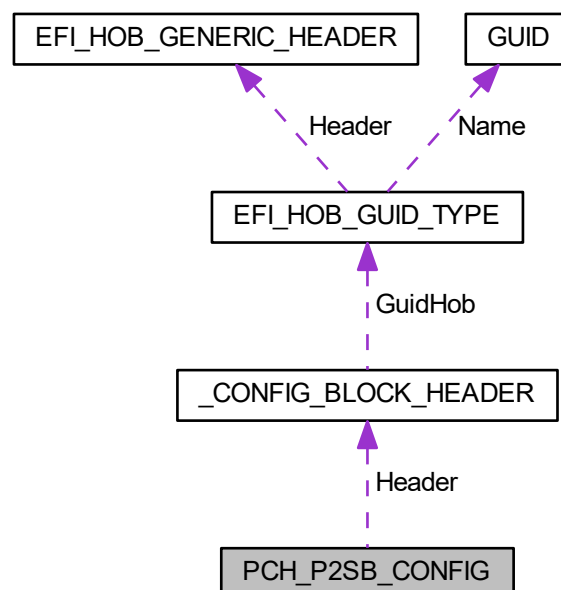
- [ThermalConfig.h](#)

13.124 PCH_P2SB_CONFIG Struct Reference

This structure contains the policies which are related to P2SB device.

```
#include <P2sbConfig.h>
```

Collaboration diagram for PCH_P2SB_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [SbAccessUnlock](#): 1
(Test) The sideband MMIO register access to specific ports will be locked before 3rd party code execution.
- UINT32 [Rsvdbits](#): 31
Reserved bits.

13.124.1 Detailed Description

This structure contains the policies which are related to P2SB device.

Definition at line 46 of file P2sbConfig.h.

13.124.2 Member Data Documentation

13.124.2.1 SbAccessUnlock

```
UINT32 PCH_P2SB_CONFIG::SbAccessUnlock
```

(Test) The sideband MMIO register access to specific ports will be locked before 3rd party code execution.

Currently it disables PSFx access. This policy unlocks the sideband MMIO space for those IPs. **0: Lock sideband access** ; 1: Unlock sideband access. NOTE: Do not set this policy "SbAccessUnlock" unless its necessary.

Definition at line 56 of file P2sbConfig.h.

The documentation for this struct was generated from the following file:

- [P2sbConfig.h](#)

13.125 PCH_PCIE_CLOCK Struct Reference

[PCH_PCIE_CLOCK](#) describes PCIe source clock generated by PCH.

```
#include <PchPcieRpConfig.h>
```

Public Attributes

- UINT8 [Usage](#)
Purpose of given clock (see PCH_PCIE_CLOCK_USAGE). Default: Unused, 0xFF.
- UINT8 [ClkReq](#)
ClkSrc - ClkReq mapping. Default: 1:1 mapping with Clock numbers.
- UINT8 [RsvdBytes](#) [2]
Reserved byte.

13.125.1 Detailed Description

[PCH_PCIE_CLOCK](#) describes PCIe source clock generated by PCH.

Definition at line 177 of file PchPcieRpConfig.h.

The documentation for this struct was generated from the following file:

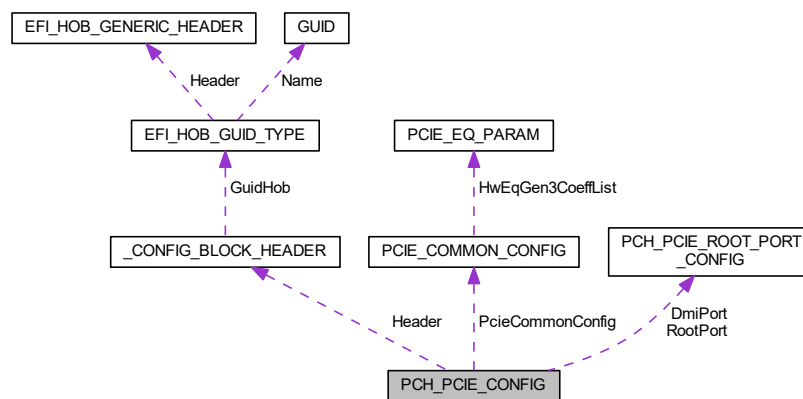
- [PchPcieRpConfig.h](#)

13.126 PCH_PCIE_CONFIG Struct Reference

The [PCH_PCIE_CONFIG](#) block describes the expected configuration of the PCH PCI Express controllers **Revision 1**:

```
#include <PchPcieRpConfig.h>
```

Collaboration diagram for PCH_PCIE_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [PCIE_COMMON_CONFIG](#) PcieCommonConfig
These members describe the configuration of each PCH PCIe root port.
- UINT8 [PchPciePort8xhDecodePortIndex](#)
(Test) The Index of PCIe Port that is selected for Port8xh Decode (0 Based)

13.126.1 Detailed Description

The [PCH_PCIE_CONFIG](#) block describes the expected configuration of the PCH PCI Express controllers **Revision 1**:

- Initial version. **Revision 2**:
- Moved EnablePort8xhDecode policy to [PCIE_COMMON_CONFIG](#)

Definition at line 221 of file PchPcieRpConfig.h.

The documentation for this struct was generated from the following file:

- [PchPcieRpConfig.h](#)

13.127 PCH_PCIE_ROOT_PORT_CONFIG Struct Reference

The PCH_PCI_EXPRESS_ROOT_PORT_CONFIG describe the feature and capability of each PCH PCIe root port.

```
#include <PchPcieRpConfig.h>
```

Public Attributes

- UINT8 [ExtSync](#)
an instance of Pcie Common Config
- UINT8 [SystemErrorEnable](#)
Indicate whether the System Error is enabled. 0: Disable; 1: Enable.
- UINT8 [MvcEnabled](#)
The Multiple VC (MVC) supports hardware to avoid HoQ block for latency sensitive TC.
- UINT8 [VppPort](#)
Virtual Pin Port is industry standard introduced to PCIe Hot Plug support in systems when GPIO pins expansion is needed.
- UINT8 [VppAddress](#)
PCIe Hot Plug VPP SMBus Address. Default is zero.
- UINT8 [RsvdBytes0](#) [3]
Reserved bytes.

13.127.1 Detailed Description

The PCH_PCI_EXPRESS_ROOT_PORT_CONFIG describe the feature and capability of each PCH PCIe root port.

Definition at line 188 of file PchPcieRpConfig.h.

13.127.2 Member Data Documentation

13.127.2.1 ExtSync

```
UINT8 PCH_PCIE_ROOT_PORT_CONFIG::ExtSync
```

an instance of Pcie Common Config

Indicate whether the extended synch is enabled. **0: Disable**; 1: Enable.

Definition at line 190 of file PchPcieRpConfig.h.

13.127.2.2 MvcEnabled

```
UINT8 PCH_PCIE_ROOT_PORT_CONFIG::MvcEnabled
```

The Multiple VC (MVC) supports hardware to avoid HoQ block for latency sensitive TC.

Currently it is only applicable to Root Ports with 2pX4 port configuration with 2 VCs, or DMI port configuration with 3 VCs. For Root Ports 2pX4 configuration, two RPs (RP0, RP2) shall support two PCIe VCs (VC0 & VC1) and the other RPs (RP1, RP3) shall be disabled. **0: Disable**; 1: Enable

Definition at line 203 of file PchPcieRpConfig.h.

13.127.2.3 VppPort

```
UINT8 PCH_PCIE_ROOT_PORT_CONFIG::VppPort
```

Virtual Pin Port is industry standard introduced to PCIe Hot Plug support in systems when GPIO pins expansion is needed.

It is server specific feature. **0x00: Default**; 0xFF: Disabled

Definition at line 209 of file PchPcieRpConfig.h.

The documentation for this struct was generated from the following file:

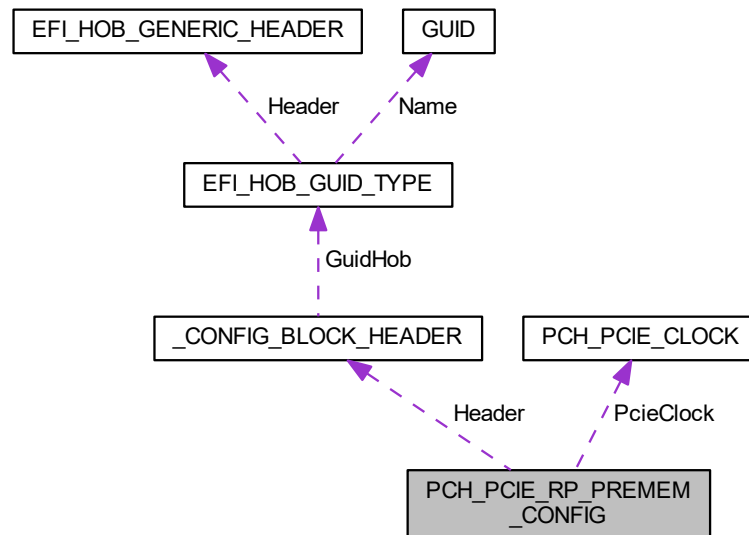
- [PchPcieRpConfig.h](#)

13.128 PCH_PCIE_RP_PREMEM_CONFIG Struct Reference

The [PCH_PCIE_RP_PREMEM_CONFIG](#) block describes early configuration of the PCH PCI Express controllers
Revision 1:

```
#include <PchPcieRpConfig.h>
```

Collaboration diagram for PCH_PCIE_RP_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- UINT32 [RpEnabledMask](#)
Root Port enabling mask.
- [PCH_PCIE_CLOCK PcieClock](#) [PCH_MAX_PCIE_CLOCKS]
Configuration of PCIe source clocks.
- UINT8 [Bifurcation](#) [PCH_MAX_PCIE_CONTROLLERS]
*Per Controller Bifurcation Configuration 0: **Disabled**; 1: 4x1; 2: 1x2_2x1; 3: 2x2; 4: 1x4; 5: 4x2; 6: 1x4_2x2; 7: 2x2_1x4; 8: 2x4; 9: 1x8 (see: PCIE_BIFURCATION_CONFIG)*

13.128.1 Detailed Description

The [PCH_PCIE_RP_PREMEM_CONFIG](#) block describes early configuration of the PCH PCI Express controllers
Revision 1:

- Initial version.

Definition at line 243 of file `PchPcieRpConfig.h`.

13.128.2 Member Data Documentation

13.128.2.1 RpEnabledMask

UINT32 PCH_PCIE_RP_PREMEM_CONFIG::RpEnabledMask

Root Port enabling mask.

Bit0 presents RP1, Bit1 presents RP2, and so on. 0: Disable; 1: **Enable**.

Definition at line 250 of file PchPcieRpConfig.h.

The documentation for this struct was generated from the following file:

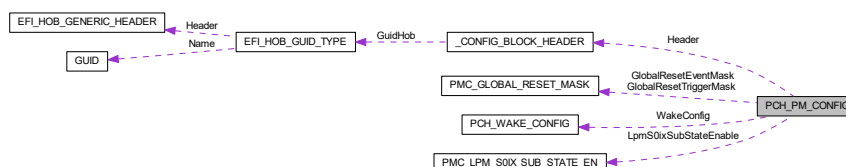
- [PchPcieRpConfig.h](#)

13.129 PCH_PM_CONFIG Struct Reference

The [PCH_PM_CONFIG](#) block describes expected miscellaneous power management settings.

```
#include <PmConfig.h>
```

Collaboration diagram for PCH_PM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [PCH_WAKE_CONFIG](#) WakeConfig
Specify Wake Policy.
- UINT32 [PchDeepSxPol](#): 4
*Deep Sx Policy. Refer to PCH_DEEP_SX_CONFIG for each value. Default is **PchDeepSxPolDisable**.*
- UINT32 [PchSlpS3MinAssert](#): 4
*SLP_S3 Minimum Assertion Width Policy. Refer to PCH_SLP_S3_MIN_ASSERT for each value. Default is **PchSlpS350ms**.*
- UINT32 [PchSlpS4MinAssert](#): 4
*SLP_S4 Minimum Assertion Width Policy. Refer to PCH_SLP_S4_MIN_ASSERT for each value. Default is **PchSlpS44s**.*
- UINT32 [PchSlpSusMinAssert](#): 4

*SLP_SUS Minimum Assertion Width Policy. Refer to PCH_SLP_SUS_MIN_ASSERT for each value. Default is **PchSlpSus4s**.*

- UUINT32 [PchSlpAMinAssert](#): 4
*SLP_A Minimum Assertion Width Policy. Refer to PCH_SLP_A_MIN_ASSERT for each value. Default is **PchSlpA2s**.*
- UUINT32 [SlpStrchSusUp](#): 1
This member describes whether or not the LPC ClockRun feature of PCH should be enabled.
- UUINT32 [SlpLanLowDc](#): 1
Enable/Disable SLP_LAN# Low on DC Power.
- UUINT32 [PwrBtnOverridePeriod](#): 3
PCH power button override period.
- UUINT32 [DisableEnergyReport](#): 1
(Test) Disable/Enable PCH to CPU enery report feature.
- UUINT32 [DisableDsxAcPresentPulldown](#): 1
When set to Disable, PCH will internal pull down AC_PRESENT in deep SX and during G3 exit.
- UUINT32 [DisableNativePowerButton](#): 1
Power button native mode disable.
- UUINT32 [MeWakeSts](#): 1
*Clear the ME_WAKE_STS bit in the Power and Reset Status (PRSTS) register. 0: Disable; 1: **Enable**.*
- UUINT32 [WolOvrWkSts](#): 1
*Clear the WOL_OVR_WK_STS bit in the Power and Reset Status (PRSTS) register. 0: Disable; 1: **Enable**.*
- UUINT32 [PsOnEnable](#): 1
Decide if PS_ON is to be enabled.
- UUINT32 [CpuC10GatePinEnable](#): 1
Enable/Disable platform support for CPU_C10_GATE# pin to control gating of CPU VccIO and VccSTG rails instead of SLP_S0# pin.
- UUINT32 [PmcDbgMsgEn](#): 1
Control whether to enable PMC debug messages to Trace Hub.
- UUINT32 [ModPhySusPgEnable](#): 1
Enable/Disable ModPHY SUS Power Domain Dynamic Gating.
- UUINT32 [Usb2PhySusPgEnable](#): 1
(Test) This policy option enables USB2 PHY SUS Well Power Gating functionality.
- UUINT32 [OsIdleEnable](#): 1
Enable Os Idle Mode.
- UUINT32 [V1p05PhyExtFetControlEn](#): 1
*Enable control using EXT_PWR_GATE# pin of external FET to power gate v1p05-PHY 0: **Disable**; 1: Enable.*
- UUINT32 [V1p05IsExtFetControlEn](#): 1
*Enable control using EXT_PWR_GATE2# pin of external FET to power gate v1p05-IS supply 0: **Disable**; 1: Enable.*
- UUINT32 [S0ixAutoDemotion](#): 1
Enable/Disable the Low Power Mode Host S0ix Auto-Demotion feature.
- UUINT32 [LatchEventsC10Exit](#): 1
Enable/Disable Latch Events C10 Exit.
- UUINT8 [PchPwrCycDur](#)
Reset Power Cycle Duration could be customized in the unit of second.
- UUINT8 [PciePIISsc](#)
Specifies the Pcie PII Spread Spectrum Percentage The value of this policy is in 1/10th percent units.
- UUINT8 [C10DynamicThresholdAdjustment](#)
Tells BIOS to enable C10 dynamic threshold adjustment mode.
- UUINT8 [Rsvd0](#) [1]
Reserved bytes.
- [PMC_LPM_S0IX_SUB_STATE_EN LpmS0ixSubStateEnable](#)
(Test) Low Power Mode Enable/Disable config.

- UINT8 [GlobalResetMasksOverride](#)
Set true to enable override of Global Reset Event/Trigger masks.
- UINT8 [Rsvd1](#) [3]
Reserved bytes.
- UINT32 [ThermTimerDelay](#): 10
PMC Therm Timer Delay Register (THERM_TIMER_DELAY).

13.129.1 Detailed Description

The [PCH_PM_CONFIG](#) block describes expected miscellaneous power management settings.

The PowerResetStatusClear field would clear the Power/Reset status bits, please set the bits if you want PCH Init driver to clear it, if you want to check the status later then clear the bits.

Revision 1:

- Initial version. **Revision 2**
- Added C10DynamicThresholdAdjustment **Revision 3:**
- Added SkipVccInConfig

Definition at line 226 of file PmConfig.h.

13.129.2 Member Data Documentation

13.129.2.1 C10DynamicThresholdAdjustment

```
UINT8 PCH_PM_CONFIG::C10DynamicThresholdAdjustment
```

Tells BIOS to enable C10 dynamic threshold adjustment mode.

BIOS will only attempt to enable it on PCH SKUs which support it.

Definition at line 408 of file PmConfig.h.

13.129.2.2 CpuC10GatePinEnable

```
UINT32 PCH_PM_CONFIG::CpuC10GatePinEnable
```

Enable/Disable platform support for CPU_C10_GATE# pin to control gating of CPU VccIO and VccSTG rails instead of SLP_S0# pin.

This policy needs to be set if board design includes support for CPU_C10_GATE# pin. 0: Disable; 1: **Enable**

Definition at line 312 of file PmConfig.h.

13.129.2.3 DisableDsxAcPresentPulldown

```
UINT32 PCH_PM_CONFIG::DisableDsxAcPresentPulldown
```

When set to Disable, PCH will internal pull down AC_PRESENT in deep SX and during G3 exit.

When set to Enable, PCH will not pull down AC_PRESENT. This setting is ignored when DeepSx is not supported.
Default is **0:Disable**

Definition at line 273 of file PmConfig.h.

13.129.2.4 DisableEnergyReport

```
UINT32 PCH_PM_CONFIG::DisableEnergyReport
```

(Test) Disable/Enable PCH to CPU enery report feature.

0: Disable; 1: Enable. Enery Report is must have feature. Wihtout Energy Report, the performance report by workloads/benchmarks will be unrealistic because PCH's energy is not being accounted in power/performance management algorithm. If for some reason PCH energy report is too high, which forces CPU to try to reduce its power by throttling, then it could try to disable Energy Report to do first debug. This might be due to energy scaling factors are not correct or the LPM settings are not kicking in.

Definition at line 266 of file PmConfig.h.

13.129.2.5 DisableNativePowerButton

```
UINT32 PCH_PM_CONFIG::DisableNativePowerButton
```

Power button native mode disable.

While FALSE, the PMC's power button logic will act upon the input value from the GPIO unit, as normal. While TRUE, this will result in the PMC logic constantly seeing the power button as de-asserted. **Default is FALSE.**

Definition at line 280 of file PmConfig.h.

13.129.2.6 GlobalResetMasksOverride

```
UINT8 PCH_PM_CONFIG::GlobalResetMasksOverride
```

Set true to enable override of Global Reset Event/Trigger masks.

Values from GlobalResetTriggerMask and GlobalResetEventMask will be used as override value. **0: Disable**, 1: Enable

Definition at line 442 of file PmConfig.h.

13.129.2.7 LatchEventsC10Exit

UINT32 PCH_PM_CONFIG::LatchEventsC10Exit

Enable/Disable Latch Events C10 Exit.

When this bit is set to 1, SLP_S0# entry events in SLP_S0_DEBUG_REGx registers are captured on C10 exit (instead of C10 entry which is default) **0: Disable**; 1: Enable.

Definition at line 368 of file PmConfig.h.

13.129.2.8 LpmS0ixSubStateEnable

PMC_LPM_S0IX_SUB_STATE_EN PCH_PM_CONFIG::LpmS0ixSubStateEnable

(Test) Low Power Mode Enable/Disable config.

Configure if respective S0i2/3 sub-states are to be supported by the platform. By default all sub-states are enabled but for test purpose respective states can be disabled. **Default is 0xFF**

Definition at line 418 of file PmConfig.h.

13.129.2.9 ModPhySusPgEnable

UINT32 PCH_PM_CONFIG::ModPhySusPgEnable

Enable/Disable ModPHY SUS Power Domain Dynamic Gating.

EXT_PWR_GATE# signal (if supported on platform) can be used to control external FET for power gating ModPHY

Note

: This setting is not supported and ignored on PCH-H 0: Disable; **1: Enable**.

Definition at line 328 of file PmConfig.h.

13.129.2.10 OsIdleEnable

UINT32 PCH_PM_CONFIG::OsIdleEnable

Enable Os Idle Mode.

0: Disable; **1: Enable**.

Definition at line 341 of file PmConfig.h.

13.129.2.11 PchPwrCycDur

UINT8 PCH_PM_CONFIG::PchPwrCycDur

Reset Power Cycle Duration could be customized in the unit of second.

Please refer to EDS for all support settings. PCH HW default is 4 seconds, and range is 1~4 seconds, where **0 is default**, 1 is 1 second, 2 is 2 seconds, ... 4 is 4 seconds. And make sure the setting correct, which never less than the following register.

- GEN_PMCON_B.SLP_S3_MIN_ASST_WDTH
- GEN_PMCON_B.SLP_S4_MIN_ASST_WDTH
- PWRM_CFG.SLP_A_MIN_ASST_WDTH
- PWRM_CFG.SLP_LAN_MIN_ASST_WDTH

Definition at line 395 of file PmConfig.h.

13.129.2.12 PciePllSsc

UINT8 PCH_PM_CONFIG::PciePllSsc

Specifies the Pcie Pll Spread Spectrum Percentage The value of this policy is in 1/10th percent units.

Valid spread range is 0-20. A value of 0xFF is reserved for AUTO. A value of 0 is SSC of 0.0%. A value of 20 is SSC of 2.0% The default is **0xFF: AUTO - No BIOS override**.

Definition at line 403 of file PmConfig.h.

13.129.2.13 PmcDbgMsgEn

UINT32 PCH_PM_CONFIG::PmcDbgMsgEn

Control whether to enable PMC debug messages to Trace Hub.

When Enabled, PMC HW will send debug messages to trace hub; When Disabled, PMC HW will never send debug messages to trace hub.

Note

: When enabled, system may not enter S0ix **0: Disable**; 1: Enable.

Definition at line 320 of file PmConfig.h.

13.129.2.14 PsOnEnable

```
UINT32 PCH_PM_CONFIG::PsOnEnable
```

Decide if PS_ON is to be enabled.

This is available on desktop only. PS_ON is a new C10 state from the CPU on desktop SKUs that enables a lower power target that will be required by the California Energy Commission (CEC). When FALSE, PS_ON is to be disabled.} **0: Disable**; 1: Enable.

Definition at line 305 of file PmConfig.h.

13.129.2.15 PwrBtnOverridePeriod

```
UINT32 PCH_PM_CONFIG::PwrBtnOverridePeriod
```

PCH power button override period.

000b-4s, 001b-6s, 010b-8s, 011b-10s, 100b-12s, 101b-14s **Default is 0: 4s**

Definition at line 254 of file PmConfig.h.

13.129.2.16 S0ixAutoDemotion

```
UINT32 PCH_PM_CONFIG::S0ixAutoDemotion
```

Enable/Disable the Low Power Mode Host S0ix Auto-Demotion feature.

This feature enables the PMC to autonomously manage the deepest allowed S0ix substate to combat thrashing between power management states. 0: Disable; **1: Enable**.

Definition at line 361 of file PmConfig.h.

13.129.2.17 SlpLanLowDc

```
UINT32 PCH_PM_CONFIG::SlpLanLowDc
```

Enable/Disable SLP_LAN# Low on DC Power.

0: Disable; **1: Enable**. Configure On DC PHY Power Diabale according to policy SlpLanLowDc. When this is enabled, SLP_LAN# will be driven low when ACPRESENT is low. This indicates that LAN PHY should be powered off on battery mode. This will override the DC_PP_DIS setting by WolEnableOverride.

Definition at line 248 of file PmConfig.h.

13.129.2.18 SlpStrchSusUp

```
UINT32 PCH_PM_CONFIG::SlpStrchSusUp
```

This member describes whether or not the LPC ClockRun feature of PCH should be enabled.

0: Disable; 1: Enable **0: Disable**; 1: Enable SLP_X Stretching After SUS Well Power Up

Definition at line 240 of file PmConfig.h.

13.129.2.19 ThermTimerDelay

```
UINT32 PCH_PM_CONFIG::ThermTimerDelay
```

PMC Therm Timer Delay Register (THERM_TIMER_DELAY).

Unit in us(microseconds). Adding delay time between CPU thermal trip propagating through PCH and PCH generating a Global Reset **Default is 500**

Definition at line 457 of file PmConfig.h.

13.129.2.20 Usb2PhySusPgEnable

```
UINT32 PCH_PM_CONFIG::Usb2PhySusPgEnable
```

(Test) This policy option enables USB2 PHY SUS Well Power Gating functionality.

Note

: This setting is not supported and ignored on PCH-H 0: disable USB2 PHY SUS Well Power Gating **1: enable USB2 PHY SUS Well Power Gating**

Definition at line 336 of file PmConfig.h.

The documentation for this struct was generated from the following file:

- [PmConfig.h](#)

13.130 PCH_SATA_PORT_CONFIG Struct Reference

This structure configures the features, property, and capability for each SATA port.

```
#include <SataConfig.h>
```

Public Attributes

- UINT32 [Enable](#): 1
Enable SATA port.
- UINT32 [HotPlug](#): 1
0: Disable; 1: Enable
- UINT32 [InterlockSw](#): 1
0: Disable; 1: Enable
- UINT32 [External](#): 1
0: Disable; 1: Enable
- UINT32 [SpinUp](#): 1
0: Disable; 1: Enable the COMRESET initialization Sequence to the device
- UINT32 [SolidStateDrive](#): 1
0: HDD; 1: SSD
- UINT32 [DevSlp](#): 1
0: Disable; 1: Enable DEVSLP on the port
- UINT32 [EnableDitoConfig](#): 1
0: Disable; 1: Enable DEVSLP Idle Timeout settings (DmVal, DitoVal)
- UINT32 [DmVal](#): 4
DITO multiplier. Default is 15.
- UINT32 [DitoVal](#): 10
DEVSLP Idle Timeout (DITO), Default is 625.
- UINT32 [ZpOdd](#): 1
Support zero power ODD 0: Disable, 1: Enable.
- UINT32 [DevSlpResetConfig](#): 4
0: Hardware default; 0x01: GpioResumeReset; 0x03: GpioHostDeepReset; 0x05: GpioPlatformReset; 0x07↔ : GpioDswReset
- UINT32 [SataPmPtm](#): 1
Deprecated.
- UINT32 [RxPolarity](#): 1
0: Disable; 1: Enable; Rx Polarity
- UINT32 [RsvdBits0](#): 3
Reserved fields for future expansion w/o protocol change.

13.130.1 Detailed Description

This structure configures the features, property, and capability for each SATA port.

Definition at line 75 of file SataConfig.h.

13.130.2 Member Data Documentation

13.130.2.1 Enable

```
UINT32 PCH_SATA_PORT_CONFIG::Enable
```

Enable SATA port.

It is highly recommended to disable unused ports for power savings 0: Disable; **1: Enable**

Definition at line 80 of file SataConfig.h.

13.130.2.2 ZpOdd

```
UINT32 PCH_SATA_PORT_CONFIG::ZpOdd
```

Support zero power ODD **0: Disable**, 1: Enable.

This is also used to disable ModPHY dynamic power gate.

Definition at line 94 of file SataConfig.h.

The documentation for this struct was generated from the following file:

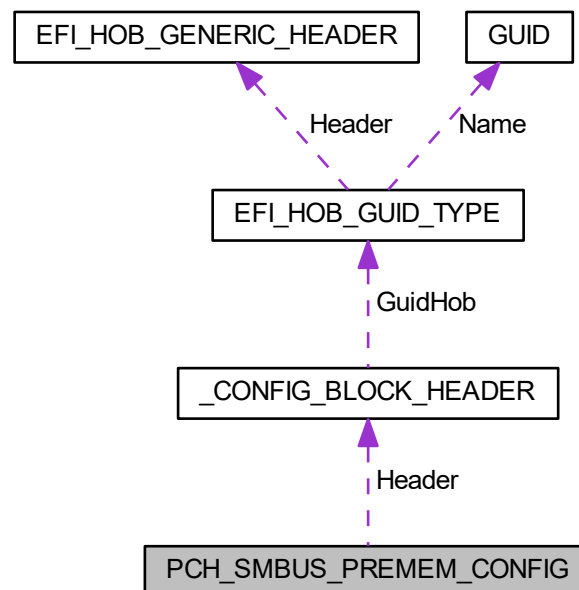
- [SataConfig.h](#)

13.131 PCH_SMBUS_PREMEM_CONFIG Struct Reference

The SMBUS_CONFIG block lists the reserved addresses for non-ARP capable devices in the platform.

```
#include <SmbusConfig.h>
```

Collaboration diagram for PCH_SMBUS_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Revision 1: Init version.
- UINT32 [Enable](#): 1
This member describes whether or not the SMBus controller of PCH should be enabled.
- UINT32 [ArpEnable](#): 1
*Enable SMBus ARP support, **0: Disable**; 1: Enable.*
- UINT32 [DynamicPowerGating](#): 1
*(Test) **Disable** or Enable Smbus dynamic power gating.*
- UINT32 [SpdWriteDisable](#): 1
*(Test) SPD Write Disable, 0: leave SPD Write Disable bit; **1: set SPD Write Disable bit.***
- UINT32 [SmbAlertEnable](#): 1
*Enable SMBus Alert pin (SMBALERT#). 0: **Disabled**, 1: **Enabled**.*
- UINT32 [RsvdBits0](#): 27
Reserved bits.
- UINT16 [SmbusIoBase](#)
*SMBUS Base Address (IO space). Default is **0xEFA0**.*
- UINT8 [Rsvd0](#)
Reserved bytes.
- UINT8 [NumRsvdSmbusAddresses](#)
The number of elements in the RsvdSmbusAddressTable.
- UINT8 [RsvdSmbusAddressTable](#) [PCH_MAX_SMBUS_RESERVED_ADDRESS]
Array of addresses reserved for non-ARP-capable SMBus devices.

13.131.1 Detailed Description

The SMBUS_CONFIG block lists the reserved addresses for non-ARP capable devices in the platform.

Definition at line 48 of file SmbusConfig.h.

13.131.2 Member Data Documentation

13.131.2.1 Enable

```
UINT32 PCH_SMBUS_PREMEM_CONFIG::Enable
```

This member describes whether or not the SMBus controller of PCH should be enabled.

0: Disable; 1: **Enable**.

Definition at line 57 of file SmbusConfig.h.

13.131.2.2 Header

`CONFIG_BLOCK_HEADER PCH_SMBUS_PREMEM_CONFIG::Header`

Revision 1: Init version.

Config Block Header

Definition at line 52 of file SmbusConfig.h.

13.131.2.3 SpdWriteDisable

`UINT32 PCH_SMBUS_PREMEM_CONFIG::SpdWriteDisable`

(Test) SPD Write Disable, 0: leave SPD Write Disable bit; **1: set SPD Write Disable bit.**

For security recommendations, SPD write disable bit must be set.

Definition at line 64 of file SmbusConfig.h.

The documentation for this struct was generated from the following file:

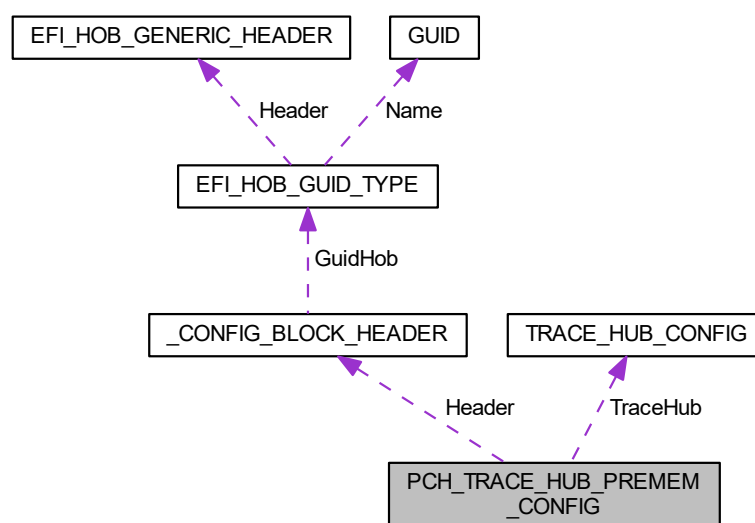
- [SmbusConfig.h](#)

13.132 PCH_TRACE_HUB_PREMEM_CONFIG Struct Reference

PCH Trace Hub PreMem Configuration Contains Trace Hub settings for PCH side tracing **Revision 1:** - Initial version.

```
#include <TraceHubConfig.h>
```

Collaboration diagram for PCH_TRACE_HUB_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [TRACE_HUB_CONFIG](#) TraceHub
Trace Hub Config.

13.132.1 Detailed Description

PCH Trace Hub PreMem Configuration Contains Trace Hub settings for PCH side tracing **Revision 1:** - Initial version.

Definition at line 122 of file TraceHubConfig.h.

The documentation for this struct was generated from the following file:

- [TraceHubConfig.h](#)

13.133 PCH_WAKE_CONFIG Struct Reference

This structure allows to customize PCH wake up capability from S5 or DeepSx by WOL, LAN, PCIE wake events.

```
#include <PmConfig.h>
```

Public Attributes

- UINT32 [PmeB0S5Dis](#): 1
Corresponds to the PME_B0_S5_DIS bit in the General PM Configuration B (GEN_PMCON_B) register.
- UINT32 [WoLEnableOverride](#): 1
Corresponds to the "WOL Enable Override" bit in the General PM Configuration B (GEN_PMCON_B) register. 0: Disable; 1: Enable.
- UINT32 [PcieWakeFromDeepSx](#): 1
Determine if enable PCIe to wake from deep Sx. 0: Disable; 1: Enable.
- UINT32 [WoWlanEnable](#): 1
Determine if WLAN wake from Sx, corresponds to the "HOST_WLAN_PP_EN" bit in the PWRM_CFG3 register. 0: Disable; 1: Enable.
- UINT32 [WoWlanDeepSxEnable](#): 1
Determine if WLAN wake from DeepSx, corresponds to the "DSX_WLAN_PP_EN" bit in the PWRM_CFG3 register. 0: Disable; 1: Enable.
- UINT32 [LanWakeFromDeepSx](#): 1
Determine if enable LAN to wake from deep Sx. 0: Disable; 1: Enable.

13.133.1 Detailed Description

This structure allows to customize PCH wake up capability from S5 or DeepSx by WOL, LAN, PCIE wake events.

Definition at line 48 of file PmConfig.h.

13.133.2 Member Data Documentation

13.133.2.1 PmeB0S5Dis

UINT32 PCH_WAKE_CONFIG::PmeB0S5Dis

Corresponds to the PME_B0_S5_DIS bit in the General PM Configuration B (GEN_PMCON_B) register.

When set to 1, this bit blocks wake events from PME_B0_STS in S5, regardless of the state of PME_B0_EN. When cleared (default), wake events from PME_B0_STS are allowed in S5 if PME_B0_EN = 1. **0: Disable**; 1: Enable.

Definition at line 54 of file PmConfig.h.

The documentation for this struct was generated from the following file:

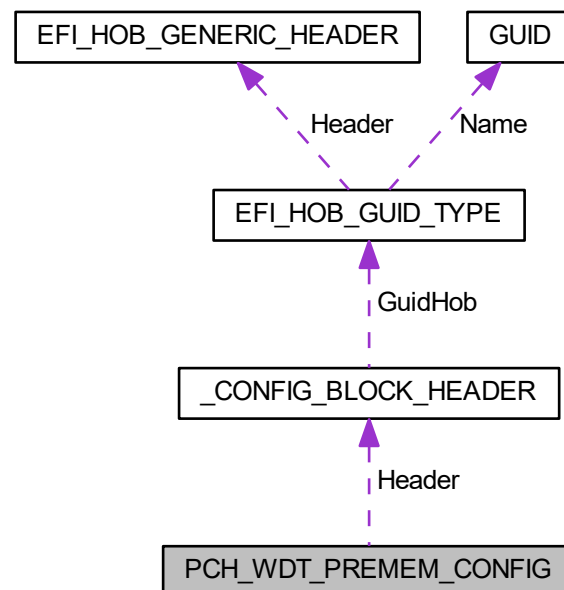
- [PmConfig.h](#)

13.134 PCH_WDT_PREMEM_CONFIG Struct Reference

This policy clears status bits and disable watchdog, then lock the WDT registers.

```
#include <WatchDogConfig.h>
```

Collaboration diagram for PCH_WDT_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [DisableAndLock](#): 1
(Test) Set 1 to clear WDT status, then disable and lock WDT registers. 0: Disable; 1: Enable.

13.134.1 Detailed Description

This policy clears status bits and disable watchdog, then lock the WDT registers.

while WDT is designed to be disabled and locked by Policy, bios should not enable WDT by WDT PPI. In such case, bios shows the warning message but not disable and lock WDT register to make sure WDT event trigger correctly.

Definition at line 51 of file WatchDogConfig.h.

The documentation for this struct was generated from the following file:

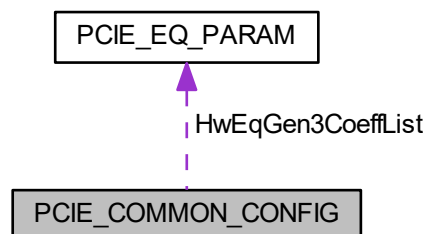
- [WatchDogConfig.h](#)

13.135 PCIE_COMMON_CONFIG Struct Reference

PCIe Common Config.

```
#include <PcieConfig.h>
```

Collaboration diagram for PCIE_COMMON_CONFIG:



Public Attributes

- UINT32 [RpFunctionSwap](#): 1
RpFunctionSwap allows BIOS to use root port function number swapping when root port of function 0 is disabled.
- UINT32 [ComplianceTestMode](#): 1
Compliance Test Mode shall be enabled when using Compliance Load Board.
- UINT32 [RsvdBits0](#): 30
Reserved bits.
- UINT8 [EnablePort8xhDecode](#)
(Test) This member describes whether PCIe root port Port 8xh Decode is enabled.
- [PCIE_EQ_PARAM](#) [HwEqGen3CoeffList](#) [PCIE_HWEQ_COEFFS_MAX]
List of coefficients used during equalization (applicable to both software and hardware EQ) Deprecated Policy.

13.135.1 Detailed Description

PCIe Common Config.

Note

This structure will be expanded to hold all common PCIe policies between SA and PCH

Definition at line 287 of file PcieConfig.h.

13.135.2 Member Data Documentation

13.135.2.1 ComplianceTestMode

UINT32 PCIE_COMMON_CONFIG::ComplianceTestMode

Compliance Test Mode shall be enabled when using Compliance Load Board.

0: Disable, 1: Enable

Definition at line 308 of file PcieConfig.h.

13.135.2.2 EnablePort8xhDecode

UINT8 PCIE_COMMON_CONFIG::EnablePort8xhDecode

(Test) This member describes whether PCIe root port Port 8xh Decode is enabled.

0: Disable; 1: Enable.

Definition at line 313 of file PcieConfig.h.

13.135.2.3 RpFunctionSwap

UINT32 PCIE_COMMON_CONFIG::RpFunctionSwap

RpFunctionSwap allows BIOS to use root port function number swapping when root port of function 0 is disabled.

A PCIe device can have higher functions only when Function0 exists. To satisfy this requirement, BIOS will always enable Function0 of a device that contains more than 0 enabled root ports.

- **Enabled: One of enabled root ports get assigned to Function0.** This offers no guarantee that any particular root port will be available at a specific DevNr:FuncNr location
- **Disabled:** Root port that corresponds to Function0 will be kept visible even though it might be not used. That way rootport - to - DevNr:FuncNr assignment is constant. This option will impact ports 1, 9, 17. NOTE: This option will not work if ports 1, 9, 17 are fused or configured for RST PCIe storage or disabled through policy In other words, it only affects ports that would become hidden because they have no device connected. NOTE: Disabling function swap may have adverse impact on power management. This option should ONLY be used when each one of root ports 1, 9, 17:
 - is configured as PCIe and has correctly configured ClkReq signal, or
 - does not own any mPhy lanes (they are configured as SATA or USB)

Definition at line 303 of file PcieConfig.h.

The documentation for this struct was generated from the following file:

- [PcieConfig.h](#)

13.136 PCIE_DEVICE_OVERRIDE Struct Reference

PCIe device table entry entry.

```
#include <PcieConfig.h>
```

Public Attributes

- [UINT16 VendorId](#)
The vendor Id of Pci Express card ASPM setting override, 0xFFFF means any Vendor ID.
- [UINT16 DeviceId](#)
The Device Id of Pci Express card ASPM setting override, 0xFFFF means any Device ID.
- [UINT8 RevId](#)
The Rev Id of Pci Express card ASPM setting override, 0xFF means all steppings.
- [UINT8 BaseClassCode](#)
The Base Class Code of Pci Express card ASPM setting override, 0xFF means all base class.
- [UINT8 SubClassCode](#)
The Sub Class Code of Pci Express card ASPM setting override, 0xFF means all sub class.
- [UINT8 EndPointAspm](#)
Override device ASPM (see: PCH_PCIE_ASPM_CONTROL) Bit 1 must be set in OverrideConfig for this field to take effect.
- [UINT16 OverrideConfig](#)
The override config bitmap (see: PCH_PCIE_OVERRIDE_CONFIG).
- [UINT16 L1SubstatesCapOffset](#)
The L1Substates Capability Offset Override.
- [UINT8 L1SubstatesCapMask](#)
L1 Substate Capability Mask.
- [UINT8 L1sCommonModeRestoreTime](#)
L1 Substate Port Common Mode Restore Time Override.
- [UINT8 L1sTpowerOnScale](#)
L1 Substate Port Tpower_on Scale Override.
- [UINT8 L1sTpowerOnValue](#)
L1 Substate Port Tpower_on Value Override.
- [UINT16 SnoopLatency](#)
SnoopLatency bit definition Note: All Reserved bits must be set to 0.
- [UINT16 NonSnoopLatency](#)
NonSnoopLatency bit definition Note: All Reserved bits must be set to 0.
- [UINT8 ForceLtrOverride](#)
Forces LTR override to be permanent The default way LTR override works is: rootport uses LTR override values provided by BIOS until connected device sends an LTR message, then it will use values from the message This settings allows force override of LTR mechanism.

13.136.1 Detailed Description

PCIe device table entry entry.

The PCIe device table is being used to override PCIe device ASPM settings. To take effect table consisting of such entries must be installed as PPI on gPchPcieDeviceTablePpiGuid. Last entry VendorId must be 0.

Definition at line 337 of file PcieConfig.h.

13.136.2 Member Data Documentation

13.136.2.1 ForceLtrOverride

```
UINT8 PCIE_DEVICE_OVERRIDE::ForceLtrOverride
```

Forces LTR override to be permanent The default way LTR override works is: rootport uses LTR override values provided by BIOS until connected device sends an LTR message, then it will use values from the message This settings allows force override of LTR mechanism.

If it's enabled, then: rootport will use LTR override values provided by BIOS forever; LTR messages sent from connected device will be ignored

Definition at line 431 of file PcieConfig.h.

13.136.2.2 L1sCommonModeRestoreTime

```
UINT8 PCIE_DEVICE_OVERRIDE::L1sCommonModeRestoreTime
```

L1 Substate Port Common Mode Restore Time Override.

(applicable if bit 2 is set in OverrideConfig) L1sCommonModeRestoreTime and L1sTpPowerOnScale can have a valid value of 0, but not the L1sTpPowerOnValue. If L1sTpPowerOnValue is zero, all L1sCommonModeRestoreTime, L1sTpPowerOnScale, and L1sTpPowerOnValue are ignored, and only L1SubstatesCapOffset is override.

Definition at line 363 of file PcieConfig.h.

13.136.2.3 L1sTpPowerOnScale

```
UINT8 PCIE_DEVICE_OVERRIDE::L1sTpPowerOnScale
```

L1 Substate Port Tpower_on Scale Override.

(applicable if bit 2 is set in OverrideConfig) L1sCommonModeRestoreTime and L1sTpPowerOnScale can have a valid value of 0, but not the L1sTpPowerOnValue. If L1sTpPowerOnValue is zero, all L1sCommonModeRestoreTime, L1sTpPowerOnScale, and L1sTpPowerOnValue are ignored, and only L1SubstatesCapOffset is override.

Definition at line 370 of file PcieConfig.h.

13.136.2.4 L1sTpowerOnValue

```
UINT8 PCIE_DEVICE_OVERRIDE::L1sTpowerOnValue
```

L1 Substate Port Tpower_on Value Override.

(applicable if bit 2 is set in OverrideConfig) L1sCommonModeRestoreTime and L1sTpowerOnScale can have a valid value of 0, but not the L1sTpowerOnValue. If L1sTpowerOnValue is zero, all L1sCommonModeRestoreTime, L1sTpowerOnScale, and L1sTpowerOnValue are ignored, and only L1SubstatesCapOffset is override.

Definition at line 377 of file PcieConfig.h.

13.136.2.5 L1SubstatesCapMask

```
UINT8 PCIE_DEVICE_OVERRIDE::L1SubstatesCapMask
```

L1 Substate Capability Mask.

(applicable if bit 2 is set in OverrideConfig) Set to zero then the L1 Substate Capability [3:0] is ignored, and only L1s values are override. Only bit [3:0] are applicable. Other bits are ignored.

Definition at line 356 of file PcieConfig.h.

13.136.2.6 L1SubstatesCapOffset

```
UINT16 PCIE_DEVICE_OVERRIDE::L1SubstatesCapOffset
```

The L1Substates Capability Offset Override.

(applicable if bit 2 is set in OverrideConfig) This field can be zero if only the L1 Substate value is going to be override.

Definition at line 350 of file PcieConfig.h.

13.136.2.7 NonSnoopLatency

```
UINT16 PCIE_DEVICE_OVERRIDE::NonSnoopLatency
```

NonSnoopLatency bit definition Note: All Reserved bits must be set to 0.

BIT[15] - When set to 1b, indicates that the values in bits 9:0 are valid When clear values in bits 9:0 will be ignored
BITS[14:13] - Reserved
BITS[12:10] - Value in bits 9:0 will be multiplied with the scale in these bits
000b - 1 ns
001b - 32 ns
010b - 1024 ns
011b - 32,768 ns
100b - 1,048,576 ns
101b - 33,554,432 ns
110b - Reserved
111b - Reserved
BITS[9:0] - Non Snoop Latency Value. The value in these bits will be multiplied with the scale in bits 12:10

This field takes effect only if bit 3 is set in OverrideConfig.

Definition at line 422 of file PcieConfig.h.

13.136.2.8 SnoopLatency

UINT16 PCIE_DEVICE_OVERRIDE::SnoopLatency

SnoopLatency bit definition Note: All Reserved bits must be set to 0.

BIT[15] - When set to 1b, indicates that the values in bits 9:0 are valid When clear values in bits 9:0 will be ignored
BITS[14:13] - Reserved BITS[12:10] - Value in bits 9:0 will be multiplied with the scale in these bits 000b - 1 ns
001b - 32 ns 010b - 1024 ns 011b - 32,768 ns 100b - 1,048,576 ns 101b - 33,554,432 ns 110b - Reserved 111b -
Reserved BITS[9:0] - Snoop Latency Value. The value in these bits will be multiplied with the scale in bits 12:10

This field takes effect only if bit 3 is set in OverrideConfig.

Definition at line 400 of file PcieConfig.h.

The documentation for this struct was generated from the following file:

- [PcieConfig.h](#)

13.137 PCIE_EQ_PARAM Struct Reference

Represent lane specific PCIe Gen3 equalization parameters.

```
#include <PcieConfig.h>
```

Public Attributes

- UINT8 [Cm](#)
Coefficient C-1.
- UINT8 [Cp](#)
Coefficient C+1.
- UINT8 [Rsvd0](#) [2]
Reserved bytes.

13.137.1 Detailed Description

Represent lane specific PCIe Gen3 equalization parameters.

Definition at line 76 of file PcieConfig.h.

The documentation for this struct was generated from the following file:

- [PcieConfig.h](#)

13.138 PCIE_IMR_CONFIG Struct Reference

PCIe IMR Config.

```
#include <PciePreMemConfig.h>
```

Public Attributes

- UINT8 [ImrEnabled](#)
*PCIe IMR. 0: **Disable**; 1: Enable.*
- UINT8 [ImrRpLocation](#)
0: PCH_PCIe; 1: CPU_PCIe. If PCIeImrEnabled is TRUE then this will use to select the Root port location from PCH PCIe or CPU PCIe. Refer PCIe_IMR_ROOT_PORT_LOCATION above
- UINT16 [ImrSize](#)
PCIe IMR size in megabytes.
- UINT8 [ImrRpSelection](#)
Index of root port that is selected for PCIe IMR (0 based)

13.138.1 Detailed Description

PCIe IMR Config.

Definition at line 53 of file PciePreMemConfig.h.

The documentation for this struct was generated from the following file:

- [PciePreMemConfig.h](#)

13.139 PCIE_LINK_EQ_PLATFORM_SETTINGS Struct Reference

PCIe Link EQ Platform Settings.

```
#include <PcieConfig.h>
```

Public Attributes

- UINT8 [PcieLinkEqMethod](#)
Tells BIOS which link EQ method should be used for this port. Please refer to PCIE_LINK_EQ_METHOD for details of supported methods. Default: PcieLinkHardwareEq.
- UINT8 [PcieLinkEqMode](#)
Tells BIOS which mode should be used for PCIe link EQ. Please refer to PCIE_LINK_EQ_MODE for details of supported modes. Default: depends on SoC.
- UINT8 [LocalTransmitterOverrideEnable](#)
Specifies if BIOS should perform local transmitter override during phase 2 of EQ process.
- UINT8 [Ph3NumberOfPresetsOrCoefficients](#)
Tells BIOS how many presets/coefficients should be used during link EQ.
- PCIE_LINK_EQ_COEFFICIENTS [Ph3CoefficientsList](#) [PCIE_LINK_EQ_COEFFICIENTS_MAX]
List of the PCIe coefficients to be used during equalization process. Only valid if PcieLinkEqMode is PcieLinkEq↔CoefficientMode.
- UINT32 [Ph3PresetList](#) [PCIE_LINK_EQ_PRESETS_MAX]
List of the PCIe preset values to be used during equalization process. Only valid if PcieLinkEqMode is PcieLinkEq↔PresetMode.
- UINT32 [Ph1DownstreamPortTransmitterPreset](#)
Specifies the value of the downstream port transmitter preset to be used during phase 1 of the equalization process. Will be applied to all lanes.

- UINT32 [Ph1UpstreamPortTransmitterPreset](#)
Specifies the value of the upstream port transmitter preset to be used during phase 1 of the equalization process. Will be applied to all lanes.
- UINT32 [Ph2LocalTransmitterOverridePreset](#)
Specifies the preset that should be used during local transmitter override during phase 2 of EQ process.
- UINT32 [PCETTimer](#)
PCET Timer value for single PCIe speed.
- UINT8 [RemotePresetCoeffoverride](#)
Remote Transmitter Preset Coefficient Override for single PCIe speed.
- UINT8 [EqPh3Bypass](#)
PCIe Equalization Phase 3 Enable Control.
- UINT8 [EqPh23Bypass](#)
PCIe Equalization Phase 2-3 Enable Control.
- UINT8 [TsLockTimer](#)
8.0GT/s Training Sequence Wait Latency For Presets / Coefficients Evaluation - Gen3 TS Lock Timer

13.139.1 Detailed Description

PCIe Link EQ Platform Settings.

Definition at line 132 of file PcieConfig.h.

13.139.2 Member Data Documentation

13.139.2.1 EqPh23Bypass

UINT8 PCIE_LINK_EQ_PLATFORM_SETTINGS::EqPh23Bypass

PCIe Equalization Phase 2-3 Enable Control.

- **Disabled** (0x0) : Disable Phase 2 - Phase 3 (Default)
- **Enabled** (0x1) : Enable Phase 2 - Phase 3

Definition at line 170 of file PcieConfig.h.

13.139.2.2 EqPh3Bypass

UINT8 PCIE_LINK_EQ_PLATFORM_SETTINGS::EqPh3Bypass

PCIe Equalization Phase 3 Enable Control.

- **Disabled** (0x0) : Disable phase 3 (Default)
 - **Enabled** (0x1) : Enable phase 3

Definition at line 164 of file PcieConfig.h.

13.139.2.3 LocalTransmitterOverrideEnable

```
UINT8 PCIE_LINK_EQ_PLATFORM_SETTINGS::LocalTransmitterOverrideEnable
```

Specifies if BIOS should perform local transmitter override during phase 2 of EQ process.

If enabled value in Ph2LocalTransmitterOverridePreset must be valid. **0: Disabled**; 1: Enabled

Definition at line 140 of file PcieConfig.h.

13.139.2.4 Ph2LocalTransmitterOverridePreset

```
UINT32 PCIE_LINK_EQ_PLATFORM_SETTINGS::Ph2LocalTransmitterOverridePreset
```

Specifies the preset that should be used during local transmitter override during phase 2 of EQ process.

Used only if LocalTransmitterOverrideEnable is TRUE. Will be applied to all PCIe lanes of the root port. Valid up to the PCIE_LINK_EQ_PRESET_MAX value. **Default: 0** < >

Definition at line 156 of file PcieConfig.h.

13.139.2.5 Ph3NumberOfPresetsOrCoefficients

```
UINT8 PCIE_LINK_EQ_PLATFORM_SETTINGS::Ph3NumberOfPresetsOrCoefficients
```

Tells BIOS how many presets/coefficients should be used during link EQ.

Entries in the Ph3CoefficientsList or Ph3PresetList(depending on chosen mode) need to be valid up to the number specified in this field.

Definition at line 145 of file PcieConfig.h.

The documentation for this struct was generated from the following file:

- [PcieConfig.h](#)

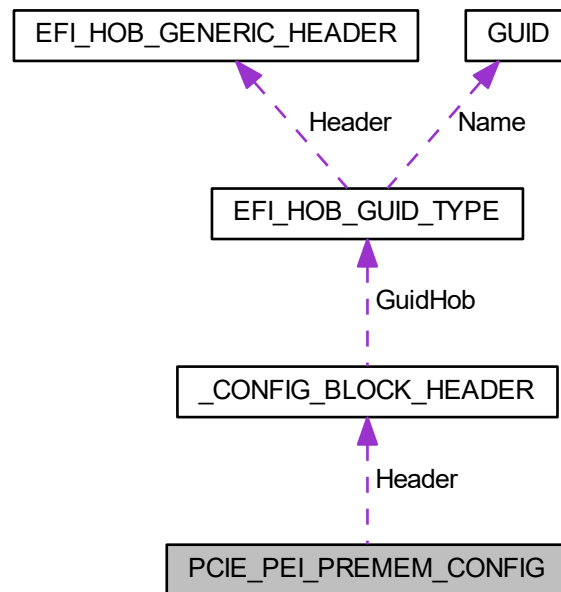
13.140 PCIE_PEI_PREMEM_CONFIG Struct Reference

PCI Express and DMI controller configuration

.

```
#include <CpuPcieConfig.h>
```

Collaboration diagram for PCIE_PEI_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.
- UINT32 [DmiMaxLinkSpeed](#): 2
Offset 28:0 : **(Test)** DMI Link Speed Control.
- UINT32 [DmiGen3EqPh2Enable](#): 2
Offset 28:2 : **(Test)** DMI Equalization Phase 2 Enable Control.
- UINT32 [DmiGen3EqPh3Method](#): 3
Offset 28:4 : **(Test)** Selects the method for performing Phase3 of Gen3 Equalization on DMI.
- UINT32 [DmiGen3ProgramStaticEq](#): 1
Offset 28:7 : **(Test)** Program DMI Gen3 EQ Phase1 Static Presets.
- UINT32 [RsvdBits0](#): 24
Offset 28:8 :Reserved for future use.
- UINT32 [InitPcieAspmAfterOprom](#): 1
Offset 32:0 : Select when PCIe ASPM programming will happen in relation to the Oprom.
- UINT32 [RsvdBits1](#): 31
Offset 32:1 :Reserved for future use.

- UINT8 [DmiGen3RootPortPreset](#) [SA_DMI_MAX_LANE]
Offset 36 Used for programming DMI Gen3 preset values per lane. Range: 0-9, 8 is default for each lane.
- UINT8 [DmiGen3EndPointPreset](#) [SA_DMI_MAX_LANE]
Offset 40/44 Used for programming DMI Gen3 preset values per lane. Range: 0-9, 7 is default for each lane.
- UINT8 [DmiGen3EndPointHint](#) [SA_DMI_MAX_LANE]
Offset 44/52 Hint value per lane for the DMI Gen3 End Point. Range: 0-6, 2 is default for each lane.
- UINT8 [DmiGen3RxCtlePeaking](#) [SA_DMI_MAX_BUNDLE]
Offset 48/60 : DMI Gen3 RxCTLEp per-Bundle control.
- UINT8 [DmiDeEmphasis](#)
Offset 64 This field is used to describe the DeEmphasis control for DMI (-6 dB and -3.5 dB are the options)
- UINT8 [Rsvd0](#) [3]
Offset 65.

13.140.1 Detailed Description

PCI Express and DMI controller configuration

.

Note

Optional. These policies will be ignored if there is no PEG port present on board. **Revision 1:**

- Initial version.

Definition at line 86 of file CpuPcieConfig.h.

13.140.2 Member Data Documentation

13.140.2.1 DmiGen3EqPh2Enable

UINT32 PCIE_PEI_PREMEM_CONFIG::DmiGen3EqPh2Enable

Offset 28:2 : **(Test)** DMI Equalization Phase 2 Enable Control.

- Disabled (0x0) : Disable phase 2
- Enabled (0x1) : Enable phase 2
- **Auto** (0x2) : Use the current default method (Default)

Definition at line 104 of file CpuPcieConfig.h.

13.140.2.2 DmiGen3EqPh3Method

UINT32 PCIE_PEI_PREMEM_CONFIG::DmiGen3EqPh3Method

Offset 28:4 : **(Test)** Selects the method for performing Phase3 of Gen3 Equalization on DMI.

- **Auto** (0x0) : Use the current default method (Default)
- **HwEq** (0x1) : Use Adaptive Hardware Equalization
- **SwEq** (0x2) : Use Adaptive Software Equalization (Implemented in BIOS Reference Code)
- **Static** (0x3) : Use the Static EQs provided in DmiGen3EndPointPreset array for Phase1 AND Phase3 (Instead of just Phase1)
- **Disabled** (0x4) : Bypass Equalization Phase 3

Definition at line 114 of file CpuPcieConfig.h.

13.140.2.3 DmiGen3ProgramStaticEq

UINT32 PCIE_PEI_PREMEM_CONFIG::DmiGen3ProgramStaticEq

Offset 28:7 : **(Test)** Program DMI Gen3 EQ Phase1 Static Presets.

- **Disabled** (0x0) : Disable EQ Phase1 Static Presets Programming
- **Enabled** (0x1) : Enable EQ Phase1 Static Presets Programming (Default)

Definition at line 121 of file CpuPcieConfig.h.

13.140.2.4 DmiGen3RxCtlePeaking

UINT8 PCIE_PEI_PREMEM_CONFIG::DmiGen3RxCtlePeaking[SA_DMI_MAX_BUNDLE]

Offset 48/60 : DMI Gen3 RxCTLEp per-Bundle control.

The range of the setting is (0-15). This setting has to be specified based upon platform design and must follow the guideline. Default is 12.

Definition at line 142 of file CpuPcieConfig.h.

13.140.2.5 DmiMaxLinkSpeed

UINT32 PCIE_PEI_PREMEM_CONFIG::DmiMaxLinkSpeed

Offset 28:0 : **(Test)** DMI Link Speed Control.

- **Auto** (0x0) : Maximum possible link speed (Default)
- Gen1 (0x1) : Limit Link to Gen1 Speed
- Gen2 (0x2) : Limit Link to Gen2 Speed
- Gen3 (0x3) : Limit Link to Gen3 Speed

Definition at line 96 of file CpuPcieConfig.h.

13.140.2.6 InitPcieAspmAfterOprom

UINT32 PCIE_PEI_PREMEM_CONFIG::InitPcieAspmAfterOprom

Offset 32:0 : Select when PCIe ASPM programming will happen in relation to the Oprom.

- **Before** (0x0) : Do PCIe ASPM programming before Oprom. (Default)
- After (0x1) : Do PCIe ASPM programming after Oprom. This will require an SMI handler to save/restore ASPM settings.

Definition at line 130 of file CpuPcieConfig.h.

The documentation for this struct was generated from the following file:

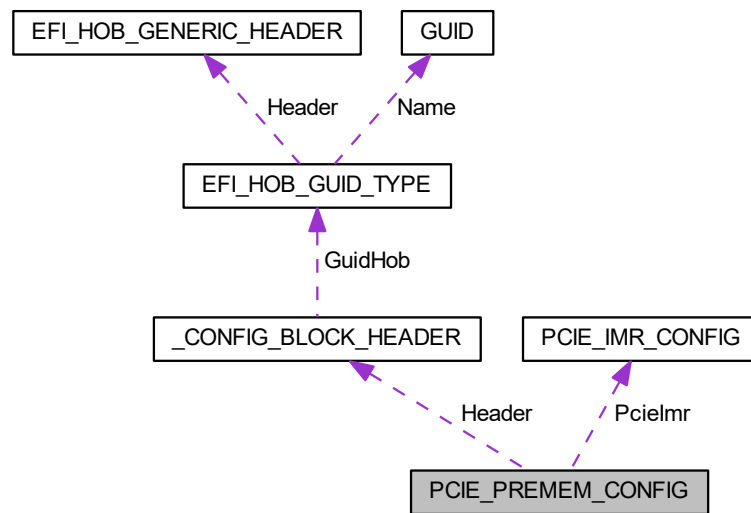
- [CpuPcieConfig.h](#)

13.141 PCIE_PREMEM_CONFIG Struct Reference

PCIe Pre-Memory Configuration **Revision 1**: - Initial version.

```
#include <PciePreMemConfig.h>
```

Collaboration diagram for PCIE_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Offset 0 - 27 Config Block Header.
- [PCIE_IMR_CONFIG PciImr](#)
IMR Configuration.

13.141.1 Detailed Description

PCIe Pre-Memory Configuration **Revision 1**: - Initial version.

Definition at line 65 of file `PciePreMemConfig.h`.

The documentation for this struct was generated from the following file:

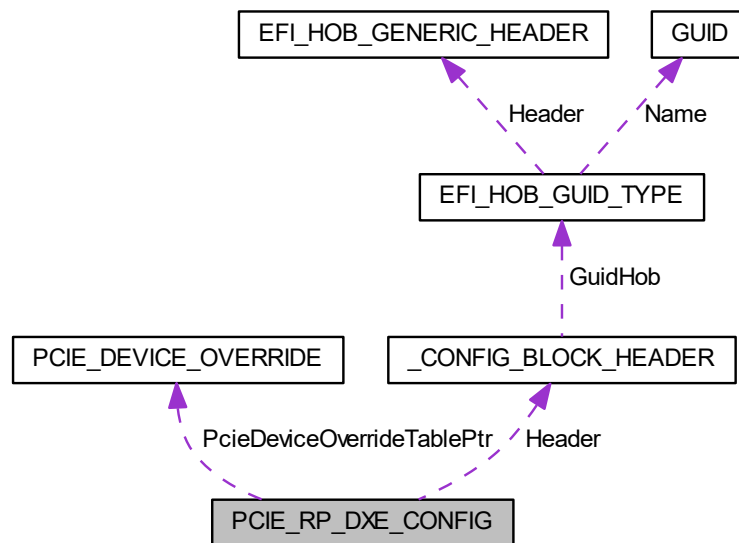
- [PciePreMemConfig.h](#)

13.142 PCIE_RP_DXE_CONFIG Struct Reference

The [PCIE_RP_DXE_CONFIG](#) block describes the expected configuration of the PCH PCI Express controllers in DXE phase.

```
#include <PchPcieRpConfig.h>
```

Collaboration diagram for PCIE_RP_DXE_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [PCIE_DEVICE_OVERRIDE](#) * [PcieDeviceOverrideTablePtr](#)
PCIe device override table The PCIe device table is being used to override PCIe device ASPM settings.

13.142.1 Detailed Description

The [PCIE_RP_DXE_CONFIG](#) block describes the expected configuration of the PCH PCI Express controllers in DXE phase.

Revision 1:

- Init version

Definition at line 269 of file `PchPcieRpConfig.h`.

13.142.2 Member Data Documentation

13.142.2.1 PcieDeviceOverrideTablePtr

[PCIE_DEVICE_OVERRIDE](#)* PCIE_RP_DXE_CONFIG::PcieDeviceOverrideTablePtr

PCIe device override table The PCIe device table is being used to override PCIe device ASPM settings.

And it's only used in DXE phase. Please refer to [PCIE_DEVICE_OVERRIDE](#) structure for the table. Last entry VendorId must be 0.

Definition at line 279 of file PchPcieRpConfig.h.

The documentation for this struct was generated from the following file:

- [PchPcieRpConfig.h](#)

13.143 PMC_GLOBAL_RESET_MASK Union Reference

Description of Global Reset Trigger/Event Mask register.

```
#include <PmConfig.h>
```

13.143.1 Detailed Description

Description of Global Reset Trigger/Event Mask register.

Definition at line 149 of file PmConfig.h.

The documentation for this union was generated from the following file:

- [PmConfig.h](#)

13.144 PMC_INTERFACE_CONFIG Struct Reference

The [PMC_INTERFACE_CONFIG](#) block describes interaction between BIOS and PMC.

```
#include <TcssPeiConfig.h>
```

Public Attributes

- [UINT8 PmcPdEnable](#)
PMC PD Solution Enable.

13.144.1 Detailed Description

The [PMC_INTERFACE_CONFIG](#) block describes interaction between BIOS and PMC.

Definition at line 74 of file TcssPeiConfig.h.

The documentation for this struct was generated from the following file:

- [TcssPeiConfig.h](#)

13.145 PMC_LPM_S0IX_SUB_STATE_EN Union Reference

Low Power Mode Enable config.

```
#include <PmConfig.h>
```

13.145.1 Detailed Description

Low Power Mode Enable config.

Used to configure if respective S0i2/3 sub-states are to be supported by the platform. Each bit corresponds to one LPM state - LPMx->BITx. Some sub-states will require external FETs controlled by EXT_PWR_GATE#/EXT_PWR_GATE2# pins to gate v1p05-PHY or v1p05-IS supplies

Definition at line 116 of file PmConfig.h.

13.145.2 Member Data Documentation

13.145.2.1 S0i2p2En

```
UINT32 PMC_LPM_S0IX_SUB_STATE_EN::S0i2p2En
```

LPM2 - S0i2.2 Enable.

Requires EXT_PWR_GATE# controlled FET to gate v1p05 PHY. Refer to V1p05PhyExtFetControlEn.

Definition at line 125 of file PmConfig.h.

13.145.2.2 S0i3p3En

UINT32 PMC_LPM_S0IX_SUB_STATE_EN::S0i3p3En

LPM5 - S0i3.3 Enable.

Requires EXT_PWR_GATE# controlled FET to gate v1p05 PHY. Refer to V1p05PhyExtFetControlEn.

Definition at line 134 of file PmConfig.h.

13.145.2.3 S0i3p4En

UINT32 PMC_LPM_S0IX_SUB_STATE_EN::S0i3p4En

LPM7 - S0i3.4 Enable.

Requires EXT_PWR_GATE2# controlled FET to gate v1p05-SRAM/ISCLK. Refer to V1p05IsExtFetControlEn.

Definition at line 140 of file PmConfig.h.

The documentation for this union was generated from the following file:

- [PmConfig.h](#)

13.146 PPM_CUSTOM_RATIO_TABLE Struct Reference

This structure is used to describe the custom processor ratio table desired by the platform.

```
#include <CpuPowerMgmtCustomConfig.h>
```

Public Attributes

- UINT8 [MaxRatio](#)
The maximum ratio of the custom ratio table.
- UINT8 [NumberOfEntries](#)
The number of custom ratio state entries, ranges from 2 to 40 for a valid custom ratio table.
- UINT8 [Rsvd0](#) [2]
Reserved for DWORD alignment.
- UINT32 [Cpuid](#)
The CPU ID for which this custom ratio table applies.
- UINT8 [StateRatio](#) [MAX_CUSTOM_RATIO_TABLE_ENTRIES]
The processor ratios in the custom ratio table.
- UINT8 [StateRatioMax16](#) [MAX_16_CUSTOM_RATIO_TABLE_ENTRIES]
If there are more than 16 total entries in the StateRatio table, then use these 16 entries to fill max 16 table.

13.146.1 Detailed Description

This structure is used to describe the custom processor ratio table desired by the platform.

Definition at line 59 of file CpuPowerMgmtCustomConfig.h.

13.146.2 Member Data Documentation

13.146.2.1 StateRatioMax16

```
UINT8 PPM_CUSTOM_RATIO_TABLE::StateRatioMax16[MAX_16_CUSTOM_RATIO_TABLE_ENTRIES]
```

If there are more than 16 total entries in the StateRatio table, then use these 16 entries to fill max 16 table.

Note

If NumberOfEntries is 16 or less, or the first entry of this table is 0, then this table is ignored, and up to the top 16 values from the StateRatio table is used instead.

Definition at line 70 of file CpuPowerMgmtCustomConfig.h.

The documentation for this struct was generated from the following file:

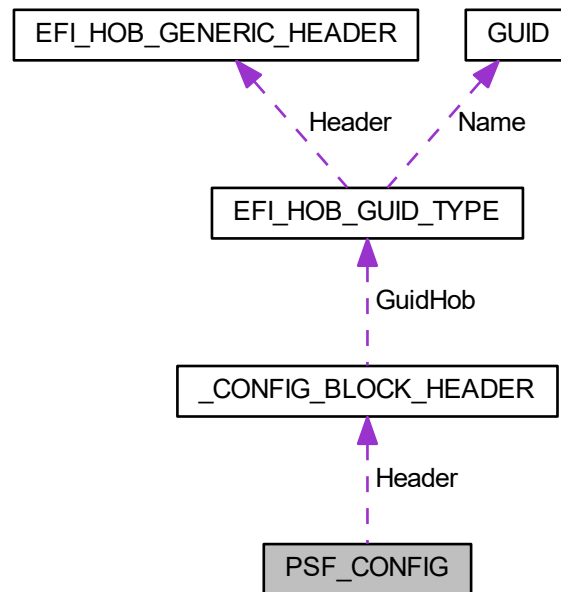
- [CpuPowerMgmtCustomConfig.h](#)

13.147 PSF_CONFIG Struct Reference

The [PSF_CONFIG](#) block describes the expected configuration of the Primary Sideband Fabric.

```
#include <PsfConfig.h>
```

Collaboration diagram for PSF_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Config Block Header.
- UINT32 [TccEnable](#): 1
Psf Tcc (Time Coordinated Computing) Enable will decrease psf transaction latency by disable some psf power management features.
- UINT32 [RsvdBits0](#): 31
Reserved bits.

13.147.1 Detailed Description

The [PSF_CONFIG](#) block describes the expected configuration of the Primary Sideband Fabric.

Definition at line 49 of file PsfConfig.h.

13.147.2 Member Data Documentation

13.147.2.1 TccEnable

```
UINT32 PSF_CONFIG::TccEnable
```

Psf Tcc (Time Coordinated Computing) Enable will decrease psf transaction latency by disable some psf power management features.

0: Disable; 1: Enable.

Definition at line 55 of file PsfConfig.h.

The documentation for this struct was generated from the following file:

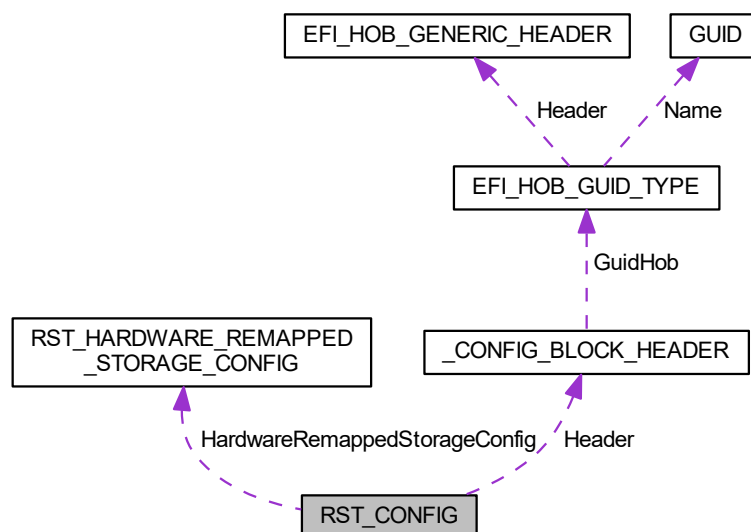
- [PsfConfig.h](#)

13.148 RST_CONFIG Struct Reference

Rapid Storage Technology settings.

```
#include <RstConfig.h>
```

Collaboration diagram for RST_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [Raid0](#): 1
*0 : Disable; 1 : **Enable** RAID0*
- UINT32 [Raid1](#): 1
*0 : Disable; 1 : **Enable** RAID1*
- UINT32 [Raid10](#): 1
*0 : Disable; 1 : **Enable** RAID10*
- UINT32 [Raid5](#): 1
*0 : Disable; 1 : **Enable** RAID5*
- UINT32 [Irrt](#): 1
*0 : Disable; 1 : **Enable** Intel Rapid Recovery Technology*
- UINT32 [OromUiBanner](#): 1
*0 : Disable; 1 : **Enable** OROM UI and BANNER*
- UINT32 [OromUiDelay](#): 2
***00b** : 2 secs; 01b : 4 secs; 10b : 6 secs; 11 : 8 secs (see : SATA_ORM_DELAY)*
- UINT32 [HddUnlock](#): 1
*0 : Disable; 1 : **Enable**. Indicates that the HDD password unlock in the OS is enabled*
- UINT32 [LedLocate](#): 1
*0 : Disable; 1 : **Enable**. Indicates that the LED/SGPIO hardware is attached and ping to locate feature is enabled on the OS*
- UINT32 [IrrtOnly](#): 1
*0 : Disable; 1 : **Enable**. Allow only IRRT drives to span internal and external ports*
- UINT32 [SmartStorage](#): 1
*0 : Disable; 1 : **Enable** RST Smart Storage caching Bit*
- UINT32 [LegacyOrom](#): 1
***0** : **Disable**; 1 : Enable RST Legacy OROM*
- UINT32 [OptaneMemory](#): 1
*0 : Disable; 1 : **Enable** RST Optane(TM) Memory*
- UINT32 [CpuAttachedStorage](#): 1
*0 : Disable; 1 : **Enable** CPU Attached Storage*
- UINT32 [RsvdBits0](#): 17
Reserved Bits.
- [RST_HARDWARE_REMAPPED_STORAGE_CONFIG](#) [HardwareRemappedStorageConfig](#) [PCH_MAX_↔
RST_PCIE_STORAGE_CR]
This member describes the details of implementation of Intel RST for PCIe Storage remapping (Intel RST Driver is required) Note: RST for PCIe Storage remapping is supported only for first SATA controller if more controllers are available.

13.148.1 Detailed Description

Rapid Storage Technology settings.

Revision 1:

- Initial version.

Definition at line 84 of file RstConfig.h.

The documentation for this struct was generated from the following file:

- [RstConfig.h](#)

13.149 RST_HARDWARE_REMAPPED_STORAGE_CONFIG Struct Reference

This structure describes the details of Intel RST for PCIe Storage remapping Note: In order to use this feature, Intel RST Driver is required.

```
#include <RstConfig.h>
```

Public Attributes

- UINT32 [Enable](#): 1
This member describes whether or not the Intel RST for PCIe Storage remapping should be enabled.
- UINT32 [RstPcieStoragePort](#): 5
*Intel RST for PCIe Storage remapping - PCIe Port Selection (1-based, **0 = autodetect**) The supported ports for PCIe Storage remapping is different depend on the platform and cycle router.*
- UINT32 [DeviceResetDelay](#): 8
PCIe Storage Device Reset Delay in milliseconds (ms), which it guarantees such delay gap is fulfilled before PCIe Storage Device configuration space is accessed after an reset caused by the link disable and enable step.
- UINT32 [RsvdBits0](#): 18
Reserved bits.

13.149.1 Detailed Description

This structure describes the details of Intel RST for PCIe Storage remapping Note: In order to use this feature, Intel RST Driver is required.

Definition at line 56 of file RstConfig.h.

13.149.2 Member Data Documentation

13.149.2.1 DeviceResetDelay

```
UINT32 RST_HARDWARE_REMAPPED_STORAGE_CONFIG::DeviceResetDelay
```

PCIe Storage Device Reset Delay in milliseconds (ms), which it guarantees such delay gap is fulfilled before PCIe Storage Device configuration space is accessed after an reset caused by the link disable and enable step.

Default value is **100ms**.

Definition at line 73 of file RstConfig.h.

13.149.2.2 Enable

```
UINT32 RST_HARDWARE_REMAPPED_STORAGE_CONFIG::Enable
```

This member describes whether or not the Intel RST for PCIe Storage remapping should be enabled.

0: Disable; 1: Enable. Note 1: If SATA Controller is disabled, PCIe Storage Remapping should be disabled as well. Note 2: If PCIe Storage remapping is enabled, the PCH integrated AHCI controllers Class Code is configured as RAID.

Definition at line 62 of file RstConfig.h.

The documentation for this struct was generated from the following file:

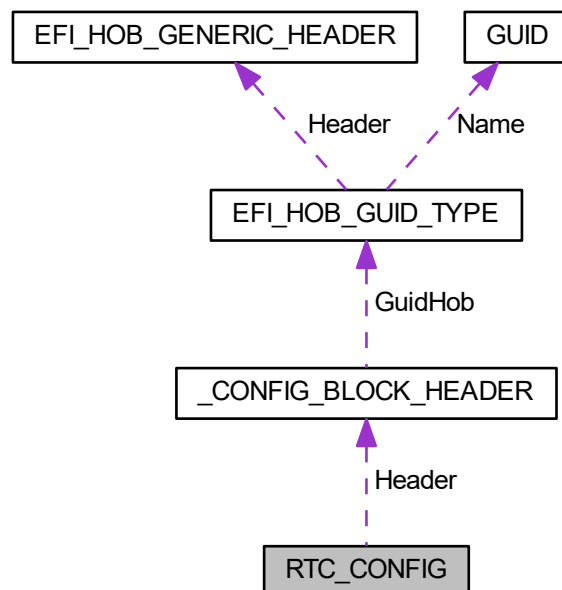
- [RstConfig.h](#)

13.150 RTC_CONFIG Struct Reference

The [RTC_CONFIG](#) block describes the expected configuration of RTC configuration.

```
#include <RtcConfig.h>
```

Collaboration diagram for RTC_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [BiosInterfaceLock](#): 1
When set, prevents RTC TS (BUC.TS) from being changed.
- UINT32 [MemoryLock](#): 1
When set, bytes 38h-3Fh in the upper 128bytes bank of RTC RAM are locked and cannot be accessed.

13.150.1 Detailed Description

The [RTC_CONFIG](#) block describes the expected configuration of RTC configuration.

Definition at line 46 of file RtcConfig.h.

13.150.2 Member Data Documentation

13.150.2.1 BiosInterfaceLock

```
UINT32 RTC_CONFIG::BiosInterfaceLock
```

When set, prevents RTC TS (BUC.TS) from being changed.

This BILD bit has different function compared to LPC/eSPI, SPI. 0: Disabled; **1: Enabled**

Definition at line 53 of file RtcConfig.h.

13.150.2.2 MemoryLock

```
UINT32 RTC_CONFIG::MemoryLock
```

When set, bytes 38h-3Fh in the upper 128bytes bank of RTC RAM are locked and cannot be accessed.

Writes will be droipped and reads will not return any guaranteed data. 0: Disabled; **1: Enabled**

Definition at line 60 of file RtcConfig.h.

The documentation for this struct was generated from the following file:

- [RtcConfig.h](#)

13.151 SA_ADDRESS_DECODE Struct Reference

SA memory address decode.

```
#include <MemoryConfig.h>
```

Public Attributes

- [UINT8 Controller](#)
Offset 0 Zero based Controller number.
- [UINT8 Channel](#)
Offset 1 Zero based Channel number.
- [UINT8 Dimm](#)
Offset 2 Zero based DIMM number.
- [UINT8 Rank](#)
Offset 3 Zero based Rank number.
- [UINT8 BankGroup](#)
Offset 4 Zero based Bank Group number.
- [UINT8 Bank](#)
Offset 5 Zero based Bank number.
- [UINT16 Cas](#)
Offset 6 Zero based CAS number.
- [UINT32 Ras](#)
Offset 8 Zero based RAS number.

13.151.1 Detailed Description

SA memory address decode.

Definition at line 179 of file MemoryConfig.h.

The documentation for this struct was generated from the following file:

- [MemoryConfig.h](#)

13.152 SA_FUNCTION_CALLS Struct Reference

Function calls into the SA.

```
#include <MemoryConfig.h>
```

Public Attributes

- [SA_IO_READ_8 loRead8](#)
Offset 0: - CPU I/O port 8-bit read.
- [SA_IO_READ_16 loRead16](#)
Offset 4: - CPU I/O port 16-bit read.
- [SA_IO_READ_32 loRead32](#)
Offset 8: - CPU I/O port 32-bit read.
- [SA_IO_WRITE_8 loWrite8](#)
Offset 12: - CPU I/O port 8-bit write.
- [SA_IO_WRITE_16 loWrite16](#)
Offset 16: - CPU I/O port 16-bit write.
- [SA_IO_WRITE_32 loWrite32](#)
Offset 20: - CPU I/O port 32-bit write.
- [SA_MMIO_READ_8 MmioRead8](#)
Offset 24: - Memory Mapped I/O port 8-bit read.
- [SA_MMIO_READ_16 MmioRead16](#)
Offset 28: - Memory Mapped I/O port 16-bit read.
- [SA_MMIO_READ_32 MmioRead32](#)
Offset 32: - Memory Mapped I/O port 32-bit read.
- [SA_MMIO_READ_64 MmioRead64](#)
Offset 36: - Memory Mapped I/O port 64-bit read.
- [SA_MMIO_WRITE_8 MmioWrite8](#)
Offset 40: - Memory Mapped I/O port 8-bit write.
- [SA_MMIO_WRITE_16 MmioWrite16](#)
Offset 44: - Memory Mapped I/O port 16-bit write.
- [SA_MMIO_WRITE_32 MmioWrite32](#)
Offset 48: - Memory Mapped I/O port 32-bit write.
- [SA_MMIO_WRITE_64 MmioWrite64](#)
Offset 52: - Memory Mapped I/O port 64-bit write.
- [SA_SMBUS_READ_8 SmbusRead8](#)
Offset 56: - Smbus 8-bit read.
- [SA_SMBUS_READ_16 SmbusRead16](#)
Offset 60: - Smbus 16-bit read.
- [SA_SMBUS_WRITE_8 SmbusWrite8](#)
Offset 64: - Smbus 8-bit write.
- [SA_SMBUS_WRITE_16 SmbusWrite16](#)
Offset 68: - Smbus 16-bit write.
- [SA_GET_PCI_DEVICE_ADDRESS GetPciDeviceAddress](#)
Offset 72: - Get PCI device address.
- [SA_GET_PCIE_DEVICE_ADDRESS GetPcieDeviceAddress](#)
Offset 76: - Get PCI express device address.
- [SA_GET_RTC_TIME GetRtcTime](#)
Offset 80: - Get the current time value.
- [SA_GET_CPU_TIME GetCpuTime](#)
Offset 84: - The current CPU time in milliseconds.
- [SA_MEMORY_COPY CopyMem](#)
Offset 88: - Perform byte copy operation.
- [SA_MEMORY_SET_BYTE SetMem](#)
Offset 92: - Perform byte initialization operation.
- [SA_MEMORY_SET_WORD SetMemWord](#)

- Offset 96: - Perform word initialization operation.
- [SA_MEMORY_SET_DWORD SetMemDword](#)
- Offset 100: - Perform dword initialization operation.
- [SA_LEFT_SHIFT_64 LeftShift64](#)
- Offset 104: - Left shift the 64-bit data value by specified number of bits.
- [SA_RIGHT_SHIFT_64 RightShift64](#)
- Offset 108: - Right shift the 64-bit data value by specified number of bits.
- [SA_MULT_U64_U32 MultU64x32](#)
- Offset 112: - Multiply a 64-bit data value by a 32-bit data value.
- [SA_DIV_U64_U64 DivU64x64](#)
- Offset 116: - Divide a 64-bit data value by a 64-bit data value.
- [SA_GET_SPD_DATA GetSpdData](#)
- Offset 120: - Read the SPD data over the SMBus, at the given Smbus SPD address and copy the data to the data structure.
- [SA_GET_RANDOM_NUMBER GetRandomNumber](#)
- Offset 124: - Get the next random 32-bit number.
- [SA_CPU_MAILBOX_READ CpuMailboxRead](#)
- Offset 128: - Perform a CPU mailbox read.
- [SA_CPU_MAILBOX_WRITE CpuMailboxWrite](#)
- Offset 132: - Perform a CPU mailbox write.
- [SA_GET_MEMORY_VDD GetMemoryVdd](#)
- Offset 136: - Get the current memory voltage (VDD).
- [SA_SET_MEMORY_VDD SetMemoryVdd](#)
- Offset 140: - Set the memory voltage (VDD) to the given value.
- [SA_CHECKPOINT CheckPoint](#)
- Offset 144: - Check point that is called at various points in the MRC.
- [SA_DEBUG_HOOK DebugHook](#)
- Offset 148: - Typically used to display to the I/O port 80h.
- [SA_DEBUG_PRINT DebugPrint](#)
- Offset 152: - Output a string to the debug stream/device.
- [SA_GET_RTC_CMOS GetRtcCmos](#)
- Offset 156: - Get the current value of the specified RTC CMOS location.
- [SA_MSR_READ_64 ReadMsr64](#)
- Offset 160: - Get the current value of the specified MSR location.
- [SA_MSR_WRITE_64 WriteMsr64](#)
- Offset 164: - Set the current value of the specified MSR location.
- [SA_MRC_RETURN_FROM_SMC MrcReturnFromSmc](#)
- Offset 168: - Hook function after returning from MrcStartMemoryConfiguration()
- [SA_MRC_DRAM_RESET MrcDramReset](#)
- Offset 172: - Assert or deassert DRAM_RESET# pin; this is used in JEDEC Reset.
- [SA_DELAY_NS MrcDelayNs](#)
- Offset 176: - Delay (stall) for the given amount of nanoseconds.
- [SA_SET_MEMORY_PMIC_VOLTAGE SetMemoryPmicVoltage](#)
- Offset 180: - Set the memory voltage (VDD/VDDQ/VPP) by PMIC for DDR5.

13.152.1 Detailed Description

Function calls into the SA.

Definition at line 251 of file MemoryConfig.h.

The documentation for this struct was generated from the following file:

- [MemoryConfig.h](#)

13.153 SA_MEMORY_DQDQS_MAPPING Struct Reference

DqDqs Mapping.

```
#include <MemoryConfig.h>
```

Public Attributes

- [UINT8 DqsMapCpu2Dram](#) [[MEM_CFG_MAX_CONTROLLERS](#)][[MEM_CFG_MAX_CHANNELS](#)][[MEM_CFG_NUM_BYTES_MAPPED](#)]
DqsMapCpu2Dram.
- [UINT8 DqMapCpu2Dram](#) [[MEM_CFG_MAX_CONTROLLERS](#)][[MEM_CFG_MAX_CHANNELS](#)][[MEM_CFG_NUM_BYTES_MAPPED](#)][8]
DqMapCpu2Dram.

13.153.1 Detailed Description

DqDqs Mapping.

Definition at line 151 of file MemoryConfig.h.

The documentation for this struct was generated from the following file:

- [MemoryConfig.h](#)

13.154 SA_MEMORY_FUNCTIONS Struct Reference

Function calls into the MRC.

```
#include <MemoryConfig.h>
```

Public Attributes

- [SA_CHANNEL_EXIST](#) [MrcChannelExist](#)
Offset 0: - Returns whether Channel is or is not present.
- [SA_PRINTF](#) [MrcPrintf](#)
Offset 4: - Print to output stream/device.
- [SA_CHANGE_MARGIN](#) [MrcChangeMargin](#)
Offset 8: - Change the margin.
- [SA_SIGN_EXTEND](#) [MrcSignExtend](#)
Offset 12: - Sign extends OldMSB to NewMSB Bits (Eg: Bit 6 to Bit 7).
- [SA_SHIFT_PI_COMMAND_TRAIN](#) [ShiftPiCommandTrain](#)
Offset 16: - Move CMD/CTL/CLK/CKE PIs during training.
- [SA_UPDATE_VREF](#) [MrcUpdateVref](#)
Offset 20: - Update the Vref value and wait until it is stable.

13.154.1 Detailed Description

Function calls into the MRC.

Definition at line 303 of file MemoryConfig.h.

The documentation for this struct was generated from the following file:

- [MemoryConfig.h](#)

13.155 SA_MEMORY_RCOMP Struct Reference

Rcomp Policies.

```
#include <MemoryConfig.h>
```

Public Attributes

- UINT16 [RcompResistor](#)
Offset 0: Reference RCOMP resistors on motherboard ~ 100 ohms.
- UINT16 [RcompTarget](#) [[MRC_MAX_RCOMP_TARGETS](#)]
Offset 1: RCOMP target values for DqOdt, DqDrv, CmdDrv, CtlDrv, ClkDrv.

13.155.1 Detailed Description

Rcomp Policies.

Definition at line 160 of file MemoryConfig.h.

The documentation for this struct was generated from the following file:

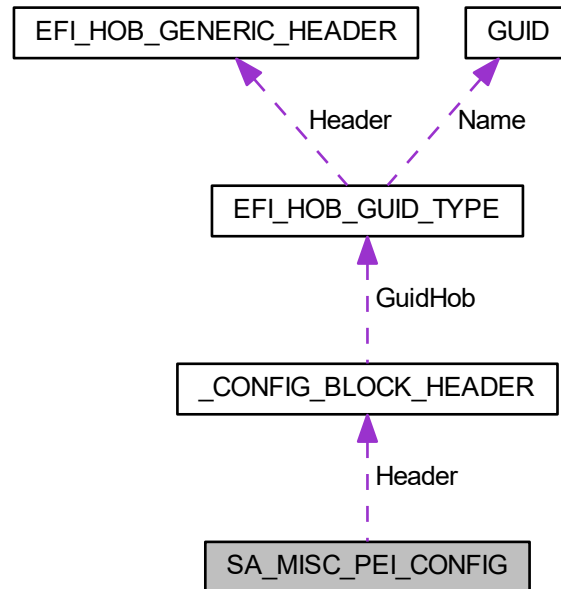
- [MemoryConfig.h](#)

13.156 SA_MISC_PEI_CONFIG Struct Reference

This configuration block is to configure SA Miscellaneous variables during PEI Post-Mem.

```
#include <SaMiscPeiConfig.h>
```

Collaboration diagram for SA_MISC_PEI_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.

13.156.1 Detailed Description

This configuration block is to configure SA Miscellaneous variables during PEI Post-Mem.

Revision 1:

- Initial version.

Definition at line 47 of file `SaMiscPeiConfig.h`.

The documentation for this struct was generated from the following file:

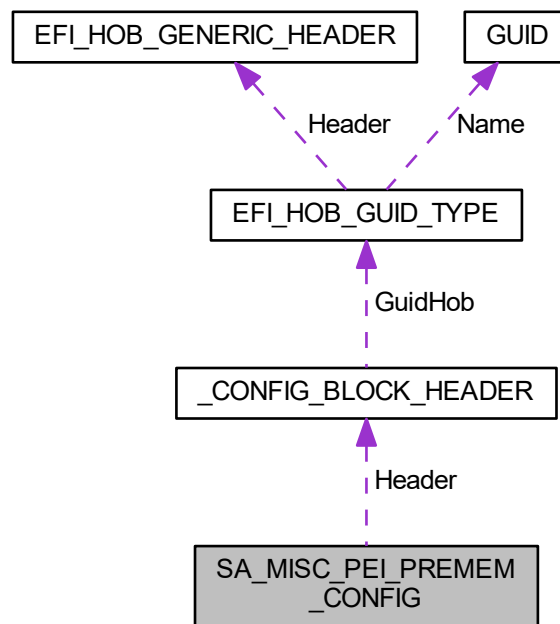
- [SaMiscPeiConfig.h](#)

13.157 SA_MISC_PEI_PREMEM_CONFIG Struct Reference

This configuration block is to configure SA Miscellaneous variables during PEI Pre-Mem phase like programming different System Agent BARs, TsegSize, MmioSize required etc.

```
#include <SaMiscPeiPreMemConfig.h>
```

Collaboration diagram for SA_MISC_PEI_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Offset 0-27 Config Block Header.
- [UINT8 SpdAddressTable](#) [MEM_CFG_MAX_SOCKETS]
Offset 28 Memory DIMMs' SPD address for reading SPD data.
- [VOID * S3DataPtr](#)
Offset 44 Memory data save pointer for S3 resume. The memory space should be allocated and filled with proper S3 resume data on a resume path.
- [UINT32 SmbusBar](#)
Offset 48 Address of System Agent SMBUS BAR: 0xEFA0
- [UINT32 TsegSize](#)
Offset 52 Size of TSEG in bytes.
- [UINT32 IedSize](#)
Offset 56 (Test) Size of IED region in bytes.
- [UINT32 SkipExtGfxScan](#):1

(Test) Offset 60:0 :1=Skip External Gfx Device Scan; 0=Scan for external graphics devices. Set this policy to skip External Graphics card scanning if the platform uses Internal Graphics only.

- UINT32 [BdatEnable](#):1

Offset 60:1 :This field enables the generation of the BIOS DATA ACPI Tables: 0=FALSE, 1=TRUE.

- UINT32 [TxtImplemented](#):1

Offset 60:2 :This field currently is used to tell MRC if it should run after TXT initialization completed: 0=Run without waiting for TXT, 1=Run after TXT initialization by callback.

- UINT32 [ScanExtGfxForLegacyOpRom](#):1

Offset 60:3 : (Test) Scan External Discrete Graphics Devices for Legacy Only VGA OpROMs.

- UINT32 [RsvdBits0](#):28

Offset 60:4 :Reserved for future use.

- UINT8 [UserBd](#)

Offset 64 0=Mobile/Mobile Halo, 1=Desktop/DT Halo, 2=Desktop 2DPC DDR5, 5=ULT/ULX/Mobile Halo Type3, 6=ULT/ULX/Mobile Halo Type4, 8=UP Server.

- UINT8 [LockPTMregs](#)

(Test) Offset 65 Lock PCU Thermal Management registers: 0=FALSE, 1=TRUE

- UINT8 [BdatTestType](#)

Offset 66 When BdatEnable is set to TRUE, this option selects the type of data which will be populated in the BIOS Data ACPI Tables: 0=RMT, 1=RMT Per Bit, 2=Margin 2D.

- UINT8 [CridEnable](#)

Offset 67 For Platforms supporting Intel(R) SIPP, this policy is use control enable/disable Compatibility Revision ID (CRID) feature: 0=FALSE, 1=TRUE.

- UINT32 [AcpiReservedMemorySize](#)

Offset 68 The Size of a Reserved memory buffer allocated in previous boot for S3 resume used. Originally it is retrieved from AcpiVariableCompatibility variable.

- UINT32 [OpRomScanTempMmioBar](#)

(Test) Offset 72 Temporary address to MMIO map OpROMs during VGA scanning. Used for ScanExtGfxForLegacyOpRom feature. MUST BE 16MB ALIGNED!

- UINT32 [OpRomScanTempMmioLimit](#)

(Test) Offset 76 Limit address for OpROM MMIO range. Used for ScanExtGfxForLegacyOpRom feature. (OpROMScanTempMmioLimit - OpRomScanTempMmioBar) MUST BE >= 16MB!

- UINT64 [AcpiReservedMemoryBase](#)

Offset 80 The Base address of a Reserved memory buffer allocated in previous boot for S3 resume used. Originally it is retrieved from AcpiVariableCompatibility variable.

- UINT64 [SystemMemoryLength](#)

Offset 88 Total system memory length from previous boot, this is required for S3 resume. Originally it is retrieved from AcpiVariableCompatibility variable.

- UINT8 [WrcFeatureEnable](#)

Offset 96: Enable/Disable WRC (Write Cache) feature of IOP. When enabled, supports IO devices allocating onto the ring and into LLC.

- UINT8 [FirstDimmBitMask](#)

Offset 97: Defines which DIMM should be populated first on a 2DPC board.

- UINT8 [FirstDimmBitMaskEcc](#)

Offset 98: Defines which ECC DIMM should be populated first on a 2DPC board.

- UINT8 [DisableMrcRetrainingOnRtcPowerLoss](#)

Offset 99: Enable/Disable DisableMrcRetrainingOnRtcPowerLoss.

- UINT8 [Rsvd](#) [4]

Reserved for config block alignment.

13.157.1 Detailed Description

This configuration block is to configure SA Miscellaneous variables during PEI Pre-Mem phase like programming different System Agent BARs, TsegSize, MmioSize required etc.

Revision 1:

- Initial version. **Revision 2:**
- Deprecate ledSize. **Revision 3:**
- Adding ResizableBarSupport.

Definition at line 56 of file SaMiscPeiPreMemConfig.h.

13.157.2 Member Data Documentation

13.157.2.1 FirstDimmBitMask

```
UINT8 SA_MISC_PEI_PREMEM_CONFIG::FirstDimmBitMask
```

Offset 97: Defines which DIMM should be populated first on a 2DPC board.

4 bit mask: Bit[0]: MC0 DIMM0, Bit[1]: MC0 DIMM1, Bit[2]: MC1 DIMM0, Bit[3]: MC1 DIMM1. For each MC, the first DIMM to be populated should be set to '1'. Note: this mask is only for non-ECC DIMM.

Definition at line 124 of file SaMiscPeiPreMemConfig.h.

13.157.2.2 FirstDimmBitMaskEcc

```
UINT8 SA_MISC_PEI_PREMEM_CONFIG::FirstDimmBitMaskEcc
```

Offset 98: Defines which ECC DIMM should be populated first on a 2DPC board.

4 bit mask: Bit[0]: MC0 DIMM0, Bit[1]: MC0 DIMM1, Bit[2]: MC1 DIMM0, Bit[3]: MC1 DIMM1. For each MC, the first DIMM to be populated should be set to '1'. For example, if one MC is T-topology, there is no special population rule, can put it as 11 for this MC and it means either D0 or D1 can be be populated firstly. Note: this mask is only for ECC DIMM, not for non-ECC DIMM.

Definition at line 128 of file SaMiscPeiPreMemConfig.h.

13.157.2.3 IedSize

UINT32 SA_MISC_PEI_PREMEM_CONFIG::IedSize

Offset 56 (**Test**) Size of IED region in bytes.

0 : IED Disabled (no memory occupied) 0x400000 : 4MB SMM memory occupied by IED (Part of TSEG) **Note: Enabling IED may also enlarge TsegSize together.**

Deprecated

Definition at line 91 of file SaMiscPeiPreMemConfig.h.

13.157.2.4 ScanExtGfxForLegacyOpRom

UINT32 SA_MISC_PEI_PREMEM_CONFIG::ScanExtGfxForLegacyOpRom

Offset 60:3 : (**Test**) Scan External Discrete Graphics Devices for Legacy Only VGA OpROMs.

When enabled, if the primary graphics device is an external discrete graphics device, Si will scan the graphics device for legacy only VGA OpROMs.

This is intended to ease the implementation of a BIOS feature to automatically enable CSM if the Primary Gfx device only supports Legacy VBIOS (No UEFI GOP Present). Otherwise disabling CSM won't result in no video being displayed. This is useful for platforms that implement PCIe slots that allow the end user to install an arbitrary Gfx device.

This setting will only take effect if SkipExtGfxScan == 0. It is ignored otherwise.

- Disabled (0x0) : Don't Scan for Legacy Only VGA OpROMs (Default)
- **Enabled** (0x1) : Scan External Gfx for Legacy Only VGA OpROM

Definition at line 111 of file SaMiscPeiPreMemConfig.h.

13.157.2.5 SpdAddressTable

UINT8 SA_MISC_PEI_PREMEM_CONFIG::SpdAddressTable[MEM_CFG_MAX_SOCKETS]

Offset 28 Memory DIMMs' SPD address for reading SPD data.

TGL Mapping 0 - Controller 0 Channel 0 Dimm 0 - DDR4 - DDR5 - LPDDR4 - LPDDR5 1 - Controller 0 Channel 0 Dimm 1 - DDR4 2 - Controller 0 Channel 1 Dimm 0 ----- DDR5 - LPDDR4 - LPDDR5 3 - Controller 0 Channel 1 Dimm 1 ----- DDR5 2DPC 4 - Controller 0 Channel 2 Dimm 0 ----- LPDDR4 - LPDDR5 6 - Controller 0 Channel 3 Dimm 0 ----- LPDDR4 - LPDDR5 8 - Controller 1 Channel 0 Dimm 0 - DDR4 - DDR5 - LPDDR4 - LPDDR5 9 - Controller 1 Channel 0 Dimm 1 - DDR4 10 - Controller 1 Channel 1 Dimm 0 ----- DDR5 - LPDDR4 - LPDDR5 11 - Controller 1 Channel 1 Dimm 1 ----- DDR5 2DPC 12 - Controller 1 Channel 2 Dimm 0 ----- LPDDR4 - LPDDR5 14 - Controller 1 Channel 3 Dimm 0 ----- LPDDR4 - LPDDR5

Definition at line 74 of file SaMiscPeiPreMemConfig.h.

13.157.2.6 TsegSize

```
UINT32 SA_MISC_PEI_PREMEM_CONFIG::TsegSize
```

Offset 52 Size of TSEG in bytes.

(Must be power of 2) **0x400000**: 4MB for Release build (When IED enabled, it will be 8MB) 0x1000000 : 16MB for Debug build (Regardless IED enabled or disabled)

Definition at line 82 of file SaMiscPeiPreMemConfig.h.

The documentation for this struct was generated from the following file:

- [SaMiscPeiPreMemConfig.h](#)

13.158 SA_XDCI_IRQ_INT_CONFIG Struct Reference

The SA XDCI INT Pin and IRQ number.

```
#include <TcssPeiConfig.h>
```

Public Attributes

- UINT8 [IntPing](#)
Int Pin Number.
- UINT8 [Irq](#)
Irq Number.

13.158.1 Detailed Description

The SA XDCI INT Pin and IRQ number.

Definition at line 82 of file TcssPeiConfig.h.

The documentation for this struct was generated from the following file:

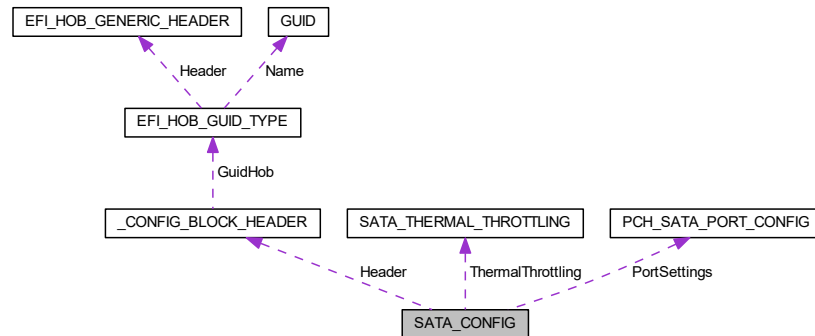
- [TcssPeiConfig.h](#)

13.159 SATA_CONFIG Struct Reference

The [SATA_CONFIG](#) block describes the expected configuration of the SATA controllers.

```
#include <SataConfig.h>
```

Collaboration diagram for SATA_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT8 [Enable](#)
This member describes whether or not the SATA controllers should be enabled.
- UINT8 [TestMode](#)
(Test) 0: Disable; 1: Allow entrance to the PCH SATA test modes
- UINT8 [SalpSupport](#)
0: Disable; 1: Enable Aggressive Link Power Management
- UINT8 [PwrOptEnable](#)
0: Disable; 1: Enable SATA Power Optimizer on PCH side.
- UINT8 [EsataSpeedLimit](#)
EsataSpeedLimit When enabled, BIOS will configure the PxSCTL.SPD to 2 to limit the eSATA port speed.
- UINT8 [LedEnable](#)
SATA LED indicates SATA controller activity. 0: Disable; 1: Enable SATA LED.
- UINT8 [RaidDeviceld](#)
This option allows to configure SATA controller device ID while in RAID mode.
- UINT8 [SataRstInterrupt](#)
Controls which interrupts will be linked to SATA controller CAP list This option will take effect only if SATA controller is in RAID mode Default: PchSataMsix
- UINT8 [SataMode](#)
Determines the system will be configured to which SATA mode.
- UINT8 [SpeedLimit](#)
Indicates the maximum speed the SATA controller can support.
- UINT8 [EnclosureSupport](#)
Enclosure Management Support. 0: Disable; 1: Enable.
- UINT8 [SgpioSupport](#)
Controls whenever Serial GPIO support is enabled for controller 0: Disable; 1: Enable.
- [PCH_SATA_PORT_CONFIG](#) PortSettings [PCH_MAX_SATA_PORTS]
This member configures the features, property, and capability for each SATA port.
- [SATA_THERMAL_THROTTLING](#) ThermalThrottling
This field decides the settings of Sata thermal throttling.

13.159.1 Detailed Description

The [SATA_CONFIG](#) block describes the expected configuration of the SATA controllers.

Revision 1:

- Initial version. **Revision 2:**
- Added DevSlpGpioPinMux.

Definition at line 134 of file SataConfig.h.

13.159.2 Member Data Documentation

13.159.2.1 Enable

```
UINT8 SATA_CONFIG::Enable
```

This member describes whether or not the SATA controllers should be enabled.

0: Disable; 1: **Enable**.

Definition at line 139 of file SataConfig.h.

13.159.2.2 EsataSpeedLimit

```
UINT8 SATA_CONFIG::EsataSpeedLimit
```

EsataSpeedLimit When enabled, BIOS will configure the PxSCTL.SPD to 2 to limit the eSATA port speed.

Please be noted, this setting could be cleared by HBA reset, which might be issued by EFI AHCI driver when POST time, or by SATA inbox driver/RST driver after POST. To support the Speed Limitation when POST, the EFI AHCI driver should preserve the setting before and after initialization. For support it after POST, it's dependent on driver's behavior. **0: Disable**; 1: Enable

Definition at line 153 of file SataConfig.h.

13.159.2.3 RaidDeviceld

```
UINT8 SATA_CONFIG::RaidDeviceId
```

This option allows to configure SATA controller device ID while in RAID mode.

Refer to SATA_RAID_DEV_ID enumeration for supported options. Choosing Client will allow RST driver loading, RSTe driver will not be able to load Choosing Alternate will not allow RST inbox driver loading in Windows Choosing Server will allow RSTe driver loading, RST driver will not load **0: Client**; 1: Alternate; 2: Server

Definition at line 163 of file SataConfig.h.

13.159.2.4 SataMode

```
UINT8 SATA_CONFIG::SataMode
```

Determines the system will be configured to which SATA mode.

Refer to SATA_MODE enumeration for supported options. Default is **SataModeAhci**.

Definition at line 175 of file SataConfig.h.

13.159.2.5 SpeedLimit

```
UINT8 SATA_CONFIG::SpeedLimit
```

Indicates the maximum speed the SATA controller can support.

Refer to SATA_SPEED enumeration for supported options. **0h: SataSpeedDefault**; 1h: 1.5 Gb/s (Gen 1); 2h: 3 Gb/s(Gen 2); 3h: 6 Gb/s (Gen 1)

Definition at line 181 of file SataConfig.h.

13.159.2.6 ThermalThrottling

```
SATA_THERMAL_THROTTLING SATA_CONFIG::ThermalThrottling
```

This field decides the settings of Sata thermal throttling.

When the Suggested Setting is enabled, PCH RC will use the suggested representative values.

Definition at line 196 of file SataConfig.h.

The documentation for this struct was generated from the following file:

- [SataConfig.h](#)

13.160 SATA_THERMAL_THROTTLING Struct Reference

This structure lists PCH supported SATA thermal throttling register setting for customization.

```
#include <SataConfig.h>
```

Public Attributes

- UINT32 [P0T1M](#): 2
Port 0 T1 Multiplier.
- UINT32 [P0T2M](#): 2
Port 0 T2 Multiplier.
- UINT32 [P0T3M](#): 2
Port 0 T3 Multiplier.
- UINT32 [P0TDisp](#): 2
Port 0 Tdispatch.
- UINT32 [P1T1M](#): 2
Port 1 T1 Multiplier.
- UINT32 [P1T2M](#): 2
Port 1 T2 Multiplier.
- UINT32 [P1T3M](#): 2
Port 1 T3 Multiplier.
- UINT32 [P1TDisp](#): 2
Port 1 Tdispatch.
- UINT32 [P0Tinact](#): 2
Port 0 Tinactive.
- UINT32 [P0TDispFinit](#): 1
Port 0 Alternate Fast Init Tdispatch.
- UINT32 [P1Tinact](#): 2
Port 1 Tinactive.
- UINT32 [P1TDispFinit](#): 1
Port 1 Alternate Fast Init Tdispatch.
- UINT32 [SuggestedSetting](#): 1
*0: Disable; 1: **Enable** suggested representative values*
- UINT32 [RsvdBits0](#): 9
Reserved bits.

13.160.1 Detailed Description

This structure lists PCH supported SATA thermal throttling register setting for customization.

The settings is programmed through SATA Index/Data registers. When the SuggestedSetting is enabled, the customized values are ignored.

Definition at line 107 of file SataConfig.h.

The documentation for this struct was generated from the following file:

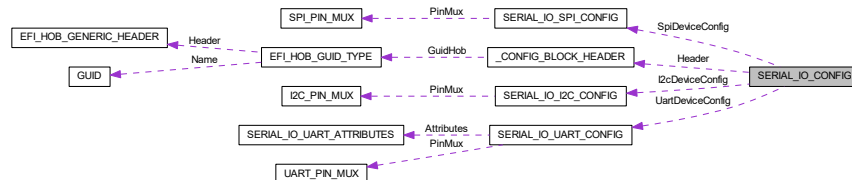
- [SataConfig.h](#)

13.161 SERIAL_IO_CONFIG Struct Reference

The [SERIAL_IO_CONFIG](#) block provides the configurations to set the Serial IO controllers.

```
#include <SerialIoConfig.h>
```

Collaboration diagram for SERIAL_IO_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [SERIAL_IO_SPI_CONFIG](#) [SpiDeviceConfig](#) [PCH_MAX_SERIALIO_SPI_CONTROLLERS]
SPI Configuration.
- [SERIAL_IO_I2C_CONFIG](#) [I2cDeviceConfig](#) [PCH_MAX_SERIALIO_I2C_CONTROLLERS]
I2C Configuration.
- [SERIAL_IO_UART_CONFIG](#) [UartDeviceConfig](#) [PCH_MAX_SERIALIO_UART_CONTROLLERS]
UART Configuration.

13.161.1 Detailed Description

The [SERIAL_IO_CONFIG](#) block provides the configurations to set the Serial IO controllers.

Revision 1:

- Initial version. **Revision 2:**
- Modified SPI Config to support pinmux functionality.

Definition at line 53 of file SerialIoConfig.h.

The documentation for this struct was generated from the following file:

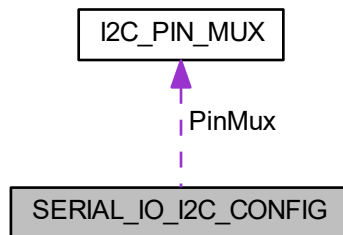
- [SerialIoConfig.h](#)

13.162 SERIAL_IO_I2C_CONFIG Struct Reference

Serial IO I2C Controller Configuration.

```
#include <SerialIoDevices.h>
```

Collaboration diagram for SERIAL_IO_I2C_CONFIG:



Public Attributes

- `UINT8` [PadTermination](#)
SerialIoI2cPci see SERIAL_IO_I2C_MODE
- [I2C_PIN_MUX](#) [PinMux](#)
I2C pin mux configuration.

13.162.1 Detailed Description

Serial IO I2C Controller Configuration.

Definition at line 236 of file SerialIoDevices.h.

13.162.2 Member Data Documentation

13.162.2.1 PadTermination

```
UINT8 SERIAL_IO_I2C_CONFIG::PadTermination
```

SerialIoI2cPci see **SERIAL_IO_I2C_MODE**

I2C Pads Internal Termination. For more information please see Platform Design Guide. Supported values (check GPIO_ELECTRICAL_CONFIG for reference): **GpioTermNone: No termination**, GpioTermWpu1K: 1kOhm weak pull-up, GpioTermWpu5K: 5kOhm weak pull-up, GpioTermWpu20K: 20kOhm weak pull-up

Definition at line 247 of file SerialIoDevices.h.

The documentation for this struct was generated from the following file:

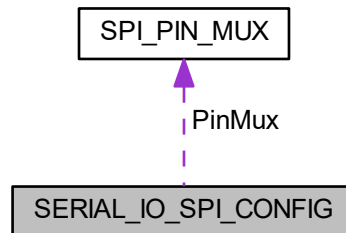
- [SerialIoDevices.h](#)

13.163 SERIAL_IO_SPI_CONFIG Struct Reference

The [SERIAL_IO_SPI_CONFIG](#) provides the configurations to set the Serial IO SPI controller.

```
#include <SerialIoDevices.h>
```

Collaboration diagram for SERIAL_IO_SPI_CONFIG:



Public Attributes

- [UINT8 Mode](#)
SerialIoSpiPci see SERIAL_IO_SPI_MODE
- [UINT8 DefaultCsOutput](#)
0 = CS0 CS1, CS2, CS3. Default CS used by the SPI HC
- [UINT8 CsPolarity](#) [PCH_MAX_SERIALIO_SPI_CHIP_SELECTS]
Selects SPI ChipSelect signal polarity, 0 = low 1 = High
- [UINT8 CsEnable](#) [PCH_MAX_SERIALIO_SPI_CHIP_SELECTS]
0 = Enable 1 = Disable. Based on this setting GPIO for given SPIx CSx will be configured in Native mode
- [UINT8 CsMode](#)
0 = HW Control 1 = SW Control. Sets Chip Select Control mode Hardware or Software.
- [UINT8 CsState](#)
0 = CS is set to low 1 = CS is set to high
- [SPI_PIN_MUX PinMux](#)
SPI Pinmux configuration.

13.163.1 Detailed Description

The [SERIAL_IO_SPI_CONFIG](#) provides the configurations to set the Serial IO SPI controller.

Definition at line 94 of file SerialIoDevices.h.

The documentation for this struct was generated from the following file:

- [SerialIoDevices.h](#)

13.164 SERIAL_IO_UART_ATTRIBUTES Struct Reference

UART Settings.

```
#include <SerialIoDevices.h>
```

Public Attributes

- UINT32 [BaudRate](#)
115200 Max 6000000 MdePkg.dec PcdUartDefaultBaudRate
- UINT8 [Parity](#)
1 - No Parity see *EFI_PARITY_TYPE* MdePkg.dec PcdUartDefaultParity
- UINT8 [DataBits](#)
8 MdePkg.dec PcdUartDefaultDataBits
- UINT8 [StopBits](#)
1 - One Stop Bit see *EFI_STOP_BITS_TYPE* MdePkg.dec PcdUartDefaultStopBits
- UINT8 [AutoFlow](#)
FALSE IntelFrameworkModulePkg.dsc PcdIsaBusSerialUseHalfHandshake

13.164.1 Detailed Description

UART Settings.

Definition at line 151 of file SerialIoDevices.h.

The documentation for this struct was generated from the following file:

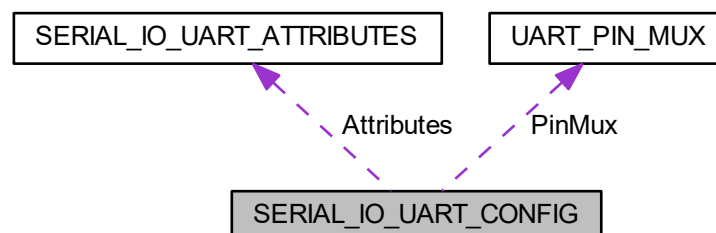
- [SerialIoDevices.h](#)

13.165 SERIAL_IO_UART_CONFIG Struct Reference

Serial IO UART Controller Configuration.

```
#include <SerialIoDevices.h>
```

Collaboration diagram for SERIAL_IO_UART_CONFIG:



Public Attributes

- [SERIAL_IO_UART_ATTRIBUTES Attributes](#)
see SERIAL_IO_UART_ATTRIBUTES
- [UART_PIN_MUX PinMux](#)
UART pin mux configuration.
- [UINT8 Mode](#)
SerialloUartPci *see SERIAL_IO_UART_MODE*
- [UINT8 DBG2](#)
FALSE *If TRUE adds UART to DBG2 table and overrides UartPg to SerialloUartPgDisabled*
- [UINT8 PowerGating](#)
SerialloUartPgAuto *Applies to Hidden/COM/SkipInit see SERIAL_IO_UART_PG*
- [UINT8 DmaEnable](#)
TRUE *Applies to SerialloUartPci only. Informs OS driver to use DMA, if false it will run in PIO mode*

13.165.1 Detailed Description

Serial IO UART Controller Configuration.

Definition at line 174 of file SerialloDevices.h.

The documentation for this struct was generated from the following file:

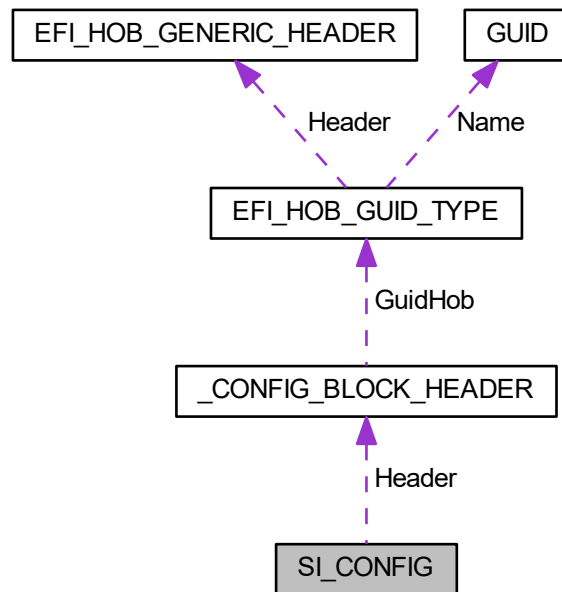
- [SerialloDevices.h](#)

13.166 SI_CONFIG Struct Reference

The Silicon Policy allows the platform code to publish a set of configuration information that the RC drivers will use to configure the silicon hardware.

```
#include <SiConfig.h>
```

Collaboration diagram for SI_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0 - 27 Config Block Header.
- [UINT8 CsmFlag](#)
offset 28 CSM status flag.
- [UINT8 SkipSsidProgramming](#)
This is used to skip the SSID programming in silicon code.
- [UINT8 RsvdBytes0](#) [2]
offset 30 - 31
- [UINT16 CustomizedSvid](#)
When SkipSsidProgramming is FALSE, silicon code will use this as default value to program the SVID for all internal devices.
- [UINT16 CustomizedSsid](#)
When SkipSsidProgramming is FALSE, silicon code will use this as default value to program the Sid for all internal devices.
- [UINT32 * SsidTablePtr](#)
SsidTablePtr contains the SVID_SID_INIT_ENTRY table.
- [UINT16 NumberOfSsidTableEntry](#)
Number of valid enties in SsidTablePtr.
- [UINT8 RsvdBytes1](#) [2]
offset 42 - 43
- [UINT8 SkipBiosDoneWhenFwUpdate](#)
This is used to skip setting BIOS_DONE MSR during firmware update boot mode.
- [UINT8 RsvdBytes2](#) [3]
Offset 45 - 47.

13.166.1 Detailed Description

The Silicon Policy allows the platform code to publish a set of configuration information that the RC drivers will use to configure the silicon hardware.

Revision 1: - Initial version. **Revision 2:** - Deprecate CsmFlag

Definition at line 52 of file SiConfig.h.

13.166.2 Member Data Documentation

13.166.2.1 CsmFlag

```
UINT8 SI_CONFIG::CsmFlag
```

offset 28 CSM status flag.

Deprecated

Definition at line 57 of file SiConfig.h.

13.166.2.2 CustomizedSsid

```
UINT16 SI_CONFIG::CustomizedSsid
```

When SkipSsidProgramming is FALSE, silicon code will use this as default value to program the Sid for all internal devices.

0: use silicon default SSID 0x7270 , Non-zero: use customized SSID. offset 34 - 35

Definition at line 77 of file SiConfig.h.

13.166.2.3 CustomizedSvid

```
UINT16 SI_CONFIG::CustomizedSvid
```

When SkipSsidProgramming is FALSE, silicon code will use this as default value to program the SVID for all internal devices.

0: use silicon default SVID 0x8086 , Non-zero: use customized SVID. offset 32 - 33

Definition at line 71 of file SiConfig.h.

13.166.2.4 NumberOfSsidTableEntry

UINT16 SI_CONFIG::NumberOfSsidTableEntry

Number of valid enties in SsidTablePtr.

This is valid when SkipSsidProgramming is FALSE; **Default is 0.** offset 40 - 41

Definition at line 116 of file SiConfig.h.

13.166.2.5 SkipBiosDoneWhenFwUpdate

UINT8 SI_CONFIG::SkipBiosDoneWhenFwUpdate

This is used to skip setting BIOS_DONE MSR during firmware update boot mode.

When set to TRUE and boot mode is BOOT_ON_FLASH_UPDATE, skip setting BIOS_DONE MSR at EndofPei.
0: FALSE, 1: TRUE Offset 44

Definition at line 124 of file SiConfig.h.

13.166.2.6 SkipSsidProgramming

UINT8 SI_CONFIG::SkipSsidProgramming

This is used to skip the SSID programming in silicon code.

When set to TRUE, silicon code will not do any SSID programming and platform code needs to handle that by itself properly. **0: FALSE, 1: TRUE** offset 29

Definition at line 64 of file SiConfig.h.

13.166.2.7 SsidTablePtr

UINT32* SI_CONFIG::SsidTablePtr

SsidTablePtr contains the SVID_SID_INIT_ENTRY table.

This is valid when SkipSsidProgramming is FALSE; It doesn't need to contain entries for all Intel internal devices. It can only contains the SVID_SID_INIT_ENTRY entries for those Dev# Func# which needs to be overridden. In the enties, only Dev, Function, SubSystemVendorId, and SubSystemId are required. **Default is NULL.**

E.g. Platform only needs to override BDF 0:31:5 to AAAA:BBBB and BDF 0:31:3 to CCCC:DDDD, it can be done in platform like this: `STATIC SVID_SID_INIT_ENTRY mSsidTablePtr[SI_MAX_DEVICE_COUNT] = {0};`

```
VOID SiPolicyUpdate () { UINT32 EntryCount = 0; SiPolicy->SkipSsidProgramming = FALSE; SiPolicy->SsidTablePtr = mSsidTablePtr;
```

```
mSsidTablePtr[EntryCount].Address.Bits.Device = SpiDeviceNumber (); mSsidTablePtr[EntryCount].Address.Bits.Function = SpiFunctionNumber (); mSsidTablePtr[EntryCount].SvidSidValue.SubSystemVendorId = 0xAAAA; mSsidTablePtr[EntryCount].SvidSidValue.SubSystemId = 0xBBBB; EntryCount ++; mSsidTablePtr[EntryCount].Address.Bits.Device = HdaDevNumber (); mSsidTablePtr[EntryCount].Address.Bits.Function = HdaFunctionNumber (); mSsidTablePtr[EntryCount].SvidSidValue.SubSystemVendorId = 0xCCCC; mSsidTablePtr[EntryCount].SvidSidValue.SubSystemId = 0xDDDD; EntryCount ++; ASSERT (EntryCount < SI_MAX_DEVICE_COUNT); SiPolicy->NumberOfSsidTableEntry = EntryCount; } offset 36 - 39
```

Definition at line 110 of file SiConfig.h.

The documentation for this struct was generated from the following file:

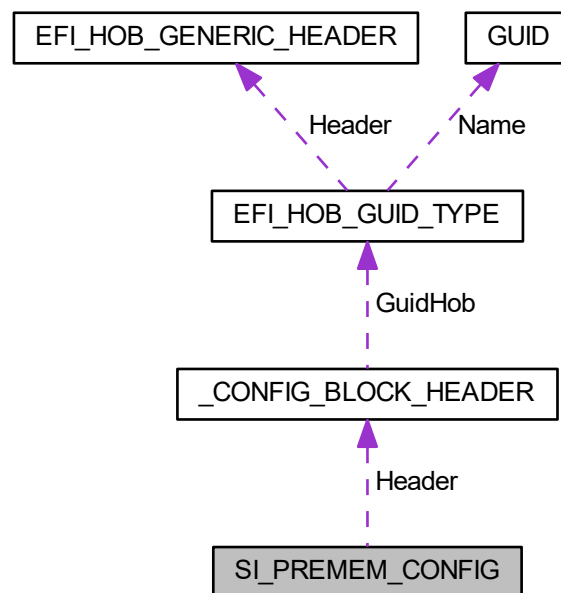
- [SiConfig.h](#)

13.167 SI_PREMEM_CONFIG Struct Reference

The Silicon PreMem Policy allows the platform code to publish a set of configuration information that the RC drivers will use to configure the silicon hardware.

```
#include <SiPreMemConfig.h>
```

Collaboration diagram for SI_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Offset 0 - 27 Config Block Header.
- [UINT32 PlatformDebugConsent](#): 4
Platform Debug Consent As a main switch to enable platform debug capability and relevant settings with specified probe type.
- [UINT32 RsvdBits](#): 28
offset 28 - 31
- [UINT8 SkipOverrideBootModeWhenFwUpdate](#)
This is used to skip override boot mode during firmware update boot mode.
- [UINT8 RsvdBytes](#) [3]
offset 32

13.167.1 Detailed Description

The Silicon PreMem Policy allows the platform code to publish a set of configuration information that the RC drivers will use to configure the silicon hardware.

Revision 1:

- Initial version.

Definition at line 58 of file SiPreMemConfig.h.

13.167.2 Member Data Documentation

13.167.2.1 PlatformDebugConsent

```
UINT32 SI_PREMEM_CONFIG::PlatformDebugConsent
```

Platform Debug Consent As a main switch to enable platform debug capability and relevant settings with specified probe type.

Manual: Do not use Platform Debug Consent to override other debug-relevant policies, but the user must set each debug option manually, aimed at advanced users.

PDC-dependent policies are listed: DciPreMemConfig->DciEn CpuTraceHubConfig->EnableMode CpuTraceHubConfig->CpuTraceHubMemReg0Size CpuTraceHubConfig->CpuTraceHubMemReg1Size PchTraceHubPreMemConfig->EnableMode PchTraceHubPreMemConfig->MemReg0Size PchTraceHubPreMemConfig->MemReg1Size

Note: DCI OOB (aka BSSB) uses CCA probe. Refer to definition of PLATFORM_DEBUG_CONSENT_PROBE_TYPE **0:Disabled**; 2:DCI OOB (All + TraceHubs); 6:2-wire DCI OOB (Low power); 7:Manual

Definition at line 78 of file SiPreMemConfig.h.

13.167.2.2 SkipOverrideBootModeWhenFwUpdate

```
UINT8 SI_PREMEM_CONFIG::SkipOverrideBootModeWhenFwUpdate
```

This is used to skip override boot mode during firmware update boot mode.

When set to TRUE and boot mode is BOOT_ON_FLASH_UPDATE, skip setting boot mode to BOOT_WITH_FU_LL_CONFIGURATION in PEI memory init. **0: FALSE**, 1: TRUE

Definition at line 86 of file SiPreMemConfig.h.

The documentation for this struct was generated from the following file:

- [SiPreMemConfig.h](#)

13.168 SMBIOS_STRUCTURE Struct Reference

The Smbios structure header.

```
#include <FirmwareVersionInfoHob.h>
```

13.168.1 Detailed Description

The Smbios structure header.

Definition at line 48 of file FirmwareVersionInfoHob.h.

The documentation for this struct was generated from the following file:

- [FirmwareVersionInfoHob.h](#)

13.169 SPD_DATA_BUFFER Struct Reference

SPD Data Buffer.

```
#include <MemoryConfig.h>
```

Public Attributes

- `UINT8 SpdData [MEM_CFG_MAX_CONTROLLERS][MEM_CFG_MAX_CHANNELS][MEM_CFG_MAX_DIMMS][MEM_CFG_MAX_SPD_SIZE]`
SpdData.

13.169.1 Detailed Description

SPD Data Buffer.

Definition at line 143 of file MemoryConfig.h.

The documentation for this struct was generated from the following file:

- [MemoryConfig.h](#)

13.170 SPD_OFFSET_TABLE Struct Reference

SPD Offset Table.

```
#include <MemoryConfig.h>
```

Public Attributes

- UINT16 [Start](#)
Offset 0.
- UINT16 [End](#)
Offset 2.
- UINT8 [BootMode](#)
Offset 4.
- UINT8 [Reserved3](#) [3]
Offset 5 Reserved for future use.

13.170.1 Detailed Description

SPD Offset Table.

Definition at line 169 of file MemoryConfig.h.

The documentation for this struct was generated from the following file:

- [MemoryConfig.h](#)

13.171 SPI_PIN_MUX Struct Reference

SPI signals pin muxing settings.

```
#include <SerialIoDevices.h>
```

Public Attributes

- UINT32 [Cs](#) [PCH_MAX_SERIALIO_SPI_CHIP_SELECTS]
*CS Pin mux configuration. Refer to GPIO_*_MUXING_SERIALIO_SPIx_CS_*.*
- UINT32 [Clk](#)
*CLK Pin mux configuration. Refer to GPIO_*_MUXING_SERIALIO_SPIx_CLK_*.*
- UINT32 [Miso](#)
*MISO Pin mux configuration. Refer to GPIO_*_MUXING_SERIALIO_SPIx_MISO_*.*
- UINT32 [Mosi](#)
*MOSI Pin mux configuration. Refer to GPIO_*_MUXING_SERIALIO_SPIx_MOSI_*.*

13.171.1 Detailed Description

SPI signals pin muxing settings.

If signal can be enable only on a single pin then this parameter is ignored by RC. Refer to GPIO_*_MUXING_SERIALIO_SPIx_* in GpioPins*.h for supported settings on a given platform

Definition at line 84 of file SerialIoDevices.h.

The documentation for this struct was generated from the following file:

- [SerialIoDevices.h](#)

13.172 SVID_SID_VALUE Struct Reference

Subsystem Vendor ID / Subsystem ID.

```
#include <SiConfig.h>
```

13.172.1 Detailed Description

Subsystem Vendor ID / Subsystem ID.

Definition at line 137 of file SiConfig.h.

The documentation for this struct was generated from the following file:

- [SiConfig.h](#)

13.173 TCSS_DEVEN_PEI_PREMEM_CONFIG Union Reference

The [TCSS_DEVEN_PEI_PREMEM_CONFIG](#) block describes Device Enable settings for TCSS.

```
#include <TcssPeiPreMemConfig.h>
```

Public Attributes

- UINT32 [TcssDevEn](#)
Maps to bits in TCSS_DEVEN_0_0_0_MCHBAR_IMPH.

13.173.1 Detailed Description

The [TCSS_DEVEN_PEI_PREMEM_CONFIG](#) block describes Device Enable settings for TCSS.

Definition at line 48 of file TcssPeiPreMemConfig.h.

The documentation for this union was generated from the following file:

- [TcssPeiPreMemConfig.h](#)

13.174 TCSS_IOM_ORI_OVERRIDE Struct Reference

The [TCSS_IOM_PEI_CONFIG](#) block describes IOM Aux/HSL override settings for TCSS.

```
#include <TcssPeiConfig.h>
```

Public Attributes

- [UINT16 AuxOri](#)
Bits defining value for IOM Aux Orientation Register.
- [UINT16 HslOri](#)
Bits defining value for IOM HSL Orientation Register.

13.174.1 Detailed Description

The [TCSS_IOM_PEI_CONFIG](#) block describes IOM Aux/HSL override settings for TCSS.

Definition at line 134 of file `TcssPeiConfig.h`.

The documentation for this struct was generated from the following file:

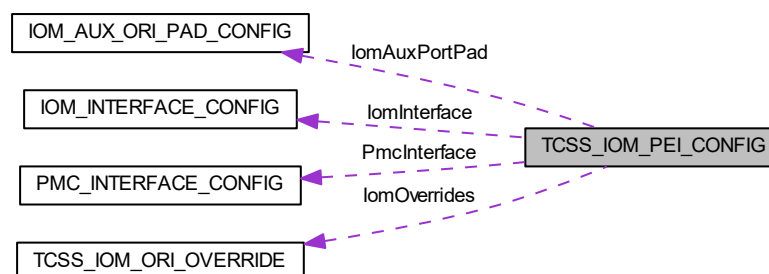
- [TcssPeiConfig.h](#)

13.175 TCSS_IOM_PEI_CONFIG Struct Reference

The [TCSS_IOM_PEI_CONFIG](#) block describes IOM settings for TCSS.

```
#include <TcssPeiConfig.h>
```

Collaboration diagram for `TCSS_IOM_PEI_CONFIG`:



Public Attributes

- [IOM_AUX_ORI_PAD_CONFIG lomAuxPortPad](#) [`MAX_IOM_AUX_BIAS_COUNT`]
The IOM_AUX_ORI_BIAS_CTRL port config setting.
- [IOM_INTERFACE_CONFIG lomInterface](#)
Config settings are BIOS <-> IOM interface.
- [PMC_INTERFACE_CONFIG PmcInterface](#)
Config settings for BIOS <-> PMC interface.
- [UINT8 TcStateLimit](#)
Tcss C-State deep stage.
- [UINT8 Reserved](#) [3]
Reserved bytes for future use.

13.175.1 Detailed Description

The [TCSS_IOM_PEI_CONFIG](#) block describes IOM settings for TCSS.

Definition at line 142 of file TcssPeiConfig.h.

The documentation for this struct was generated from the following file:

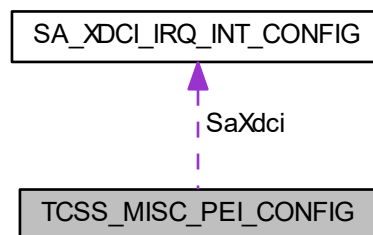
- [TcssPeiConfig.h](#)

13.176 TCSS_MISC_PEI_CONFIG Struct Reference

The [TCSS_MISC_PEI_CONFIG](#) block describes MISC settings for TCSS.

```
#include <TcssPeiConfig.h>
```

Collaboration diagram for TCSS_MISC_PEI_CONFIG:



Public Attributes

- [SA_XDCI_IRQ_INT_CONFIG](#) [SaXdcI](#)
System Agent XdcI Int Pin and Irq setting.
- UINT8 [EnableTcssCovTypeA](#) [MAX_TCSS_USB3_PORTS]
Enable / Disable Type C Port X Convert to TypeA.
- UINT8 [MappingPchXhciUsbA](#) [MAX_TCSS_USB3_PORTS]
PCH xhci port number for Type C Port x mapping.
- UINT8 [IomStayInTCColdSeconds](#)
Set IOM stay in TC cold seconds.
- UINT8 [IomBeforeEnteringTCColdSeconds](#)
Set IOM before entering TC cold seconds.
- UINT8 [Rsvd](#) [2]
Reserved bytes for future use.

13.176.1 Detailed Description

The [TCSS_MISC_PEI_CONFIG](#) block describes MISC settings for TCSS.

Definition at line 154 of file TcssPeiConfig.h.

The documentation for this struct was generated from the following file:

- [TcssPeiConfig.h](#)

13.177 TCSS_MISC_PEI_PREMEM_CONFIG Struct Reference

The [TCSS_MISC_PEI_PREMEM_CONFIG](#) block describes MISC settings for TCSS.

```
#include <TcssPeiPreMemConfig.h>
```

Public Attributes

- UINT8 [PcieMultipleSegmentEnabled](#)
This is policy to control Multiple Segment setting.
- UINT8 [Rsvd](#) [3]
Deprecate.

13.177.1 Detailed Description

The [TCSS_MISC_PEI_PREMEM_CONFIG](#) block describes MISC settings for TCSS.

Revision 1: - Initial version.

Definition at line 76 of file TcssPeiPreMemConfig.h.

13.177.2 Member Data Documentation

13.177.2.1 PcieMultipleSegmentEnabled

```
UINT8 TCSS_MISC_PEI_PREMEM_CONFIG::PcieMultipleSegmentEnabled
```

This is policy to control Multiple Segment setting.

When Disabled all the iTBT PCIe RP are located at Segment0 When Enabled all the iTBT PCIe RP are located at Segment1, FSP Wrapper need to update PCIEXBAR and PcdPciExpressRegionLength to 512MB prior to Fspm↔ WrapperPeim. **0: Disable**, 1: Enable

Definition at line 85 of file TcssPeiPreMemConfig.h.

13.177.2.2 Rsvd

```
UINT8 TCSS_MISC_PEI_PREMEM_CONFIG::Rsvd[3]
```

Deprecate.

Reserved bytes for future use, align to multiple 4

Definition at line 86 of file TcssPeiPreMemConfig.h.

The documentation for this struct was generated from the following file:

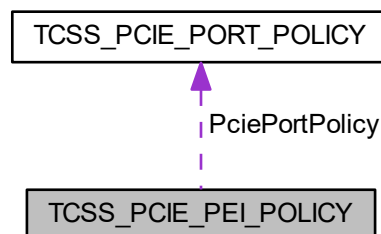
- [TcssPeiPreMemConfig.h](#)

13.178 TCSS_PCIE_PEI_POLICY Struct Reference

[TCSS_PCIE_PEI_POLICY](#) describes PCIe port settings for TCSS.

```
#include <TcssPeiConfig.h>
```

Collaboration diagram for TCSS_PCIE_PEI_POLICY:



13.178.1 Detailed Description

[TCSS_PCIE_PEI_POLICY](#) describes PCIe port settings for TCSS.

Definition at line 127 of file TcssPeiConfig.h.

The documentation for this struct was generated from the following file:

- [TcssPeiConfig.h](#)

13.179 TCSS_PCIE_PORT_POLICY Struct Reference

The [TCSS_PCIE_PORT_POLICY](#) block describes PCIe settings for TCSS.

```
#include <TcssPeiConfig.h>
```

Public Attributes

- [UINT8 AcsEnabled](#)
*Indicate whether the ACS is enabled. 0: Disable; 1: **Enable**.*
- [UINT8 DpcEnabled](#)
*Downstream Port Containment. 0: Disable; 1: **Enable***
- [UINT8 RpDpcExtensionsEnabled](#)
*RP Extensions for Downstream Port Containment. 0: Disable; 1: **Enable***
- [UINT8 LtrEnable](#)
*Latency Tolerance Reporting Mechanism. 0: **Disable**; 1: Enable.*
- [UINT8 PtmEnabled](#)
Enables PTM capability.
- [UINT8 Aspm](#)
The ASPM configuration of the root port (see: [PCH_PCIE_ASPM_CONTROL](#)). Default is
- [UINT8 SlotNumber](#)
Indicates the slot number for the root port. Default is the value as root port index.
- [UINT8 SlotPowerLimitScale](#)
*(Test) Specifies scale used for slot power limit value. Leave as 0 to set to default. Default is **zero**.*
- [UINT16 SlotPowerLimitValue](#)
*(Test) Specifies upper limit on power supplies by slot. Leave as 0 to set to default. Default is **zero**.*
- [UINT8 AdvancedErrorReporting](#)
*Indicate whether the Advanced Error Reporting is enabled. 0: **Disable**; 1: Enable.*
- [UINT8 UnsupportedRequestReport](#)
*Indicate whether the Unsupported Request Report is enabled. 0: **Disable**; 1: Enable.*
- [UINT8 FatalErrorReport](#)
*Indicate whether the Fatal Error Report is enabled. 0: **Disable**; 1: Enable.*
- [UINT8 NoFatalErrorReport](#)
*Indicate whether the No Fatal Error Report is enabled. 0: **Disable**; 1: Enable.*
- [UINT8 CorrectableErrorReport](#)
*Indicate whether the Correctable Error Report is enabled. 0: **Disable**; 1: Enable.*
- [UINT8 SystemErrorOnFatalError](#)
*Indicate whether the System Error on Fatal Error is enabled. 0: **Disable**; 1: Enable.*
- [UINT8 SystemErrorOnNonFatalError](#)
*Indicate whether the System Error on Non Fatal Error is enabled. 0: **Disable**; 1: Enable.*
- [UINT8 SystemErrorOnCorrectableError](#)
*Indicate whether the System Error on Correctable Error is enabled. 0: **Disable**; 1: Enable.*
- [UINT16 LtrMaxSnoopLatency](#)
Latency Tolerance Reporting, Max Snoop Latency.
- [UINT16 LtrMaxNoSnoopLatency](#)
Latency Tolerance Reporting, Max Non-Snoop Latency.
- [UINT8 SnoopLatencyOverrideMode](#)
Latency Tolerance Reporting, Snoop Latency Override Mode.
- [UINT8 SnoopLatencyOverrideMultiplier](#)
Latency Tolerance Reporting, Snoop Latency Override Multiplier.

- UINT16 [SnoopLatencyOverrideValue](#)
Latency Tolerance Reporting, Snoop Latency Override Value.
- UINT8 [NonSnoopLatencyOverrideMode](#)
Latency Tolerance Reporting, Non-Snoop Latency Override Mode.
- UINT8 [NonSnoopLatencyOverrideMultiplier](#)
Latency Tolerance Reporting, Non-Snoop Latency Override Multiplier.
- UINT16 [NonSnoopLatencyOverrideValue](#)
Latency Tolerance Reporting, Non-Snoop Latency Override Value.
- UINT8 [ForceLtrOverride](#)
0: Disable; 1: Enable.
- UINT8 [LtrConfigLock](#)
0: Disable; 1: Enable.

13.179.1 Detailed Description

The [TCSS_PCIE_PORT_POLICY](#) block describes PCIe settings for TCSS.

Definition at line 91 of file TcssPeiConfig.h.

The documentation for this struct was generated from the following file:

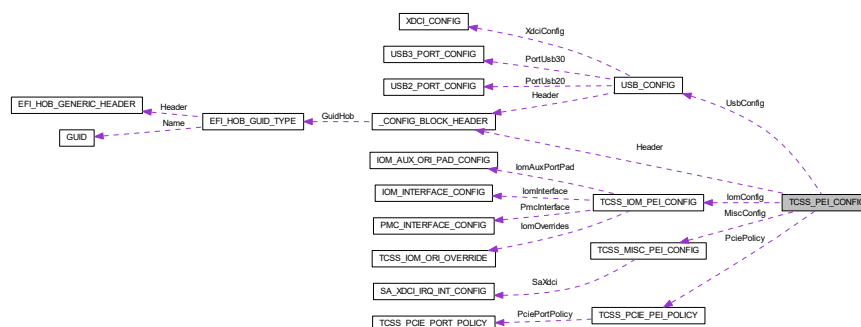
- [TcssPeiConfig.h](#)

13.180 TCSS_PEI_CONFIG Struct Reference

The [TCSS_PEI_CONFIG](#) block describes TCSS settings for SA.

```
#include <TcssPeiConfig.h>
```

Collaboration diagram for TCSS_PEI_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.
- [TCSS_PCIE_PEI_POLICY](#) PciePolicy
The PCIe Config.
- [USB_CONFIG](#) UsbConfig
USB config is shared between PCH and SA.
- [TCSS_IOM_PEI_CONFIG](#) IomConfig
The Iom Config.
- [TCSS_MISC_PEI_CONFIG](#) MiscConfig
The MISC Config.

13.180.1 Detailed Description

The [TCSS_PEI_CONFIG](#) block describes TCSS settings for SA.

Definition at line 166 of file `TcssPeiConfig.h`.

The documentation for this struct was generated from the following file:

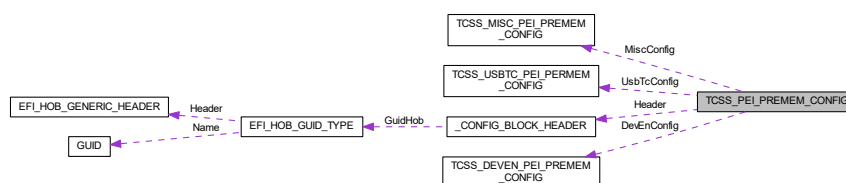
- [TcssPeiConfig.h](#)

13.181 TCSS_PEI_PREMEM_CONFIG Struct Reference

This configuration block describes TCSS settings.

```
#include <TcssPeiPreMemConfig.h>
```

Collaboration diagram for TCSS_PEI_PREMEM_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.
- [TCSS_DEVEN_PEI_PREMEM_CONFIG](#) DevEnConfig
TCSS DEVEN.
- [TCSS_USBTC_PEI_PERMEM_CONFIG](#) UsbTcConfig
USB Type C Port Configuration.
- [TCSS_MISC_PEI_PREMEM_CONFIG](#) MiscConfig
The MISC PreMem Config.

13.181.1 Detailed Description

This configuration block describes TCSS settings.

Revision 1:

- Initial version. **Revision 2:**
- Deprecate `PcieMultipleSegmentEnabled`

Definition at line 96 of file `TcssPeiPreMemConfig.h`.

The documentation for this struct was generated from the following file:

- [TcssPeiPreMemConfig.h](#)

13.182 TCSS_USBTC_PEI_PERMEM_CONFIG Struct Reference

The [TCSS_USBTC_PEI_PERMEM_CONFIG](#) block describes IOM settings for TCSS.

```
#include <TcssPeiPreMemConfig.h>
```

Public Attributes

- UINT32 [UsbTcPortEn](#): 4
bitmap for USB Type C enabled ports
- UINT32 [Rsvd](#): 28
Reserved bytes for future use.

13.182.1 Detailed Description

The [TCSS_USBTC_PEI_PERMEM_CONFIG](#) block describes IOM settings for TCSS.

Definition at line 67 of file `TcssPeiPreMemConfig.h`.

The documentation for this struct was generated from the following file:

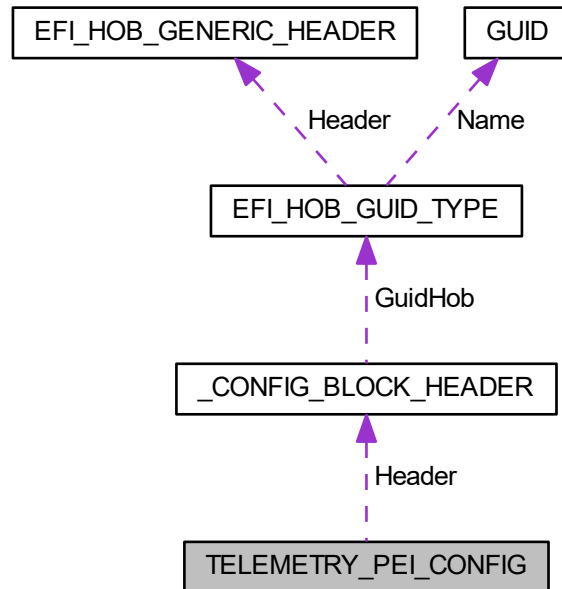
- [TcssPeiPreMemConfig.h](#)

13.183 TELEMETRY_PEI_CONFIG Struct Reference

This configuration block describes Telemetry settings in PostMem.

```
#include <TelemetryPeiConfig.h>
```

Collaboration diagram for TELEMETRY_PEI_CONFIG:



Public Attributes

- UINT32 [CpuCrashLogEnable](#)
Config Block Header.

13.183.1 Detailed Description

This configuration block describes Telemetry settings in PostMem.

Revision 1:

- Initial version.

Definition at line 63 of file `TelemetryPeiConfig.h`.

The documentation for this struct was generated from the following file:

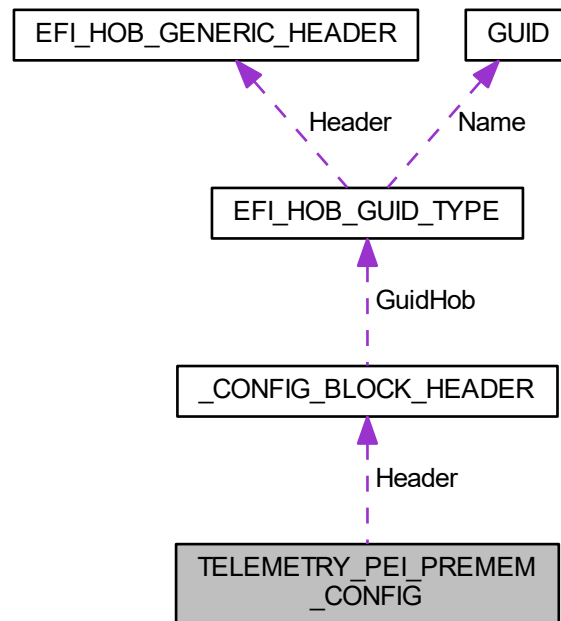
- [TelemetryPeiConfig.h](#)

13.184 TELEMETRY_PEI_PREMEM_CONFIG Struct Reference

This configuration block describes Telemetry settings in PreMem.

```
#include <TelemetryPeiConfig.h>
```

Collaboration diagram for TELEMETRY_PEI_PREMEM_CONFIG:



Public Attributes

- UINT32 [CpuCrashLogDevice](#)
Config Block Header.

13.184.1 Detailed Description

This configuration block describes Telemetry settings in PreMem.

Revision 1:

- Initial version.

Definition at line 53 of file `TelemetryPeiConfig.h`.

The documentation for this struct was generated from the following file:

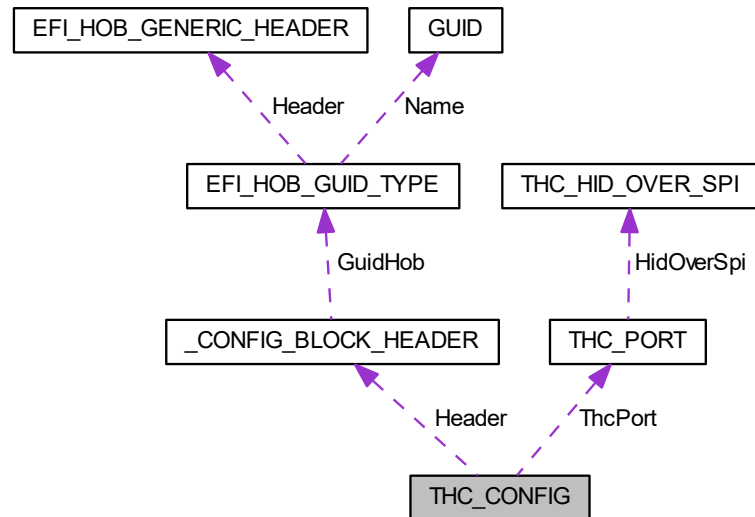
- [TelemetryPeiConfig.h](#)

13.185 THC_CONFIG Struct Reference

[THC_CONFIG](#) block provides the configurations for Touch Host Controllers.

```
#include <ThcConfig.h>
```

Collaboration diagram for THC_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [THC_PORT](#) ThcPort [2]
Port Configuration.

13.185.1 Detailed Description

[THC_CONFIG](#) block provides the configurations for Touch Host Controllers.

Assignment field in each THC port controls the THC behavior.

Available scenarios: 1: Single Port 0 used by THC0

- THC0 Enabled
- Port0 assigned to THC0
- Port1 unassigned
- THC1 will be automatically Disabled. 2: Both ports used by THC0

- THC0 Enabled
- Port0 assigned to THC0
- Port1 assigned to THC0
- THC1 will be automatically Disabled. 3: Port 0 used by THC0 and Port 1 used by THC1
- THC0 Enabled
- Port0 assigned to THC0
- THC1 Enabled
- Port1 assigned to THC1. **4: Both Ports unassigned.** Both THC Controllers will be disabled in that case.

Note

Invalid scenario that will cause ASSERT.

1. Same port Number assigned to THC0 or THC1.
2. Two Ports assigned to THC1.

Revision 1:

- Initial version. **Revision 2:**
- Add HidOverSpi THC_MODE **Revision 3:**
- Expand HidOverSpi support **Revision 4:**
- Add ResetSequencingDelay in [THC_HID_OVER_SPI](#)

Definition at line 135 of file ThcConfig.h.

The documentation for this struct was generated from the following file:

- [ThcConfig.h](#)

13.186 THC_HID_OVER_SPI Struct Reference

THC Hid Over SPI mode related settings.

```
#include <ThcConfig.h>
```


Public Attributes

- UINT32 [ResetPad](#)
Reset Pad - 0x0 = THC HW default BIOS will pass Pad information to ACPI, this Pad is used in _RST function or by the BIOS Quick SPI driver.
- UINT32 [ResetPadTrigger](#)
Reset Pad Trigger.
- UINT32 [Frequency](#)
Connection Speed - SPI Frequency in Hz.
- UINT32 [InputReportHeaderAddress](#)
Input Report Header Address.
- UINT32 [InputReportBodyAddress](#)
Input Report Body Address.
- UINT32 [OutputReportAddress](#)
Output Report Address.
- UINT32 [ReadOpcode](#)
Read Opcode.
- UINT32 [WriteOpcode](#)
Write Opcode.
- UINT32 [Flags](#)
Flags.

13.186.1 Detailed Description

THC Hid Over SPI mode related settings.

Only applicable when HidOverSpi is enabled.

Definition at line 66 of file ThcConfig.h.

The documentation for this struct was generated from the following file:

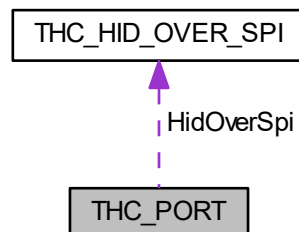
- [ThcConfig.h](#)

13.187 THC_PORT Struct Reference

Port Configuration structure required for each Port that THC might use.

```
#include <ThcConfig.h>
```

Collaboration diagram for THC_PORT:



Public Attributes

- UINT32 [Assignment](#)
Sets THCx assignment see THC_PORT_ASSIGNMENT.
- UINT32 [InterruptPinMuxing](#)
*Each GPIO PORTx/SPIx INTB Pin has different muxing options refer to GPIO_*_MUXING_THC_SPIx_*.*
- UINT8 [WakeOnTouch](#)
*1 = Enable 0 = **Disable**. Based on this setting vGPIO for given THC will be in native mode, and additional _CRS for wake will be exposed in ACPI*
- UINT8 [Mode](#)
*1 = Hid over SPI 0 = **THC**. Switch between Intel THC protocol and Industry standard HID Over SPI protocol.*
- UINT8 [Reserved](#) [2]
Reserved.
- [THC_HID_OVER_SPI](#) [HidOverSpi](#)
Hid Over Spi mode settings.
- UINT32 [ActiveLtr](#)
*0xFFFFFFFF = **Driver Default** Active Ltr*
- UINT32 [IdleLtr](#)
*0xFFFFFFFF = **Driver Default** Idle Ltr*
- UINT32 [LimitPacketSize](#)
Limit Packet Size.
- UINT32 [PerformanceLimitation](#)
Performance Limitation.
- UINT32 [HidOverSpiResetSequencingDelay](#)
Policy control for reset sequencing delay (_INI, _RST)

13.187.1 Detailed Description

Port Configuration structure required for each Port that THC might use.

Definition at line 82 of file ThcConfig.h.

The documentation for this struct was generated from the following file:

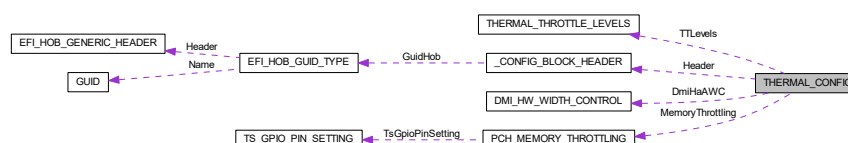
- [ThcConfig.h](#)

13.188 THERMAL_CONFIG Struct Reference

The [THERMAL_CONFIG](#) block describes the expected configuration of the Thermal IP block.

```
#include <ThermalConfig.h>
```

Collaboration diagram for THERMAL_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [PchHotEnable](#): 1
*Enable PCHHOT# pin assertion when temperature is higher than PchHotLevel. 0: **Disabled**, 1: **Enabled**.*
- [THERMAL_THROTTLE_LEVELS](#) TTLevels
This field decides the settings of Thermal throttling.
- [DMI_HW_WIDTH_CONTROL](#) DmiHaAWC
This field decides the settings of DMI throttling.
- [PCH_MEMORY_THROTTLING](#) MemoryThrottling
Memory Thermal Management settings.
- UINT16 [PchHotLevel](#)
The recommendation is the same as Cat Trip point.

13.188.1 Detailed Description

The [THERMAL_CONFIG](#) block describes the expected configuration of the Thermal IP block.

Definition at line 145 of file ThermalConfig.h.

13.188.2 Member Data Documentation

13.188.2.1 DmiHaAWC

[DMI_HW_WIDTH_CONTROL](#) THERMAL_CONFIG::DmiHaAWC

This field decides the settings of DMI throttling.

When the Suggested Setting is enabled, PCH RC will use the suggested representative values.

Definition at line 158 of file ThermalConfig.h.

13.188.2.2 PchHotLevel

UINT16 THERMAL_CONFIG::PchHotLevel

The recommendation is the same as Cat Trip point.

This field decides the temperature, default is **120**. Temperature value used for PCHHOT# pin assertion based on 2s complement format

- 0x001 positive 1°C
- 0x000 0°C
- 0x1FF negative 1°C
- 0x1D8 negative 40°C
- and so on

Definition at line 173 of file ThermalConfig.h.

13.188.2.3 TTLevels

`THERMAL_THROTTLE_LEVELS` `THERMAL_CONFIG::TTLevels`

This field decides the settings of Thermal throttling.

When the Suggested Setting is enabled, PCH RC will use the suggested representative values.

Definition at line 153 of file ThermalConfig.h.

The documentation for this struct was generated from the following file:

- [ThermalConfig.h](#)

13.189 THERMAL_THROTTLE_LEVELS Struct Reference

This structure lists PCH supported throttling register setting for customization.

```
#include <ThermalConfig.h>
```

Public Attributes

- `UINT32 T0Level`: 9
Customized T0Level value. If SuggestedSetting is used, this setting is ignored.
- `UINT32 T1Level`: 9
Customized T1Level value. If SuggestedSetting is used, this setting is ignored.
- `UINT32 T2Level`: 9
Customized T2Level value. If SuggestedSetting is used, this setting is ignored.
- `UINT32 TTEnable`: 1
Enable the thermal throttle function. If SuggestedSetting is used, this settings is ignored.
- `UINT32 TTState13Enable`: 1
When set to 1 and the programmed GPIO pin is a 1, then PMSync state 13 will force at least T2 state.
- `UINT32 TTLock`: 1
When set to 1, this entire register (TL) is locked and remains locked until the next platform reset.
- `UINT32 SuggestedSetting`: 1
*0: Disable; 1: **Enable** suggested representative values.*
- `UINT32 PchCrossThrottling`: 1
ULT processors support thermal management and cross thermal throttling between the processor package and LP PCH.
- `UINT32 Rsvd0`
Reserved bytes.

13.189.1 Detailed Description

This structure lists PCH supported throttling register setting for customization.

When the SuggestedSetting is enabled, the customized values are ignored.

Definition at line 47 of file ThermalConfig.h.

13.189.2 Member Data Documentation

13.189.2.1 PchCrossThrottling

```
UINT32 THERMAL_THROTTLE_LEVELS::PchCrossThrottling
```

ULT processors support thermal management and cross thermal throttling between the processor package and LP PCH.

The PMSYNC message from PCH to CPU includes specific bit fields to update the PCH thermal status to the processor which is factored into the processor throttling. Enable/Disable PCH Cross Throttling; 0: Disabled, 1: **Enabled**.

Definition at line 69 of file ThermalConfig.h.

13.189.2.2 TTLock

```
UINT32 THERMAL_THROTTLE_LEVELS::TTLock
```

When set to 1, this entire register (TL) is locked and remains locked until the next platform reset.

If SuggestedSetting is used, this setting is ignored.

Definition at line 61 of file ThermalConfig.h.

13.189.2.3 TTState13Enable

```
UINT32 THERMAL_THROTTLE_LEVELS::TTState13Enable
```

When set to 1 and the programmed GPIO pin is a 1, then PMSync state 13 will force at least T2 state.

If SuggestedSetting is used, this setting is ignored.

Definition at line 56 of file ThermalConfig.h.

The documentation for this struct was generated from the following file:

- [ThermalConfig.h](#)

13.190 TRACE_HUB_CONFIG Struct Reference

[TRACE_HUB_CONFIG](#) block describes TraceHub settings.

```
#include <TraceHubConfig.h>
```

Public Attributes

- UINT8 [EnableMode](#)
Trace hub mode.
- UINT8 [MemReg0Size](#)
Trace hub memory buffer region size policy.
- UINT8 [AetEnabled](#)
AET Trace.

13.190.1 Detailed Description

[TRACE_HUB_CONFIG](#) block describes TraceHub settings.

Definition at line 78 of file TraceHubConfig.h.

13.190.2 Member Data Documentation

13.190.2.1 AetEnabled

```
UINT8 TRACE_HUB_CONFIG::AetEnabled
```

AET Trace.

AET base address can be set to FW Base either from CPU trace hub or PCH one. AetEnabled must be exclusive, if AetEnabled = 1 for CPU trace hub, must AetEnabled = 0 for PCH one. The default is set to PCH. CPU Trace Hub **0 = Disabled**; 1 = Enabled PCH Trace Hub 0 = Disabled; **1 = Enabled**

Definition at line 104 of file TraceHubConfig.h.

13.190.2.2 EnableMode

```
UINT8 TRACE_HUB_CONFIG::EnableMode
```

Trace hub mode.

Default is disabled. Target Debugger mode refers to debug tool running on target device itself and it works as a conventional PCI device; Host Debugger mode refers to SUT debugged via probe on host, configured as ACPI device with PCI configuration sapce hidden. **0 = Disable**; 1 = Target Debugger mode; 2 = Host Debugger mode Refer to TRACE_HUB_ENABLE_MODE

Definition at line 86 of file TraceHubConfig.h.

13.190.2.3 MemReg0Size

```
UINT8 TRACE_HUB_CONFIG::MemReg0Size
```

Trace hub memory buffer region size policy.

The available memory size options are: 0:0MB (none), 1:1MB, **2:8MB**, 3:64MB, 4:128MB, 5:256MB, 6:512MB. Note : Limitation of total buffer size (CPU + PCH) is 512MB. If iTbt is enabled, the total size limits to 256 MB. Refer to TRACE_BUFFER_SIZE

Definition at line 93 of file TraceHubConfig.h.

The documentation for this struct was generated from the following file:

- [TraceHubConfig.h](#)

13.191 TS_GPIO_PIN_SETTING Struct Reference

This structure configures PCH memory throttling thermal sensor GPIO PIN settings.

```
#include <ThermalConfig.h>
```

Public Attributes

- UINT32 [PmsyncEnable](#): 1
GPIO PM_SYNC enable, 0:Disabled, 1:Enabled When enabled, RC will overrides the selected GPIO native mode.
- UINT32 [C0TransmitEnable](#): 1
GPIO Transmit enable in C0 state, 0:Disabled, 1:Enabled
- UINT32 [PinSelection](#): 1
GPIO Pin assignment selection, 0: default, 1: secondary.

13.191.1 Detailed Description

This structure configures PCH memory throttling thermal sensor GPIO PIN settings.

Definition at line 103 of file ThermalConfig.h.

13.191.2 Member Data Documentation

13.191.2.1 PmsyncEnable

```
UINT32 TS_GPIO_PIN_SETTING::PmsyncEnable
```

GPIO PM_SYNC enable, 0:Disabled, 1:**Enabled** When enabled, RC will overrides the selected GPIO native mode.

For GPIO_C, PinSelection 0: CPU_GP_0 (default) or 1: CPU_GP_1 For GPIO_D, PinSelection 0: CPU_GP_3 (default) or 1: CPU_GP_2 For CNL: CPU_GP_0 is GPP_E3, CPU_GP_1 is GPP_E7, CPU_GP_2 is GPP_B3, CPU_GP_3 is GPP_B4.

Definition at line 111 of file ThermalConfig.h.

The documentation for this struct was generated from the following file:

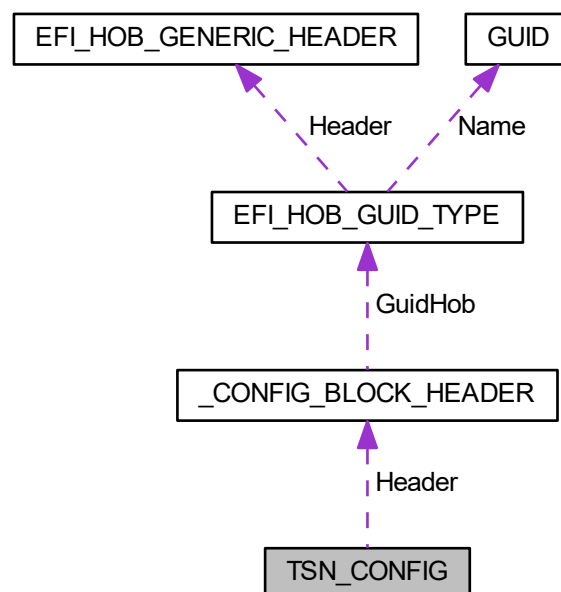
- [ThermalConfig.h](#)

13.192 TSN_CONFIG Struct Reference

The [TSN_CONFIG](#) block describes policies related to Time Sensitive Networking(TSN)

```
#include <TsnConfig.h>
```

Collaboration diagram for TSN_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT8 [Enable](#)
*Determines if enable PCH internal TSN, 0: Disable; 1: **Enable**.*
- UINT8 [TsnLinkSpeed](#)
*Refer to "TSN_LINK_SPEED_OPTIONS". Default is **TSN_LINK_SPEED_2_5G_34MHZ***
- UINT8 [MultiVcEnable](#)
Enable Multi-Vc via Tc to DMA mapping. Deafult is 0:disable
- UINT8 [Rsvd0](#)
Reserved bytes, align to multiple 4.
- UINT32 [TsnMacAddressHigh](#)
MAC Address High bits. Only two lower bytes are used.
- UINT32 [TsnMacAddressLow](#)
MAC Address Low bits.

13.192.1 Detailed Description

The [TSN_CONFIG](#) block describes policies related to Time Sensitive Networking(TSN)

Revision 1:

- Initial version. **Revision 2:**
- Added MultiVcEnable

Definition at line 51 of file TsnConfig.h.

The documentation for this struct was generated from the following file:

- [TsnConfig.h](#)

13.193 UART_PIN_MUX Struct Reference

UART signals pin muxing settings.

```
#include <SerialIoDevices.h>
```

Public Attributes

- UINT32 [Rx](#)
*RXD Pin mux configuration. Refer to GPIO_*_MUXING_SERIALIO_UARTx_RXD_*.*
- UINT32 [Tx](#)
*TXD Pin mux configuration. Refer to GPIO_*_MUXING_SERIALIO_UARTx_TXD_*.*
- UINT32 [Rts](#)
*RTS Pin mux configuration. Refer to GPIO_*_MUXING_SERIALIO_UARTx_RTS_*.*
- UINT32 [Cts](#)
*CTS Pin mux configuration. Refer to GPIO_*_MUXING_SERIALIO_UARTx_CTS_*.*

13.193.1 Detailed Description

UART signals pin muxing settings.

If signal can be enable only on a single pin then this parameter is ignored by RC. Refer to GPIO_*_MUXING_SE↵
RIALIO_UARTx_* in GpioPins*.h for supported settings on a given platform

Definition at line 164 of file SerialIoDevices.h.

The documentation for this struct was generated from the following file:

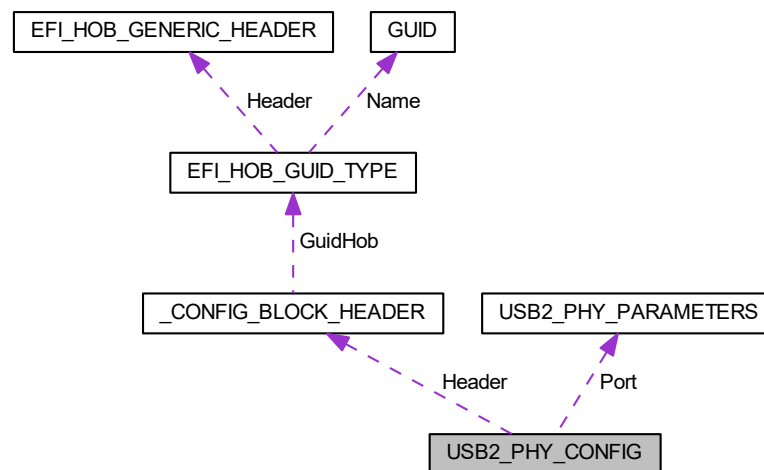
- [SerialIoDevices.h](#)

13.194 USB2_PHY_CONFIG Struct Reference

This structure holds info on how to tune electrical parameters of USB2 ports based on board layout.

```
#include <Usb2PhyConfig.h>
```

Collaboration diagram for USB2_PHY_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [USB2_PHY_PARAMETERS](#) Port [MAX_USB2_PORTS]
This structure configures per USB2 port physical settings.

13.194.1 Detailed Description

This structure holds info on how to tune electrical parameters of USB2 ports based on board layout.

Revision 1:

- Initial version.

Definition at line 95 of file Usb2PhyConfig.h.

13.194.2 Member Data Documentation

13.194.2.1 Port

```
USB2_PHY_PARAMETERS USB2_PHY_CONFIG::Port [MAX_USB2_PORTS]
```

This structure configures per USB2 port physical settings.

It allows to setup the port location and port length, and configures the port strength accordingly. Changing this policy values from default ones may require disabling USB2 PHY Sus Well Power Gating through Usb2PhySusPgEnable on PCH-LP

Definition at line 103 of file Usb2PhyConfig.h.

The documentation for this struct was generated from the following file:

- [Usb2PhyConfig.h](#)

13.195 USB2_PHY_PARAMETERS Struct Reference

This structure configures per USB2 AFE settings.

```
#include <Usb2PhyConfig.h>
```

Public Attributes

- [UINT8 Petxiset](#)
Per Port HS Preemphasis Bias (PERPORTPETXISSET) 000b - 0mV 001b - 11.25mV 010b - 16.9mV 011b - 28.15mV 100b - 28.15mV 101b - 39.35mV 110b - 45mV 111b - 56.3mV.
- [UINT8 Txiset](#)
Per Port HS Transmitter Bias (PERPORTTXISSET) 000b - 0mV 001b - 11.25mV 010b - 16.9mV 011b - 28.15mV 100b - 28.15mV 101b - 39.35mV 110b - 45mV 111b - 56.3mV.
- [UINT8 Predeemp](#)
Per Port HS Transmitter Emphasis (IUSBTXEMPHASISEN) 00b - Emphasis OFF 01b - De-emphasis ON 10b - Pre-emphasis ON 11b - Pre-emphasis & De-emphasis ON.
- [UINT8 Pehalfbit](#)
Per Port Half Bit Pre-emphasis (PERPORTTXPEHALF) 1b - half-bit pre-emphasis 0b - full-bit pre-emphasis.

13.195.1 Detailed Description

This structure configures per USB2 AFE settings.

It allows to setup the port electrical parameters.

Definition at line 49 of file Usb2PhyConfig.h.

The documentation for this struct was generated from the following file:

- [Usb2PhyConfig.h](#)

13.196 USB2_PORT_CONFIG Struct Reference

This structure configures per USB2.0 port settings like enabling and overcurrent protection.

```
#include <UsbConfig.h>
```

Public Attributes

- UINT32 [OverCurrentPin](#): 8
These members describe the specific over current pin number of USB 2.0 Port N.
- UINT32 [Enable](#): 1
*0: Disable; 1: **Enable**.*
- UINT32 [PortResetMessageEnable](#): 1
0: Disable USB2 Port Reset Message; 1: Enable USB2 Port Reset Message
- UINT32 [RsvdBits0](#): 22
Reserved bits.

13.196.1 Detailed Description

This structure configures per USB2.0 port settings like enabling and overcurrent protection.

Definition at line 54 of file UsbConfig.h.

13.196.2 Member Data Documentation

13.196.2.1 OverCurrentPin

```
UINT32 USB2_PORT_CONFIG::OverCurrentPin
```

These members describe the specific over current pin number of USB 2.0 Port N.

It is SW's responsibility to ensure that a given port's bit map is set only for one OC pin Description. USB2 and USB3 on the same combo Port must use the same OC pin.

Definition at line 60 of file UsbConfig.h.

The documentation for this struct was generated from the following file:

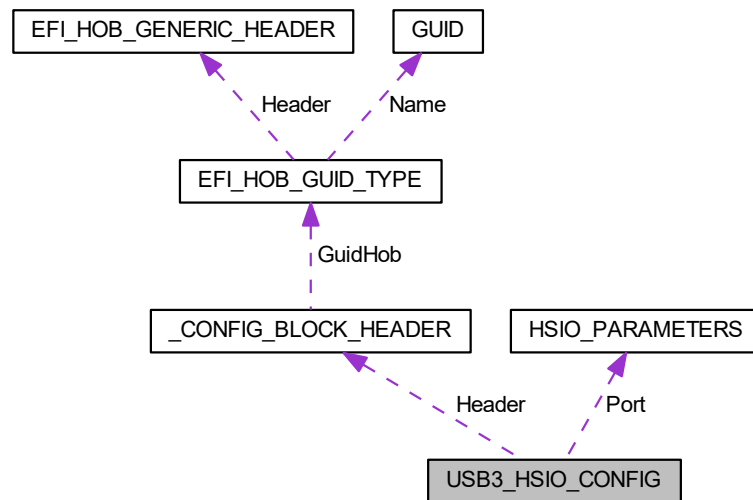
- [UsbConfig.h](#)

13.197 USB3_HSIO_CONFIG Struct Reference

Structure for holding USB3 tuning parameters.

```
#include <Usb3HsioConfig.h>
```

Collaboration diagram for USB3_HSIO_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- [HSIO_PARAMETERS](#) Port [MAX_USB3_PORTS]
These members describe whether the USB3 Port N of PCH is enabled by platform modules.

13.197.1 Detailed Description

Structure for holding USB3 tuning parameters.

Revision 1:

- Initial version. **Revision 2:**
- USB 3.0 TX Output Unique Transition Bit Scale policies added

Definition at line 155 of file Usb3HsioConfig.h.

The documentation for this struct was generated from the following file:

- [Usb3HsioConfig.h](#)

13.198 USB3_PORT_CONFIG Struct Reference

This structure configures per USB3.x port settings like enabling and overcurrent protection.

```
#include <UsbConfig.h>
```

Public Attributes

- UINT32 [OverCurrentPin](#): 8
These members describe the specific over current pin number of USB 3.x Port N.
- UINT32 [Enable](#): 1
*0: Disable; 1: **Enable**.*
- UINT32 [RsvdBits0](#): 23
Reserved bits.

13.198.1 Detailed Description

This structure configures per USB3.x port settings like enabling and overcurrent protection.

Definition at line 69 of file UsbConfig.h.

13.198.2 Member Data Documentation

13.198.2.1 OverCurrentPin

```
UINT32 USB3_PORT_CONFIG::OverCurrentPin
```

These members describe the specific over current pin number of USB 3.x Port N.

It is SW's responsibility to ensure that a given port's bit map is set only for one OC pin Description. USB2 and USB3 on the same combo Port must use the same OC pin.

Definition at line 75 of file UsbConfig.h.

The documentation for this struct was generated from the following file:

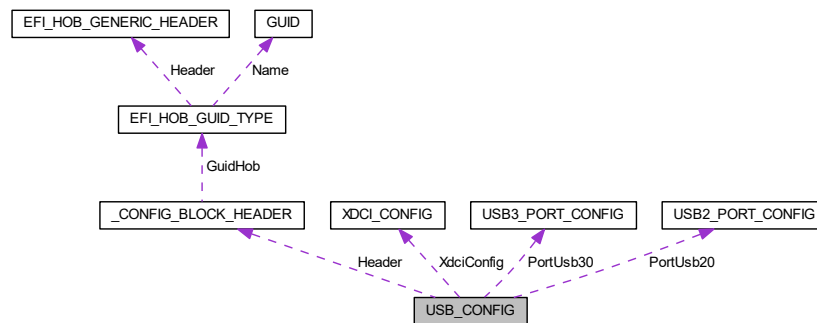
- [UsbConfig.h](#)

13.199 USB_CONFIG Struct Reference

This member describes the expected configuration of the USB controller, Platform modules may need to refer Setup options, schematic, BIOS specification to update this field.

```
#include <UsbConfig.h>
```

Collaboration diagram for USB_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Config Block Header.
- UINT32 [PdoProgramming](#): 1
This policy option when set will make BIOS program Port Disable Override register during PEI phase.
- UINT32 [OverCurrentEnable](#): 1
This option allows for control whether USB should program the Overcurrent Pins mapping into xHCI.
- UINT32 [XhciOcLock](#): 1
(Test) If this policy option is enabled then BIOS will program OCCFDONE bit in xHCI meaning that OC mapping data will be consumed by xHCI and OC mapping registers will be locked.
- UINT32 [LtrOverrideEnable](#): 1
Enabling this feature will allow for overriding LTR values for xHCI controller.
- UINT32 [UaolEnable](#): 1
Enable/disable option for USB Audio Offload feture.
- UINT32 [HsiiEnable](#): 1
*Enable HS Interrupt IN Alarm 0: disable - Disable HSII 1: **enable** - **Enable HSII***
- UINT32 [RsvdBits0](#): 26
Reserved bits.
- UINT32 [LtrHighIdleTimeOverride](#)
High Idle Time Control override value This setting is used only if LtrOverrideEnable is enabled.
- UINT32 [LtrMediumIdleTimeOverride](#)
Medium Idle Time Control override value This setting is used only if LtrOverrideEnable is enabled.
- UINT32 [LtrLowIdleTimeOverride](#)
Low Idle Time Control override value This setting is used only if LtrOverrideEnable is enabled.
- [USB2_PORT_CONFIG](#) [PortUsb20](#) [MAX_USB2_PORTS]
These members describe whether the USB2 Port N of PCH is enabled by platform modules.
- [USB3_PORT_CONFIG](#) [PortUsb30](#) [MAX_USB3_PORTS]
These members describe whether the USB3 Port N of PCH is enabled by platform modules.
- [XDCI_CONFIG](#) [XhciConfig](#)
This member describes whether or not the xDCI controller should be enabled.

13.199.1 Detailed Description

This member describes the expected configuration of the USB controller, Platform modules may need to refer Setup options, schematic, BIOS specification to update this field.

The Usb20OverCurrentPins and Usb30OverCurrentPins field must be updated by referring the schematic.

Revision 1:

- Initial version. **Revision 2:**
- Added UaolEnable to control USB Audio Offload Capability. **Revision 3:**
- Add HsiiEnable enable option to control HSII feature

Definition at line 106 of file UsbConfig.h.

13.199.2 Member Data Documentation

13.199.2.1 LtrOverrideEnable

```
UINT32 USB_CONFIG::LtrOverrideEnable
```

Enabling this feature will allow for overriding LTR values for xHCI controller.

Values used for programming will be taken from this config block and BIOS will disregard recommended ones. **0: disable - do not override recommended LTR values** 1: enable - override recommended LTR values

Definition at line 141 of file UsbConfig.h.

13.199.2.2 OverCurrentEnable

```
UINT32 USB_CONFIG::OverCurrentEnable
```

This option allows for control whether USB should program the Overcurrent Pins mapping into xHCI.

Disabling this feature will disable overcurrent detection functionality. Overcurrent Pin mapping data is contained in respective port structures (i.e. USB30_PORT_CONFIG) in OverCurrentPin field. By default this Overcurrent functionality should be enabled and disabled only for OBS debug usage. **1: Will program USB OC pin mapping in respective xHCI controller registers** 0: Will clear OC pin mapping allow for OBS usage of OC pins

Definition at line 124 of file UsbConfig.h.

13.199.2.3 PdoProgramming

```
UINT32 USB_CONFIG::PdoProgramming
```

This policy option when set will make BIOS program Port Disable Override register during PEI phase.

When disabled BIOS will not program the PDO during PEI phase and leave PDO register unlocked for later programming. If this is disabled, platform code MUST set it before booting into OS. **1: Enable** 0: Disable

Definition at line 115 of file UsbConfig.h.

13.199.2.4 UaolEnable

```
UINT32 USB_CONFIG::UaolEnable
```

Enable/disable option for USB Audio Offload feture.

Disabling this will disable UAOL capability in xHCI and UAOL ACPI definitions will be hidden. 0: disable UAOL **1: enable UAOL**

Definition at line 148 of file UsbConfig.h.

13.199.2.5 XhciOcLock

```
UINT32 USB_CONFIG::XhciOcLock
```

(Test) If this policy option is enabled then BIOS will program OCCFDONE bit in xHCI meaning that OC mapping data will be consumed by xHCI and OC mapping registers will be locked.

OverCurrent mapping data is taken from respective port data structure from OverCurrentPin field. If Enable↔ OverCurrent policy is enabled this also should be enabled, otherwise xHCI won't consume OC mapping data. **1: Program OCCFDONE bit and make xHCI consume OverCurrent mapping data** 0: Do not program OCCFDONE bit making it possible to use OBS debug on OC pins.

Definition at line 134 of file UsbConfig.h.

The documentation for this struct was generated from the following file:

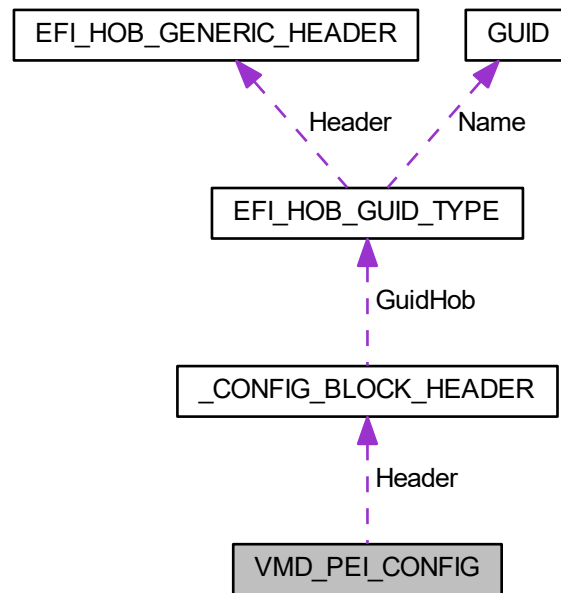
- [UsbConfig.h](#)

13.200 VMD_PEI_CONFIG Struct Reference

This configuration block is to configure VMD related variables used in PostMem PEI.

```
#include <VmdPeiConfig.h>
```

Collaboration diagram for VMD_PEI_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.
- [UINT8 VmdEnable](#)
Offset 28 This field used to enable VMD controller 1=Enable 0=Disable(default)
- [UINT8 VmdPortBEnable](#)
Deprecated < Offset 29 This field used to enable VMD portA Support 1=Enable and 0=Disable (default)
- [UINT8 VmdPortCEnable](#)
Deprecated < Offset 30 This field used to enable VMD portB Support 1=Enable and 0=Disable (default)
- [UINT8 VmdPortDEnable](#)
Deprecated < Offset 31 This field used to enable VMD portC Support 1=Enable and 0=Disable (default)
- [UINT8 VmdCfgBarSize](#)
Deprecated < Offset 32 This field used to enable VMD portD Support 1=Enable and 0=Disable (default)
- [UINT8 VmdCfgBarAttr](#)
Offset 34 This is used to set VMD Config Bar Attributes 0: VMD_32BIT_NONPREFETCH, 1: VMD_64BIT_PREFETCH, 2: VMD_64BIT_NONPREFETCH(Default)
- [UINT8 VmdMemBarSize1](#)
Offset 35 This is used to set the VMD Mem Bar1 size. 25 (32MB).

- [UINT8 VmdMemBar1Attr](#)
Offset 36 This is used to set VMD Mem Bar1 Attributes 0: VMD_32BIT_NONPREFETCH(Default) 1: VMD_64BIT_NONPREFETCH, 2: VMD_64BIT_PREFETCH.
- [UINT8 VmdMemBarSize2](#)
Offset 37 This is used to set the VMD Mem Bar2 size. 20(1MB).
- [UINT8 VmdMemBar2Attr](#)
Offset 38 This is used to set VMD Mem Bar2 Attributes 0: VMD_32BIT_NONPREFETCH 1: VMD_64BIT_NONPREFETCH(Default), 2: VMD_64BIT_PREFETCH.
- [UINT8 VmdGlobalMapping](#)
Offset 39 This field used to enable Global Mapping 1=Enable 0=Disable(default)
- [RP_BDF_DATA VmdPortEnable](#) [VMD_MAX_DEVICES]
Offset 40 to 163 This field used to store b/d/f for each root port along with enable Support 1=Enable 0=Disable(default)
- [UINT32 VmdCfgBarBase](#)
This config block will be updated as per the EFI variable.
- [UINT32 VmdMemBar1Base](#)
Temp Address VMD CFG BAR Default is 0xA0000000
- [UINT32 VmdMemBar2Base](#)
Temp Address VMD CFG BAR Default is 0xA2000000

13.200.1 Detailed Description

This configuration block is to configure VMD related variables used in PostMem PEI.

If VMD Device is not supported, all policies can be ignored. **Revision 1:**

- Initial version. **Revision 2:**
- Deprecated VmdPortAEnable, VmdPortBEnable, VmdPortCEnable, VmdPortDEnable.
- Added VmdPortEnable[VMD_MAX_DEVICES] and structure to hold Vmd EFI Variable details. (Added B/D/F fields along with Port Enable for up to max 31 devices). **Revision 3:** Added policy to get the Bar values from platform PCD. **Revision 4:** Added VmdGlobalMapping to map all the storage devices under VMD

Definition at line 67 of file VmdPeiConfig.h.

13.200.2 Member Data Documentation

13.200.2.1 VmdCfgBarSize

UINT8 VMD_PEI_CONFIG::VmdCfgBarSize

Deprecated < Offset 32 This field used to enable VMD portD Support 1=Enable and 0=Disable (default)

Offset 33 This is used to set the VMD Config Bar Size. 25(32MB)

Definition at line 74 of file VmdPeiConfig.h.

The documentation for this struct was generated from the following file:

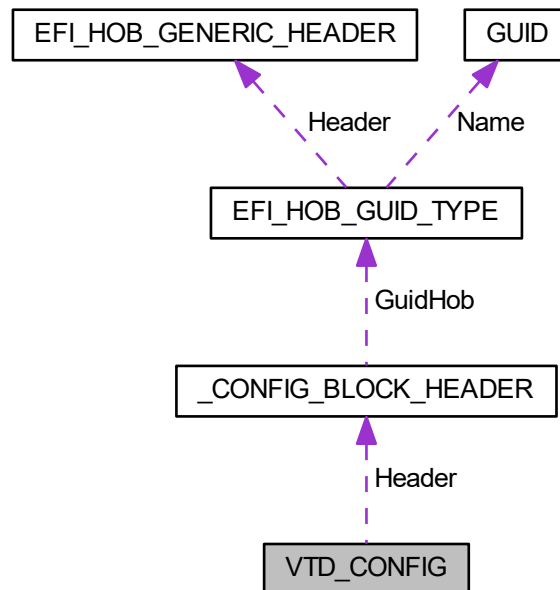
- [VmdPeiConfig.h](#)

13.201 VTD_CONFIG Struct Reference

The data elements should be initialized by a Platform Module.

```
#include <VtdConfig.h>
```

Collaboration diagram for VTD_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER Header](#)
Offset 0-27 Config Block Header.
- [UINT8 VtdDisable](#)
Offset 28: VT-D Support can be verified by reading CAP ID register as explained in BIOS Spec.
- [UINT8 X2ApicOptOut](#)
Offset 29: This field is used to enable the X2APIC_OPT_OUT bit in the DMAR table. 1=Enable/Set and 0=Disable/Clear
- [UINT8 DmaControlGuarantee](#)
Offset 30: This field is used to enable the DMA_CONTROL_GUARANTEE bit in the DMAR table. 1=Enable/Set and 0=Disable/Clear
- [UINT8 VtdIgdEnable](#)
Offset 31: This field is used to enable the VtdIgdEnable Policy. 1=Enable/Set and 0=Disable/Clear
- [UINT8 VtdIpuEnable](#)
Offset 32: This field is used to enable the VtdIpuEnable Policy. 1=Enable/Set and 0=Disable/Clear
- [UINT8 VtdIopEnable](#)
Offset 33: This field is used to enable the VtdIopEnable Policy. 1=Enable/Set and 0=Disable/Clear
- [UINT8 PreBootDmaMask](#)

Deprecate.

- UINT32 [BaseAddress](#) [VTD_ENGINE_NUMBER]
Offset 36: This field is used to describe the base addresses for VT-d function.
- UINT32 [DmaBufferSize](#)
Offset 48: Protect Memory Region (PMR) DMA buffer size.
- UINT32 [Reserved](#)
Offset 52: Reserved.
- [EFI_PHYSICAL_ADDRESS](#) [EnableDmaBuffer](#)
Offset 56: Pointer to EnableDmaBuffer function.

13.201.1 Detailed Description

The data elements should be initialized by a Platform Module.

The data structure is for VT-d driver initialization

Revision 1:

- Initial version. **Revision 2:**
- Added field [EnableDmaBuffer](#).

Definition at line 72 of file [VtdConfig.h](#).

13.201.2 Member Data Documentation

13.201.2.1 [EnableDmaBuffer](#)

[EFI_PHYSICAL_ADDRESS](#) [VTD_CONFIG::EnableDmaBuffer](#)

Offset 56: Pointer to [EnableDmaBuffer](#) function.

Platform code provides an interface to enable DMA buffer. [VTD_ENABLE_DMA_BUFFER](#) is a prototype for that interface.

Definition at line 93 of file [VtdConfig.h](#).

13.201.2.2 [PreBootDmaMask](#)

UINT8 [VTD_CONFIG::PreBootDmaMask](#)

Deprecate.

Offset 35: Convey [PcdVTdPolicyPropertyMask](#) value from [EDK2 IntelSiliconPkg](#)

Definition at line 89 of file [VtdConfig.h](#).

13.201.2.3 VtdDisable

```
UINT8 VTD_CONFIG::VtdDisable
```

Offset 28: VT-D Support can be verified by reading CAP ID register as explained in BIOS Spec.

This policy is for debug purpose only. If VT-D is not supported, all other policies in this config block will be ignored.

0 = To use Vt-d; 1 = Avoids programming Vtd bars, Vtd overrides and DMAR table.

Definition at line 82 of file VtdConfig.h.

The documentation for this struct was generated from the following file:

- [VtdConfig.h](#)

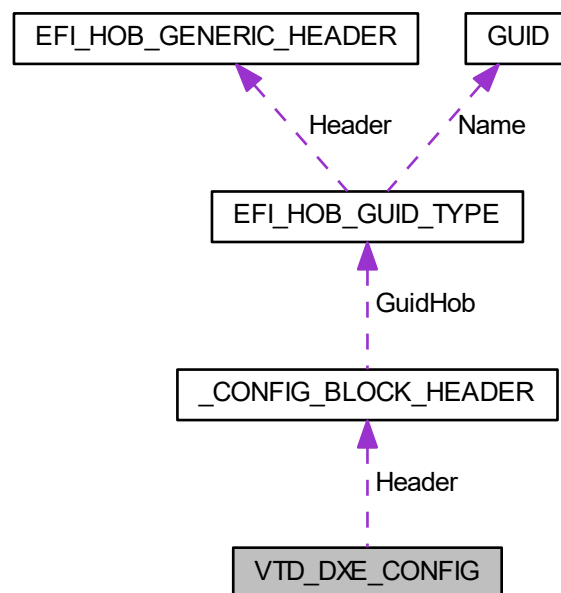
13.202 VTD_DXE_CONFIG Struct Reference

The data structure is for VT-d driver initialization in DXE

Revision 1:

```
#include <VtdConfig.h>
```

Collaboration diagram for VTD_DXE_CONFIG:



Public Attributes

- [CONFIG_BLOCK_HEADER](#) Header
Offset 0-27 Config Block Header.

13.202.1 Detailed Description

The data structure is for VT-d driver initialization in DXE

Revision 1:

- Initial version. **Revision 2:**
- Added TE Disable for Igd. **Revision 3:**
- Removed TE Disable for Igd.

Definition at line 106 of file VtdConfig.h.

The documentation for this struct was generated from the following file:

- [VtdConfig.h](#)

13.203 XDCI_CONFIG Struct Reference

The [XDCI_CONFIG](#) block describes the configurations of the xDCI Usb Device controller.

```
#include <UsbConfig.h>
```

Public Attributes

- UINT32 [Enable](#): 1
This member describes whether or not the xDCI controller should be enabled.
- UINT32 [RsvdBits0](#): 31
Reserved bits.

13.203.1 Detailed Description

The [XDCI_CONFIG](#) block describes the configurations of the xDCI Usb Device controller.

Definition at line 84 of file UsbConfig.h.

13.203.2 Member Data Documentation

13.203.2.1 Enable

```
UINT32 XDCI_CONFIG::Enable
```

This member describes whether or not the xDCI controller should be enabled.

0: Disable; 1: **Enable**.

Definition at line 89 of file UsbConfig.h.

The documentation for this struct was generated from the following file:

- [UsbConfig.h](#)

Chapter 14

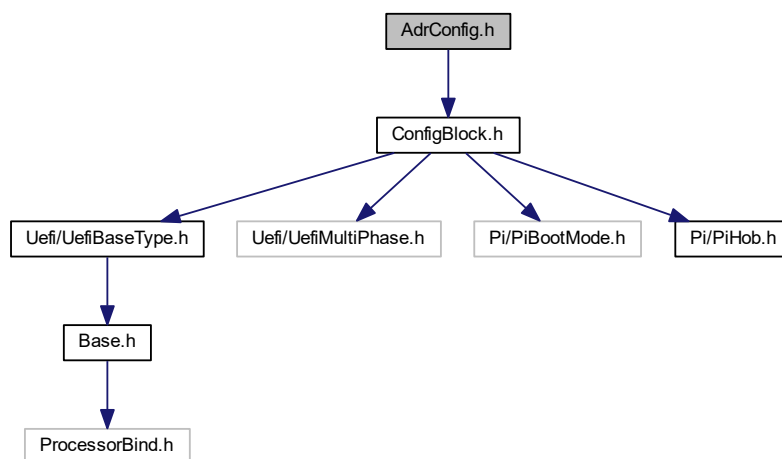
File Documentation

14.1 AdrConfig.h File Reference

ADR policy.

```
#include <ConfigBlock.h>
```

Include dependency graph for AdrConfig.h:



Classes

- union [ADR_SOURCE_ENABLE](#)

ADR Source Enable.

- struct [ADR_CONFIG](#)

*ADR Configuration **Revision 1**: - Initial version.*

14.1.1 Detailed Description

ADR policy.

Copyright

INTEL CONFIDENTIAL Copyright 2019 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains a 'Sample Driver' and is licensed as such under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to the additional terms of the license agreement.

Specification Reference:

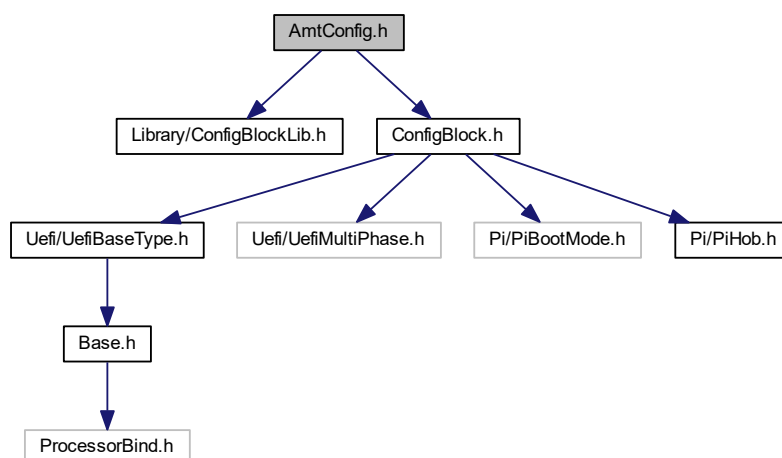
14.2 AmtConfig.h File Reference

AMT Config Block for PEI/DXE phase.

```
#include <Library/ConfigBlockLib.h>
```

```
#include <ConfigBlock.h>
```

Include dependency graph for AmtConfig.h:



Classes

- struct [AMT_PEI_CONFIG](#)
AMT Pei Configuration Structure.
- struct [AMT_DXE_CONFIG](#)
AMT Dxe Configuration Structure.

Typedefs

- typedef [VOID](#)(* [AMT_REPORT_ERROR](#)) (IN [AMT_ERROR_MSG_ID](#) MsgId)
Show AMT Error message.

Enumerations

- enum [AMT_ERROR_MSG_ID](#)
AMT Error Message ID.

14.2.1 Detailed Description

AMT Config Block for PEI/DXE phase.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.2.2 Typedef Documentation

Classes

- struct [GUID](#)
128 bit buffer containing a unique identifier value.
- struct [IPv4_ADDRESS](#)
4-byte buffer.
- struct [IPv6_ADDRESS](#)
16-byte buffer.
- struct [_LIST_ENTRY](#)
[_LIST_ENTRY](#) structure definition.

Macros

- #define [GLOBAL_REMOVE_IF_UNREFERENCED](#)
Remove the global variable from the linked image if there are no references to it after all compiler and linker optimizations have been performed.
- #define [UNREACHABLE\(\)](#)
Signal compilers and analyzers that this call is not reachable.
- #define [NORETURN](#)
Signal compilers and analyzers that the function cannot return.
- #define [ANALYZER_UNREACHABLE\(\)](#)
Signal the analyzer that this call is not reachable.
- #define [ANALYZER_NORETURN](#)
Signal the analyzer that the function cannot return.
- #define [RETURNS_TWICE](#)
Tell the code optimizer that the function will return twice.
- #define [__CONCATENATE\(a, b\)](#) [__CONCATENATE\(a, b\)](#)
Private worker functions for [ASM_PFX\(\)](#)
- #define [ASM_PFX\(name\)](#) [__CONCATENATE](#) ([__USER_LABEL_PREFIX__](#), name)
The [USER_LABEL_PREFIX](#) macro predefined by GNUC represents the prefix on symbols in assembly language.
- #define [CONST](#) const
Datum is read-only.
- #define [STATIC](#) static
Datum is scoped to the current file or function.
- #define [VOID](#) void
Undeclared type.
- #define [IN](#)
Datum is passed to the function.
- #define [OUT](#)
Datum is returned from the function.
- #define [OPTIONAL](#)
Passing the datum to the function is optional, and a NULL is passed if the value is not supplied.
- #define [TRUE](#) ((BOOLEAN)(1==1))
Boolean true value.
- #define [FALSE](#) ((BOOLEAN)(0==1))
Boolean false value.
- #define [NULL](#) ((VOID *) 0)
*NULL pointer (VOID *)*
- #define [MAX_INT8](#) ((INT8)0x7F)
Maximum values for common UEFI Data Types.

- `#define MIN_INT8 (((INT8) -127) - 1)`
Minimum values for the signed UEFI Data Types.
- `#define _INT_SIZE_OF(n) ((sizeof (n) + sizeof (UINTN) - 1) &~(sizeof (UINTN) - 1))`
Return the size of argument that has been aligned to sizeof (UINTN).
- `#define VA_START(Marker, Parameter) (Marker = (VA_LIST) ((UINTN) & (Parameter) + _INT_SIZE_OF (Parameter)))`
Retrieves a pointer to the beginning of a variable argument list, based on the name of the parameter that immediately precedes the variable argument list.
- `#define VA_ARG(Marker, TYPE) (*(TYPE *) ((Marker += _INT_SIZE_OF (TYPE)) - _INT_SIZE_OF (TYPE)))`
Returns an argument of a specified type from a variable argument list and updates the pointer to the variable argument list to point to the next argument.
- `#define VA_END(Marker) (Marker = (VA_LIST) 0)`
Terminates the use of a variable argument list.
- `#define VA_COPY(Dest, Start) ((void)((Dest) = (Start)))`
Initializes a VA_LIST as a copy of an existing VA_LIST.
- `#define _BASE_INT_SIZE_OF(TYPE) ((sizeof (TYPE) + sizeof (UINTN) - 1) / sizeof (UINTN))`
Returns the size of a data type in sizeof(UINTN) units rounded up to the nearest UINTN boundary.
- `#define BASE_ARG(Marker, TYPE) (*(TYPE *) ((Marker += _BASE_INT_SIZE_OF (TYPE)) - _BASE_INT_SIZE_OF (TYPE)))`
Returns an argument of a specified type from a variable argument list and updates the pointer to the variable argument list to point to the next argument.
- `#define OFFSET_OF(TYPE, Field) ((UINTN) &(((TYPE *)0)->Field))`
The macro that returns the byte offset of a field in a data structure.
- `#define STATIC_ASSERT _Static_assert`
Portable definition for compile time assertions.
- `#define BASE_CR(Record, TYPE, Field) ((TYPE *) ((CHAR8 *) (Record) - OFFSET_OF (TYPE, Field)))`
Macro that returns a pointer to the data structure that contains a specified field of that data structure.
- `#define ALIGN_VALUE(Value, Alignment) ((Value) + (((Alignment) - (Value)) & ((Alignment) - 1)))`
Rounds a value up to the next boundary using a specified alignment.
- `#define ALIGN_POINTER(Pointer, Alignment) ((VOID *) (ALIGN_VALUE ((UINTN)(Pointer), (Alignment))))`
Adjust a pointer by adding the minimum offset required for it to be aligned on a specified alignment boundary.
- `#define ALIGN_VARIABLE(Value) ALIGN_VALUE ((Value), sizeof (UINTN))`
Rounds a value up to the next natural boundary for the current CPU.
- `#define MAX(a, b) (((a) > (b)) ? (a) : (b))`
Return the maximum of two operands.
- `#define MIN(a, b) (((a) < (b)) ? (a) : (b))`
Return the minimum of two operands.
- `#define ABS(a) (((a) < 0) ? -(a) : (a))`
Return the absolute value of a signed operand.
- `#define ENCODE_ERROR(StatusCode) ((RETURN_STATUS)(MAX_BIT | (StatusCode)))`
Produces a RETURN_STATUS code with the highest bit set.
- `#define ENCODE_WARNING(StatusCode) ((RETURN_STATUS)(StatusCode))`
Produces a RETURN_STATUS code with the highest bit clear.
- `#define RETURN_ERROR(StatusCode) (((INTN)(RETURN_STATUS)(StatusCode)) < 0)`
Returns TRUE if a specified RETURN_STATUS code is an error code.
- `#define RETURN_SUCCESS 0`
The operation completed successfully.
- `#define RETURN_LOAD_ERROR ENCODE_ERROR (1)`
The image failed to load.
- `#define RETURN_INVALID_PARAMETER ENCODE_ERROR (2)`
The parameter was incorrect.

- #define [RETURN_UNSUPPORTED ENCODE_ERROR](#) (3)
The operation is not supported.
- #define [RETURN_BAD_BUFFER_SIZE ENCODE_ERROR](#) (4)
The buffer was not the proper size for the request.
- #define [RETURN_BUFFER_TOO_SMALL ENCODE_ERROR](#) (5)
The buffer was not large enough to hold the requested data.
- #define [RETURN_NOT_READY ENCODE_ERROR](#) (6)
There is no data pending upon return.
- #define [RETURN_DEVICE_ERROR ENCODE_ERROR](#) (7)
The physical device reported an error while attempting the operation.
- #define [RETURN_WRITE_PROTECTED ENCODE_ERROR](#) (8)
The device can not be written to.
- #define [RETURN_OUT_OF_RESOURCES ENCODE_ERROR](#) (9)
The resource has run out.
- #define [RETURN_VOLUME_CORRUPTED ENCODE_ERROR](#) (10)
An inconsistency was detected on the file system causing the operation to fail.
- #define [RETURN_VOLUME_FULL ENCODE_ERROR](#) (11)
There is no more space on the file system.
- #define [RETURN_NO_MEDIA ENCODE_ERROR](#) (12)
The device does not contain any medium to perform the operation.
- #define [RETURN_MEDIA_CHANGED ENCODE_ERROR](#) (13)
The medium in the device has changed since the last access.
- #define [RETURN_NOT_FOUND ENCODE_ERROR](#) (14)
The item was not found.
- #define [RETURN_ACCESS_DENIED ENCODE_ERROR](#) (15)
Access was denied.
- #define [RETURN_NO_RESPONSE ENCODE_ERROR](#) (16)
The server was not found or did not respond to the request.
- #define [RETURN_NO_MAPPING ENCODE_ERROR](#) (17)
A mapping to the device does not exist.
- #define [RETURN_TIMEOUT ENCODE_ERROR](#) (18)
A timeout time expired.
- #define [RETURN_NOT_STARTED ENCODE_ERROR](#) (19)
The protocol has not been started.
- #define [RETURN_ALREADY_STARTED ENCODE_ERROR](#) (20)
The protocol has already been started.
- #define [RETURN_ABORTED ENCODE_ERROR](#) (21)
The operation was aborted.
- #define [RETURN_ICMP_ERROR ENCODE_ERROR](#) (22)
An ICMP error occurred during the network operation.
- #define [RETURN_TFTP_ERROR ENCODE_ERROR](#) (23)
A TFTP error occurred during the network operation.
- #define [RETURN_PROTOCOL_ERROR ENCODE_ERROR](#) (24)
A protocol error occurred during the network operation.
- #define [RETURN_INCOMPATIBLE_VERSION ENCODE_ERROR](#) (25)
A function encountered an internal version that was incompatible with a version requested by the caller.
- #define [RETURN_SECURITY_VIOLATION ENCODE_ERROR](#) (26)
The function was not performed due to a security violation.
- #define [RETURN_CRC_ERROR ENCODE_ERROR](#) (27)
A CRC error was detected.
- #define [RETURN_END_OF_MEDIA ENCODE_ERROR](#) (28)

- The beginning or end of media was reached.*

 - #define `RETURN_END_OF_FILE_ENCODE_ERROR` (31)
- The end of the file was reached.*

 - #define `RETURN_INVALID_LANGUAGE_ENCODE_ERROR` (32)
- The language specified was invalid.*

 - #define `RETURN_COMPROMISED_DATA_ENCODE_ERROR` (33)
- The security status of the data is unknown or compromised and the data must be updated or replaced to restore a valid security status.*

 - #define `RETURN_HTTP_ERROR_ENCODE_ERROR` (35)
- A HTTP error occurred during the network operation.*

 - #define `RETURN_WARN_UNKNOWN_GLYPH_ENCODE_WARNING` (1)
- The string contained one or more characters that the device could not render and were skipped.*

 - #define `RETURN_WARN_DELETE_FAILURE_ENCODE_WARNING` (2)
- The handle was closed, but the file was not deleted.*

 - #define `RETURN_WARN_WRITE_FAILURE_ENCODE_WARNING` (3)
- The handle was closed, but the data to the file was not flushed properly.*

 - #define `RETURN_WARN_BUFFER_TOO_SMALL_ENCODE_WARNING` (4)
- The resulting buffer was too small, and the data was truncated to the buffer size.*

 - #define `RETURN_WARN_STALE_DATA_ENCODE_WARNING` (5)
- The data has not been updated within the timeframe set by local policy for this type of data.*

 - #define `RETURN_WARN_FILE_SYSTEM_ENCODE_WARNING` (6)
- The resulting buffer contains UEFI-compliant file system.*

 - #define `SIGNATURE_16(A, B) ((A) | (B << 8))`
- Returns a 16-bit signature built from 2 ASCII characters.*

 - #define `SIGNATURE_32(A, B, C, D) (SIGNATURE_16(A, B) | (SIGNATURE_16(C, D) << 16))`
- Returns a 32-bit signature built from 4 ASCII characters.*

 - #define `SIGNATURE_64(A, B, C, D, E, F, G, H) (SIGNATURE_32(A, B, C, D) | ((UINT64) (SIGNATURE_32(E, F, G, H)) << 32))`
- Returns a 64-bit signature built from 8 ASCII characters.*

 - #define `RETURN_ADDRESS(L) ((VOID *) 0)`
- Get the return address of the calling function.*

 - #define `ARRAY_SIZE(Array) (sizeof (Array) / sizeof ((Array)[0]))`
- Return the number of elements in an array.*

Typedefs

- typedef struct `_LIST_ENTRY` `LIST_ENTRY`
- LIST_ENTRY structure definition.*
- typedef `CHAR8 *` `VA_LIST`
- Variable used to traverse the list of arguments.*
- typedef `UINTN *` `BASE_LIST`
- Pointer to the start of a variable argument list stored in a memory buffer.*

14.3.1 Detailed Description

Root include file for Mde Package Base type modules.

This is the include file for any module of type base. Base modules only use types defined via this include file and can be ported easily to any environment. There are a set of base libraries in the Mde Package that can be used to implement base modules.

Copyright (c) 2006 - 2021, Intel Corporation. All rights reserved.
 Portions copyright (c) 2008 - 2009, Apple Inc. All rights reserved.
 SPDX-License-Identifier: BSD-2-Clause-Patent

14.3.2 Macro Definition Documentation

14.3.2.1 `_BASE_INT_SIZE_OF`

```
#define _BASE_INT_SIZE_OF(  
    TYPE ) ((sizeof (TYPE) + sizeof (UINTN) - 1) / sizeof (UINTN))
```

Returns the size of a data type in sizeof(UINTN) units rounded up to the nearest UINTN boundary.

Parameters

<code>TYPE</code>	The data type to determine the size of.
-------------------	---

Returns

The size of TYPE in sizeof (UINTN) units rounded up to the nearest UINTN boundary.

Definition at line 752 of file Base.h.

14.3.2.2 `_INT_SIZE_OF`

```
#define _INT_SIZE_OF(  
    n ) ((sizeof (n) + sizeof (UINTN) - 1) &~(sizeof (UINTN) - 1))
```

Return the size of argument that has been aligned to sizeof (UINTN).

Parameters

<code>n</code>	The parameter size to be aligned.
----------------	-----------------------------------

Returns

The aligned size.

Definition at line 579 of file Base.h.

14.3.2.3 `ABS`

```
#define ABS(  
    a ) ((a) < 0) ? (-a) : (a)
```

Return the absolute value of a signed operand.

This macro returns the absolute value of the signed operand specified by a.

Parameters

<i>a</i>	The signed operand.
----------	---------------------

Returns

The absolute value of the signed operand.

Definition at line 956 of file Base.h.

14.3.2.4 ALIGN_POINTER

```
#define ALIGN_POINTER(  
    Pointer,  
    Alignment ) ((VOID *) (ALIGN_VALUE ((UINTN) (Pointer), (Alignment))))
```

Adjust a pointer by adding the minimum offset required for it to be aligned on a specified alignment boundary.

This function rounds the pointer specified by *Pointer* to the next alignment boundary specified by *Alignment*. The pointer to the aligned address is returned.

Parameters

<i>Pointer</i>	The pointer to round up.
<i>Alignment</i>	The alignment boundary to use to return an aligned pointer.

Returns

Pointer to the aligned address.

Definition at line 899 of file Base.h.

14.3.2.5 ALIGN_VALUE

```
#define ALIGN_VALUE(  
    Value,  
    Alignment ) ((Value) + (((Alignment) - (Value)) & ((Alignment) - 1)))
```

Rounds a value up to the next boundary using a specified alignment.

This function rounds *Value* up to the next boundary using the specified *Alignment*. This aligned value is returned.

Parameters

<i>Value</i>	The value to round up.
<i>Alignment</i>	The alignment boundary used to return the aligned value.

Returns

A value up to the next boundary.

Definition at line 884 of file Base.h.

14.3.2.6 ALIGN_VARIABLE

```
#define ALIGN_VARIABLE(  
    Value ) ALIGN_VALUE ((Value), sizeof (UINTN))
```

Rounds a value up to the next natural boundary for the current CPU.

This is 4-bytes for 32-bit CPUs and 8-bytes for 64-bit CPUs.

This function rounds the value specified by Value up to the next natural boundary for the current CPU. This rounded value is returned.

Parameters

<i>Value</i>	The value to round up.
--------------	------------------------

Returns

Rounded value specified by Value.

Definition at line 913 of file Base.h.

14.3.2.7 ANALYZER_NORETURN

```
#define ANALYZER_NORETURN
```

Signal the analyzer that the function cannot return.

This excludes compilers.

Definition at line 157 of file Base.h.

14.3.2.8 ANALYZER_UNREACHABLE

```
#define ANALYZER_UNREACHABLE( )
```

Signal the analyzer that this call is not reachable.

This excludes compilers.

Definition at line 131 of file Base.h.

14.3.2.9 ARRAY_SIZE

```
#define ARRAY_SIZE(  
    Array ) (sizeof (Array) / sizeof ((Array)[0]))
```

Return the number of elements in an array.

Parameters

<i>Array</i>	An object of array type. Array is only used as an argument to the sizeof operator, therefore Array is never evaluated. The caller is responsible for ensuring that Array's type is not incomplete; that is, Array must have known constant size.
--------------	--

Returns

The number of elements in Array. The result has type UINTN.

Definition at line 1319 of file Base.h.

14.3.2.10 BASE_ARG

```
#define BASE_ARG(  
    Marker,  
    TYPE ) (*(TYPE *) ((Marker += _BASE_INT_SIZE_OF (TYPE)) - _BASE_INT_SIZE_OF (TYPE)↵  
PE))
```

Returns an argument of a specified type from a variable argument list and updates the pointer to the variable argument list to point to the next argument.

This function returns an argument of the type specified by TYPE from the beginning of the variable argument list specified by Marker. Marker is then updated to point to the next argument in the variable argument list. The method for computing the pointer to the next argument in the argument list is CPU specific following the EFIABI ABI.

Parameters

<i>Marker</i>	The pointer to the beginning of a variable argument list.
<i>TYPE</i>	The type of argument to retrieve from the beginning of the variable argument list.

Returns

An argument of the type specified by TYPE.

Definition at line 770 of file Base.h.

14.3.2.11 BASE_CR

```
#define BASE_CR(  
    Record,  
    TYPE,  
    Field ) ((TYPE *) ((CHAR8 *) (Record) - OFFSET_OF (TYPE, Field)))
```

Macro that returns a pointer to the data structure that contains a specified field of that data structure.

This is a lightweight method to hide information by placing a public data structure inside a larger private data structure and using a pointer to the public data structure to retrieve a pointer to the private data structure.

This function computes the offset, in bytes, of field specified by *Field* from the beginning of the data structure specified by *TYPE*. This offset is subtracted from *Record*, and is used to return a pointer to a data structure of the type specified by *TYPE*. If the data type specified by *TYPE* does not contain the field specified by *Field*, then the module will not compile.

Parameters

<i>Record</i>	Pointer to the field specified by <i>Field</i> within a data structure of type <i>TYPE</i> .
<i>TYPE</i>	The name of the data structure type to return. This data structure must contain the field specified by <i>Field</i> .
<i>Field</i>	The name of the field in the data structure specified by <i>TYPE</i> to which <i>Record</i> points.

Returns

A pointer to the structure from one of it's elements.

Definition at line 870 of file Base.h.

14.3.2.12 ENCODE_ERROR

```
#define ENCODE_ERROR(  
    StatusCode ) ((RETURN_STATUS) (MAX_BIT | (StatusCode)))
```

Produces a RETURN_STATUS code with the highest bit set.

Parameters

<i>StatusCode</i>	The status code value to convert into a warning code. <i>StatusCode</i> must be in the range 0x00000000..0x7FFFFFFF.
-------------------	--

Returns

The value specified by *StatusCode* with the highest bit set.

Definition at line 973 of file Base.h.

14.3.2.13 ENCODE_WARNING

```
#define ENCODE_WARNING(  
    StatusCode ) ((RETURN_STATUS) (StatusCode))
```

Produces a RETURN_STATUS code with the highest bit clear.

Parameters

<i>StatusCode</i>	The status code value to convert into a warning code. StatusCode must be in the range 0x00000000..0x7FFFFFFF.
-------------------	---

Returns

The value specified by StatusCode with the highest bit clear.

Definition at line 984 of file Base.h.

14.3.2.14 FALSE

```
#define FALSE ((BOOLEAN) (0==1))
```

Boolean false value.

UEFI Specification defines this value to be 0, but this form is more portable.

Definition at line 315 of file Base.h.

14.3.2.15 MAX

```
#define MAX(  
    a,  
    b ) (((a) > (b)) ? (a) : (b))
```

Return the maximum of two operands.

This macro returns the maximum of two operand specified by a and b. Both a and b must be the same numerical types, signed or unsigned.

Parameters

<i>a</i>	The first operand with any numerical type.
<i>b</i>	The second operand. Can be any numerical type as long as is the same type as a.

Returns

Maximum of two operands.

Definition at line 928 of file Base.h.

14.3.2.16 MIN

```
#define MIN(  
    a,  
    b )  ((a) < (b)) ?  (a) :  (b)
```

Return the minimum of two operands.

This macro returns the minimal of two operand specified by a and b. Both a and b must be the same numerical types, signed or unsigned.

Parameters

<i>a</i>	The first operand with any numerical type.
<i>b</i>	The second operand. It should be the same any numerical type with a.

Returns

Minimum of two operands.

Definition at line 943 of file Base.h.

14.3.2.17 NORETURN

```
#define NORETURN
```

Signal compilers and analyzers that the function cannot return.

It is up to the compiler to remove any code past a call to functions flagged with this attribute.

Definition at line 107 of file Base.h.

14.3.2.18 OFFSET_OF

```
#define OFFSET_OF(  
    TYPE,  
    Field )  ((UINTN) &(((TYPE *)0)->Field))
```

The macro that returns the byte offset of a field in a data structure.

This function returns the offset, in bytes, of field specified by Field from the beginning of the data structure specified by TYPE. If TYPE does not contain Field, the module will not compile.

Parameters

<i>TYPE</i>	The name of the data structure that contains the field specified by Field.
<i>Field</i>	The name of the field in the data structure.

Returns

Offset, in bytes, of field.

Definition at line 790 of file Base.h.

14.3.2.19 RETURN_ADDRESS

```
#define RETURN_ADDRESS(  
    L ) ((VOID *) 0)
```

Get the return address of the calling function.

Parameters

<i>L</i>	Return Level.
----------	---------------

Returns

0 as compilers don't support this feature.

Definition at line 1305 of file Base.h.

14.3.2.20 RETURN_BUFFER_TOO_SMALL

```
#define RETURN_BUFFER_TOO_SMALL ENCODE_ERROR (5)
```

The buffer was not large enough to hold the requested data.

The required buffer size is returned in the appropriate parameter when this error occurs.

Definition at line 1029 of file Base.h.

14.3.2.21 RETURN_ERROR

```
#define RETURN_ERROR(  
    StatusCode ) (((INTN) (RETURN_STATUS) (StatusCode)) < 0)
```

Returns TRUE if a specified RETURN_STATUS code is an error code.

This function returns TRUE if StatusCode has the high bit set. Otherwise, FALSE is returned.

Parameters

<i>StatusCode</i>	The status code value to evaluate.
-------------------	------------------------------------

Return values

<i>TRUE</i>	The high bit of StatusCode is set.
<i>FALSE</i>	The high bit of StatusCode is clear.

Definition at line 997 of file Base.h.

14.3.2.22 RETURNS_TWICE

```
#define RETURNS_TWICE
```

Tell the code optimizer that the function will return twice.

This prevents wrong optimizations which can cause bugs. Tell the code optimizer that the function will return twice.
This prevents wrong optimizations which can cause bugs.

Definition at line 177 of file Base.h.

14.3.2.23 SIGNATURE_16

```
#define SIGNATURE_16(  
    A,  
    B ) ((A) | (B << 8))
```

Returns a 16-bit signature built from 2 ASCII characters.

This macro returns a 16-bit value built from the two ASCII characters specified by A and B.

Parameters

<i>A</i>	The first ASCII character.
<i>B</i>	The second ASCII character.

Returns

A 16-bit value built from the two ASCII characters specified by A and B.

Definition at line 1219 of file Base.h.

14.3.2.24 SIGNATURE_32

```
#define SIGNATURE_32(  
    A,  
    B,  
    C,  
    D ) (SIGNATURE_16 (A, B) | (SIGNATURE_16 (C, D) << 16))
```

Returns a 32-bit signature built from 4 ASCII characters.

This macro returns a 32-bit value built from the four ASCII characters specified by A, B, C, and D.

Parameters

<i>A</i>	The first ASCII character.
<i>B</i>	The second ASCII character.
<i>C</i>	The third ASCII character.
<i>D</i>	The fourth ASCII character.

Returns

A 32-bit value built from the two ASCII characters specified by A, B, C and D.

Definition at line 1236 of file Base.h.

14.3.2.25 SIGNATURE_64

```
#define SIGNATURE_64(  
    A,  
    B,  
    C,  
    D,  
    E,  
    F,  
    G,  
    H ) (SIGNATURE_32 (A, B, C, D) | ((UINT64) (SIGNATURE_32 (E, F, G, H)) << 32))
```

Returns a 64-bit signature built from 8 ASCII characters.

This macro returns a 64-bit value built from the eight ASCII characters specified by A, B, C, D, E, F, G, and H.

Parameters

<i>A</i>	The first ASCII character.
<i>B</i>	The second ASCII character.
<i>C</i>	The third ASCII character.
<i>D</i>	The fourth ASCII character.
<i>E</i>	The fifth ASCII character.
<i>F</i>	The sixth ASCII character.
<i>G</i>	The seventh ASCII character.
<i>H</i>	The eighth ASCII character.

Returns

A 64-bit value built from the two ASCII characters specified by A, B, C, D, E, F, G and H.

Definition at line 1257 of file Base.h.

14.3.2.26 STATIC_ASSERT

```
#define STATIC_ASSERT _Static_assert
```

Portable definition for compile time assertions.

Equivalent to C11 `static_assert` macro from `assert.h`.

Parameters

<i>Expression</i>	Boolean expression.
<i>Message</i>	Raised compiler diagnostic message when expression is false.

Definition at line 806 of file Base.h.

14.3.2.27 TRUE

```
#define TRUE ((BOOLEAN) (1==1))
```

Boolean true value.

UEFI Specification defines this value to be 1, but this form is more portable.

Definition at line 309 of file Base.h.

14.3.2.28 UNREACHABLE

```
#define UNREACHABLE( )
```

Signal compilers and analyzers that this call is not reachable.

It is up to the compiler to remove any code past that point.

Definition at line 77 of file Base.h.

14.3.2.29 VA_ARG

```
#define VA_ARG(  
    Marker,  
    TYPE ) (*(TYPE *) ((Marker += _INT_SIZE_OF (TYPE)) - _INT_SIZE_OF (TYPE)))
```

Returns an argument of a specified type from a variable argument list and updates the pointer to the variable argument list to point to the next argument.

This function returns an argument of the type specified by TYPE from the beginning of the variable argument list specified by Marker. Marker is then updated to point to the next argument in the variable argument list. The method for computing the pointer to the next argument in the argument list is CPU-specific following the EFIABI ABI.

Parameters

<i>Marker</i>	VA_LIST used to traverse the list of arguments.
<i>TYPE</i>	The type of argument to retrieve from the beginning of the variable argument list.

Returns

An argument of the type specified by TYPE.

Definition at line 711 of file Base.h.

14.3.2.30 VA_COPY

```
#define VA_COPY(  
    Dest,  
    Start ) ((void)((Dest) = (Start)))
```

Initializes a VA_LIST as a copy of an existing VA_LIST.

This macro initializes Dest as a copy of Start, as if the VA_START macro had been applied to Dest followed by the same sequence of uses of the VA_ARG macro as had previously been used to reach the present state of Start.

Parameters

<i>Dest</i>	VA_LIST used to traverse the list of arguments.
<i>Start</i>	VA_LIST used to traverse the list of arguments.

Definition at line 736 of file Base.h.

14.3.2.31 VA_END

```
#define VA_END(  
    Marker ) (Marker = (VA_LIST) 0)
```

Terminates the use of a variable argument list.

This function initializes Marker so it can no longer be used with [VA_ARG\(\)](#). After this macro is used, the only way to access the variable argument list is by using [VA_START\(\)](#) again.

Parameters

<i>Marker</i>	VA_LIST used to traverse the list of arguments.
---------------	---

Definition at line 723 of file Base.h.

14.3.2.32 VA_START

```
#define VA_START(  
    Marker,  
    Parameter ) (Marker = (VA_LIST) ((UINTN) & (Parameter) + __INT_SIZE_OF (Parameter)))
```

Retrieves a pointer to the beginning of a variable argument list, based on the name of the parameter that immediately precedes the variable argument list.

This function initializes Marker to point to the beginning of the variable argument list that immediately follows Parameter. The method for computing the pointer to the next argument in the argument list is CPU-specific following the EFI API ABI.

Parameters

<i>Marker</i>	The VA_LIST used to traverse the list of arguments.
<i>Parameter</i>	The name of the parameter that immediately precedes the variable argument list.

Returns

A pointer to the beginning of a variable argument list.

Definition at line 693 of file Base.h.

14.3.3 Typedef Documentation

14.3.3.1 BASE_LIST

```
typedef UINTN* BASE_LIST
```

Pointer to the start of a variable argument list stored in a memory buffer.

Same as UINT8 *.

Definition at line 743 of file Base.h.

14.3.3.2 VA_LIST

```
typedef CHAR8* VA_LIST
```

Variable used to traverse the list of arguments.

This type can vary by implementation and could be an array or structure.

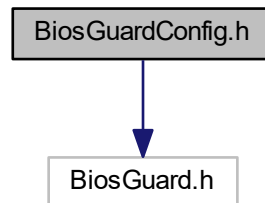
Definition at line 675 of file Base.h.

14.4 BiosGuardConfig.h File Reference

CPU BIOS Guard Config Block.

```
#include <BiosGuard.h>
```

Include dependency graph for BiosGuardConfig.h:



Classes

- struct [BIOS_GUARD_CONFIG](#)
BIOS Guard Configuration Structure.

Macros

- #define [BIOS_GUARD_CONFIG_REVISION](#) 1
BIOS Guard Configuration Revision.

Typedefs

- typedef [EFI_STATUS](#)(* [PLATFORM_SEND_EC_COMMAND](#)) ([IN](#) [EC_COMMAND_TYPE](#) EcCmdType, [IN](#) [UINT8](#) EcCmd, [IN](#) [UINT8](#) SendData, [IN OUT](#) [UINT8](#) *ReceiveData)
This function is for platform code to provide EC Commands since different BIOS might have different EC.

Enumerations

- enum [EC_COMMAND_TYPE](#)
Enums for EC Command Type.

Variables

- [EFI_GUID](#) gBiosGuardConfigGuid
BIOS Guard Configuration [GUID](#).

14.4.1 Detailed Description

CPU BIOS Guard Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2019 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.4.2 Typedef Documentation

14.4.2.1 PLATFORM_SEND_EC_COMMAND

```
typedef EFI_STATUS( * PLATFORM_SEND_EC_COMMAND)(IN EC_COMMAND_TYPE EcCmdType, IN UINT8 EcCmd,
IN UINT8 SendData, IN OUT UINT8 *ReceiveData)
```

This function is for platform code to provide EC Commands since different BIOS might have different EC.

Platform code need to provide a function for CPU code to call to communicate with EC.

Parameters

in	<i>EcCmdType</i>	- EC Command Type.
in	<i>EcCmd</i>	- EC Command Byte to send.
in	<i>SendData</i>	- EC Data Byte to send.
in	<i>ReceiveData</i>	- EC Data Byte received.

Return values

<i>EFI_SUCCESS</i>	Command Read/ Write Success.
<i>EFI_DEVICE_ERROR</i>	Command Read/ Write Error.
<i>EFI_OUT_OF_RESOURCES</i>	No enough resources (such as out of memory).

Definition at line 93 of file BiosGuardConfig.h.

14.5 CnviConfig.h File Reference

CNVi policy.

Classes

- struct [CNVI_PIN_MUX](#)
CNVi signals pin muxing settings.
- struct [CNVI_CONFIG](#)
The [CNVI_CONFIG](#) block describes the expected configuration of the CNVi IP.
- struct [CNVI_PREMEM_CONFIG](#)
The [CNVI_PREMEM_CONFIG](#) block describes the expected configuration of the CNVi IP.

Enumerations

- enum [CNVI_MODE](#)
CNVi Mode options.

14.5.1 Detailed Description

CNVi policy.

Copyright

INTEL CONFIDENTIAL Copyright 2016 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

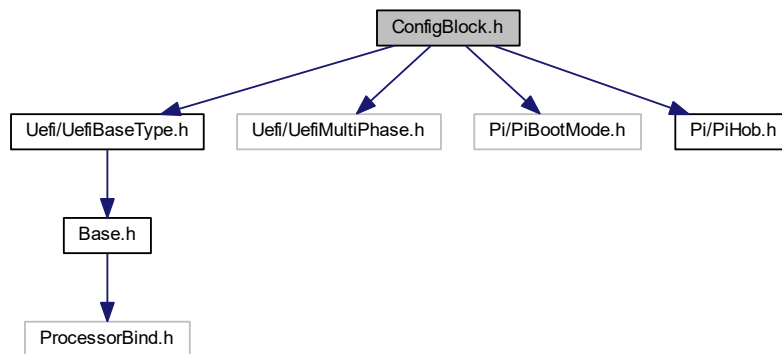
Specification Reference:

14.6 ConfigBlock.h File Reference

Header file for Config Block Lib implementation.

```
#include <Uefi/UefiBaseType.h>
#include <Uefi/UefiMultiPhase.h>
#include <Pi/PiBootMode.h>
#include <Pi/PiHob.h>
```

Include dependency graph for ConfigBlock.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [_CONFIG_BLOCK_HEADER](#)
Config Block Header.
- struct [_CONFIG_BLOCK](#)
Config Block.
- struct [_CONFIG_BLOCK_TABLE_STRUCT](#)
Config Block Table Header.

Typedefs

- typedef struct [_CONFIG_BLOCK_HEADER](#) [CONFIG_BLOCK_HEADER](#)
Config Block Header.
- typedef struct [_CONFIG_BLOCK](#) [CONFIG_BLOCK](#)
Config Block.
- typedef struct [_CONFIG_BLOCK_TABLE_STRUCT](#) [CONFIG_BLOCK_TABLE_HEADER](#)
Config Block Table Header.

14.6.1 Detailed Description

Header file for Config Block Lib implementation.

Copyright

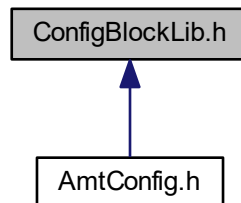
INTEL CONFIDENTIAL Copyright (c) 2019, Intel Corporation. All rights reserved.
 This program and the accompanying materials are licensed and made available under the terms and conditions of the BSD License that accompanies this distribution. The full text of the license may be found at <http://opensource.org/licenses/bsd-license.php>.

THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

14.7 ConfigBlockLib.h File Reference

Header file for Config Block Lib implementation.

This graph shows which files directly or indirectly include this file:



Functions

- **EFI_STATUS CreateConfigBlockTable** (IN UINT16 TotalSize, OUT VOID **ConfigBlockTableAddress)
Create config block table.
- **EFI_STATUS AddConfigBlock** (IN VOID *ConfigBlockTableAddress, OUT VOID **ConfigBlockAddress)
Add config block into config block table structure.
- **EFI_STATUS GetConfigBlock** (IN VOID *ConfigBlockTableAddress, IN EFI_GUID *ConfigBlockGuid, OUT VOID **ConfigBlockAddress)
Retrieve a specific Config Block data by GUID.

14.7.1 Detailed Description

Header file for Config Block Lib implementation.

Copyright

INTEL CONFIDENTIAL Copyright (c) 2019, Intel Corporation. All rights reserved.

This program and the accompanying materials are licensed and made available under the terms and conditions of the BSD License that accompanies this distribution. The full text of the license may be found at <http://opensource.org/licenses/bsd-license.php>.

THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

14.7.2 Function Documentation

14.7.2.1 AddConfigBlock()

```
EFI_STATUS AddConfigBlock (
    IN VOID * ConfigBlockTableAddress,
    OUT VOID ** ConfigBlockAddress )
```

Add config block into config block table structure.

Parameters

in	<i>ConfigBlockTableAddress</i>	- A pointer to the beginning of Config Block Table Address
out	<i>ConfigBlockAddress</i>	- On return, points to a pointer to the beginning of Config Block Address

Return values

<i>EFI_OUT_OF_RESOURCES</i>	- Config Block Table is full and cannot add new Config Block or Config Block Offset Table is full and cannot add new Config Block.
<i>EFI_SUCCESS</i>	- Successfully added Config Block

14.7.2.2 CreateConfigBlockTable()

```
EFI_STATUS CreateConfigBlockTable (
    IN UINT16 TotalSize,
    OUT VOID ** ConfigBlockTableAddress )
```

Create config block table.

Parameters

in	<i>TotalSize</i>	- Max size to be allocated for the Config Block Table
out	<i>ConfigBlockTableAddress</i>	- On return, points to a pointer to the beginning of Config Block Table Address

Return values

<i>EFI_INVALID_PARAMETER</i>	- Invalid Parameter
<i>EFI_OUT_OF_RESOURCES</i>	- Out of resources
<i>EFI_SUCCESS</i>	- Successfully created Config Block Table at ConfigBlockTableAddress

14.7.2.3 GetConfigBlock()

```
EFI_STATUS GetConfigBlock (
    IN VOID * ConfigBlockTableAddress,
    IN EFI_GUID * ConfigBlockGuid,
    OUT VOID ** ConfigBlockAddress )
```

Retrieve a specific Config Block data by [GUID](#).

Parameters

in	<i>ConfigBlockTableAddress</i>	- A pointer to the beginning of Config Block Table Address
in	<i>ConfigBlockGuid</i>	- A pointer to the GUID uses to search specific Config Block
out	<i>ConfigBlockAddress</i>	- On return, points to a pointer to the beginning of Config Block Address

Return values

<i>EFI_NOT_FOUND</i>	- Could not find the Config Block
<i>EFI_SUCCESS</i>	- Config Block found and return

14.8 CpuConfig.h File Reference

CPU Config Block.

Classes

- struct [CPU_CONFIG](#)
CPU Configuration Structure.

14.8.1 Detailed Description

CPU Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.9 CpuConfigLibPreMemConfig.h File Reference

CPU Security PreMemory Config Block.

Classes

- struct [CPU_CONFIG_LIB_PREMEM_CONFIG](#)
CPU Config Library PreMemory Configuration Structure.

14.9.1 Detailed Description

CPU Security PreMemory Config Block.

Copyright

INTEL CONFIDENTIAL Copyright (c) 2015 - 2021 Intel Corporation. All rights reserved This software and associated documentation (if any) is furnished under a license and may only be used or copied in accordance with the terms of the license. Except as permitted by the license, no part of this software or documentation may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without the express written consent of Intel Corporation. This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

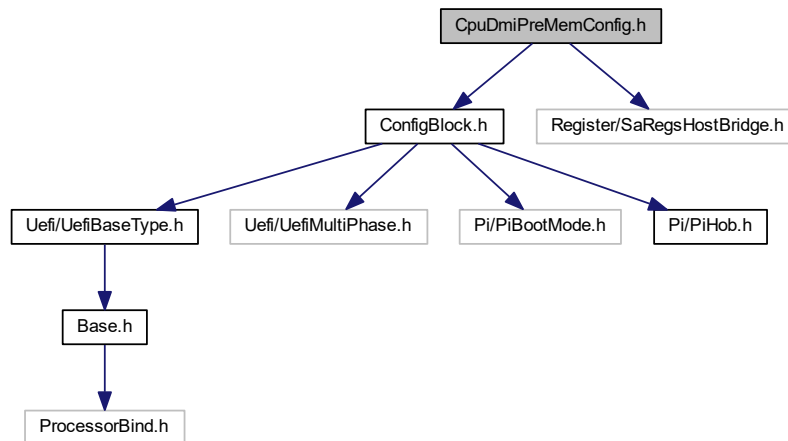
Specification Reference:

14.10 CpuDmiPreMemConfig.h File Reference

DMI policy.

```
#include <ConfigBlock.h>
#include <Register/SaRegsHostBridge.h>
```

Include dependency graph for CpuDmiPreMemConfig.h:



Classes

- struct [CPU_DMI_EQ_PARAM](#)
Represent lane specific Dmi Gen3 equalization parameters.
- struct [CPU_DMI_PREMEM_CONFIG](#)
The CPU_DMI_CONFIG block describes the expected configuration of the CPU for DMI.

Enumerations

- enum [DMI_ASPM](#)
The values before AutoConfig match the setting of PCI Express Base Specification 1.1, please be careful for adding new feature.

14.10.1 Detailed Description

DMI policy.

Copyright

INTEL CONFIDENTIAL Copyright 2019 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

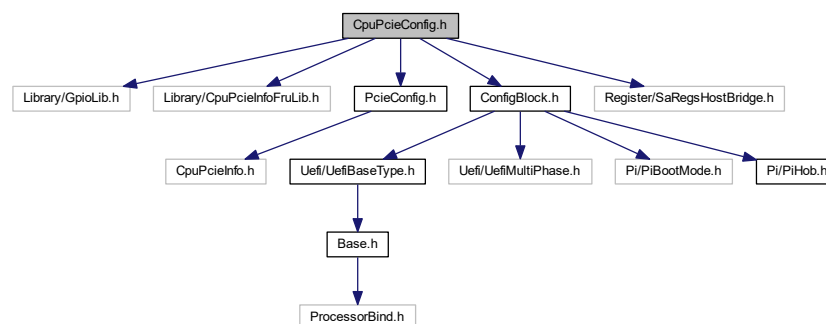
This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:**14.11 CpuPcieConfig.h File Reference**

Pcie root port policy.

```
#include <Library/GpioLib.h>
#include <Library/CpuPcieInfoFruLib.h>
#include <PcieConfig.h>
#include <ConfigBlock.h>
#include <Register/SaRegsHostBridge.h>
```

Include dependency graph for CpuPcieConfig.h:



Classes

- struct [PCIE_PEI_PREMEM_CONFIG](#)
PCI Express and DMI controller configuration
- struct [CPU_PCIE_EQ_LANE_PARAM](#)
Represent lane specific PCIe Gen3 equalization parameters.
- struct [CPU_PCIE_ROOT_PORT_CONFIG](#)
The CPU_PCIE_ROOT_PORT_CONFIG describe the feature and capability of each CPU PCIe root port.
- struct [CPU_PCIE_CONFIG](#)
*The CPU_PCIE_CONFIG block describes the expected configuration of the CPU PCI Express controllers **Revision 1** < / b>: -Initial version.*
- struct [CPU_PCIE_RP_PREMEM_CONFIG](#)
*CPU PCIe Root Port Pre-Memory Configuration Contains Root Port settings and capabilities **Revision 1**: - Initial version.*

Macros

- #define [CPU_PCIE_CONFIG_REVISION](#) 11
Making any setup structure change after code frozen will need to maintain backward compatibility, bump up structure revision and update below history table
Revision 1: - Initial version.

Enumerations

- enum [CPU_PCIE_ASPM_CONTROL](#)
The values before AutoConfig match the setting of PCI Express Base Specification 1.1, please be careful for adding new feature.
- enum [CPU_PCIE_L1SUBSTATES_CONTROL](#)
Refer to SA EDS for the SA implementation values corresponding to below PCI-E spec defined ranges.
- enum [CPU_PCIE_EQ_METHOD](#)

14.11.1 Detailed Description

Pcie root port policy.

Copyright

INTEL CONFIDENTIAL Copyright 2018 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.11.2 Macro Definition Documentation

14.11.2.1 CPU_PCIE_CONFIG_REVISION

```
#define CPU_PCIE_CONFIG_REVISION 11
```

Making any setup structure change after code frozen will need to maintain backward compatibility, bump up structure revision and update below history table

Revision 1: - Initial version.

Revision 2: - Add Gen3TxOverride and Gen4TxOverride **Revision 3:** - Deprecate Dekel Sugelch Workaround Setup Variable **Revision 4:** - Add FOMS Control Policy Setup Variable **Revision 5:** - Add Gen3HwEqOverride and Gen4HwEqOverride **Revision 6:** - Align revision with CPU_PCIE_CONFIG_REVISION value **Revision 7:**

- Deprecating Gen3PresetCoeffSelection and Gen4PresetCoeffSelection **Revision 8 :**
 - Move PcieGen3LinkEqPlatformSettings policies from [CPU_PCIE_CONFIG](#) to PCIE_ROOT_PORT_↔COMMON_CONFIG
 - Move PcieGen4LinkEqPlatformSettings policies from [CPU_PCIE_CONFIG](#) to PCIE_ROOT_PORT_↔COMMON_CONFIG
 - Move PcieGen5LinkEqPlatformSettings policies from [CPU_PCIE_CONFIG](#) to PCIE_ROOT_PORT_↔COMMON_CONFIG
 - Move OverrideEqualizationDefaults policy from [CPU_PCIE_CONFIG](#) to PCIE_ROOT_PORT_COM↔MON_CONFIG **Revision 10:** - Add Port0 Link Disable Setup Variable to CPU_PCIE_ROOT_PORT_↔CONFIG2 **Revision 11:**
 - Added MemorySize and MemoryBuffer

Definition at line 72 of file CpuPcieConfig.h.

14.11.3 Enumeration Type Documentation

14.11.3.1 CPU_PCIE_EQ_METHOD

```
enum CPU_PCIE_EQ_METHOD
```

Enumerator

CpuPcieEqDefault	Deprecated since revision 3. Behaves as PchPcieEqHardware.
CpuPcieEqHardware	Hardware equalization.
CpuPcieEqStaticCoeff	Fixed equalization (requires Coefficient settings per lane)

Definition at line 221 of file CpuPcieConfig.h.

14.12 CpuPowerMgmtBasicConfig.h File Reference

CPU Power Management Basic Config Block.

Classes

- struct [CPU_POWER_MGMT_BASIC_CONFIG](#)
CPU Power Management Basic Configuration Structure.

14.12.1 Detailed Description

CPU Power Management Basic Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2022 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.13 CpuPowerMgmtCustomConfig.h File Reference

CPU Power Management Custom Config Block.

Classes

- struct [PPM_CUSTOM_RATIO_TABLE](#)
This structure is used to describe the custom processor ratio table desired by the platform.
- struct [_PPM_CUSTOM_CTDp_TABLE](#)
PPM Custom ConfigTdp Settings.
- struct [CPU_POWER_MGMT_CUSTOM_CONFIG](#)
CPU Power Management Custom Configuration Structure.

Macros

- #define [MAX_CUSTOM_RATIO_TABLE_ENTRIES](#) 40
Defines the maximum number of custom ratio states supported.
- #define [MAX_CUSTOM_CTDp_ENTRIES](#) 3
Defines the maximum number of custom ConfigTdp entries supported.

Typedefs

- typedef struct [_PPM_CUSTOM_CTDp_TABLE](#) [PPM_CUSTOM_CTDp_TABLE](#)
PPM Custom ConfigTdp Settings.

14.13.1 Detailed Description

CPU Power Managment Custom Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2016 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.13.2 Macro Definition Documentation

14.13.2.1 MAX_CUSTOM_CTDTP_ENTRIES

```
#define MAX_CUSTOM_CTDTP_ENTRIES 3
```

Defines the maximum number of custom ConfigTdp entries supported.

Warning

: Changing this define would cause DWORD alignment issues in policy structures.

Definition at line 54 of file CpuPowerMgmtCustomConfig.h.

14.14 CpuPowerMgmtPsysConfig.h File Reference

CPU Power Management Psys(Platform) Config Block.

Classes

- struct [CPU_POWER_MGMT_PSYS_CONFIG](#)
CPU Power Management Psys(Platform) Configuration Structure.

14.14.1 Detailed Description

CPU Power Management Psys(Platform) Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2016 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.15 CpuPowerMgmtTestConfig.h File Reference

CPU Power Management Test Config Block.

Classes

- struct [CPU_POWER_MGMT_TEST_CONFIG](#)
CPU Power Management Test Configuration Structure.

Enumerations

- enum [MAX_PKG_C_STATE](#)
PPM Package C State Limit.
- enum [C_STATE_TIME_UNIT](#)
PPM Package C State Time Limit.
- enum [CUSTOM_POWER_UNIT](#)
Custom Power Units.
- enum [PPM_IRM_SETTING](#)
PPM Interrupt Redirection Mode Selection.

14.15.1 Detailed Description

CPU Power Management Test Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.15.2 Enumeration Type Documentation

14.15.2.1 CUSTOM_POWER_UNIT

enum [CUSTOM_POWER_UNIT](#)

Custom Power Units.

User can choose to enter in watts or 125 milliwatt increments.

Enumerator

PowerUnitWatts	in Watts.
PowerUnit125MilliWatts	in 125 milliwatt increments. Example: 90 power units times 125 mW equals 11.250 W.

Definition at line 78 of file CpuPowerMgmtTestConfig.h.

14.16 CpuPowerMgmtVrConfig.h File Reference

CPU Power Management VR Config Block.

Classes

- struct [CPU_POWER_MGMT_VR_CONFIG](#)
CPU Power Management VR Configuration Structure.

Macros

- #define [MAX_NUM_VRS](#) 5
Defines the maximum number of VR domains supported.

14.16.1 Detailed Description

CPU Power Management VR Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.16.2 Macro Definition Documentation

14.16.2.1 MAX_NUM_VRS

```
#define MAX_NUM_VRS 5
```

Defines the maximum number of VR domains supported.

Warning

: Changing this define would cause DWORD alignment issues in policy structures.

Definition at line 48 of file CpuPowerMgmtVrConfig.h.

14.17 CpuSecurityPreMemConfig.h File Reference

CPU Security PreMemory Config Block.

Classes

- struct [CPU_SECURITY_PREMEM_CONFIG](#)
CPU Security PreMemory Configuration Structure.

14.17.1 Detailed Description

CPU Security PreMemory Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.18 CpuTestConfig.h File Reference

CPU Test Config Block.

Classes

- struct [CPU_TEST_CONFIG](#)
CPU Test Configuration Structure.

14.18.1 Detailed Description

CPU Test Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.19 CpuTxtConfig.h File Reference

CPU TXT PreMemory Config Block.

Classes

- struct [CPU_TXT_PREMEM_CONFIG](#)
CPU TXT PreMemory Configuration Structure.

14.19.1 Detailed Description

CPU TXT PreMemory Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2019 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.20 DciConfig.h File Reference

Dci policy.

Classes

- struct [PCH_DCI_PREMEM_CONFIG](#)

The [PCH_DCI_PREMEM_CONFIG](#) block describes policies related to Direct Connection Interface (DCI)

14.20.1 Detailed Description

Dci policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.21 EspiConfig.h File Reference

Espi policy.

Classes

- struct [PCH_ESPI_CONFIG](#)

This structure contains the policies which are related to ESPI.

14.21.1 Detailed Description

Espi policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

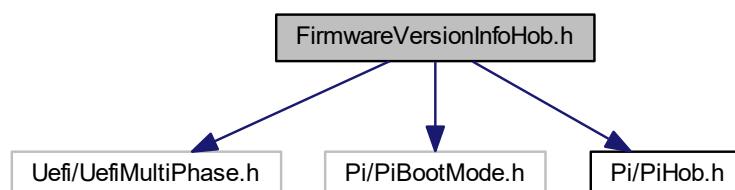
This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.22 FirmwareVersionInfoHob.h File Reference

Header file for Firmware Version Information.

```
#include <Uefi/UefiMultiPhase.h>
#include <Pi/PiBootMode.h>
#include <Pi/PiHob.h>
Include dependency graph for FirmwareVersionInfoHob.h:
```



Classes

- struct [FIRMWARE_VERSION](#)
Firmware Version Structure.
- struct [FIRMWARE_VERSION_INFO](#)
Firmware Version Information Structure.
- struct [SMBIOS_STRUCTURE](#)
The Smbios structure header.
- struct [FIRMWARE_VERSION_INFO_HOB](#)
Firmware Version Information HOB Structure.

14.22.1 Detailed Description

Header file for Firmware Version Information.

Copyright

Copyright (c) 2015 - 2018, Intel Corporation. All rights reserved.

This program and the accompanying materials are licensed and made available under the terms and conditions of the BSD License which accompanies this distribution. The full text of the license may be found at <http://opensource.org/licenses/bsd-license.php>

THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

14.23 FivrConfig.h File Reference

PCH FIVR policy.

Classes

- struct [FIVR_EXT_RAIL_CONFIG](#)
Structure for V1p05/Vnn VR rail configuration.
- struct [FIVR_VCCIN_AUX_CONFIG](#)
Structure for VCCIN_AUX voltage rail configuration.
- struct [PCH_FIVR_CONFIG](#)
The [PCH_FIVR_CONFIG](#) block describes FIVR settings.

Macros

- #define [PCH_FIVR_CONFIG_REVISION](#) 3
Revision 1: - Initial version.

Enumerations

- enum [FIVR_RAIL_SX_STATE](#)

Rail support in S0ix and Sx Settings other than FivrRailDisabled can be OR'ed.

14.23.1 Detailed Description

PCH FIVR policy.

Copyright

INTEL CONFIDENTIAL Copyright 2017 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.23.2 Macro Definition Documentation

14.23.2.1 PCH_FIVR_CONFIG_REVISION

```
#define PCH_FIVR_CONFIG_REVISION 3
```

Revision 1: - Initial version.

Revision 2: - Added BypassFivrConfig **Revision 3:** - Added FivrVccstlccMaxControl

Definition at line 43 of file FivrConfig.h.

14.24 FspErrorInfo.h File Reference

FSP Error Information HOB to describe errors inside FSP that bootloader may take some actions to handle those error scenarios.

Classes

- struct [FSP_ERROR_INFO_HOB](#)
FSP Error Information Block.

Macros

- #define [FSP_ERROR_INFO_HOB_GUID](#)
[GUID](#) value indicating the FSP error information.

14.24.1 Detailed Description

FSP Error Information HOB to describe errors inside FSP that bootloader may take some actions to handle those error scenarios.

Copyright

INTEL CONFIDENTIAL Copyright 2019 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.25 FspFixedPcds.h File Reference

This file lists all FixedAtBuild PCDs referenced in FSP integration guide.

Macros

- #define [PcdFspAreaBaseAddress](#) 0xFFE00000
FspAreaBaseAddress.
- #define [PcdFspImageIdString](#) \$ADLFSP\$
FspImageIdString.
- #define [PcdSiliconInitVersionMajor](#) 0x0C
SiliconInitVersionMajor.
- #define [PcdSiliconInitVersionMinor](#) 0x00
SiliconInitVersionMinor.
- #define [PcdSiliconInitVersionRevision](#) 0xC8
SiliconInitVersionRevision.
- #define [PcdSiliconInitVersionBuild](#) 0x50
SiliconInitVersionBuild.
- #define [PcdGlobalDataPointerAddress](#) 0xFED00148
GlobalDataPointerAddress.
- #define [PcdTemporaryRamBase](#) 0xFE000000
TemporaryRamBase.
- #define [PcdTemporaryRamSize](#) 0x00100000
TemporaryRamSize.
- #define [PcdFspReservedBufferSize](#) 0x100
FspReservedBufferSize.

14.25.1 Detailed Description

This file lists all FixedAtBuild PCDs referenced in FSP integration guide.

Those value may vary in different FSP revision to meet different requirements.

14.26 FspInfoHob.h File Reference

Header file for FSP Information HOB.

14.26.1 Detailed Description

Header file for FSP Information HOB.

Copyright

INTEL CONFIDENTIAL Copyright 2017 - 2019 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.27 FspmArchConfigPpi.h File Reference

Header file for FSP-M Arch Config PPI.

Classes

- struct [FSPM_ARCH_CONFIG_PPI](#)
This PPI provides FSP-M Arch Config PPI.

Macros

- #define [FSPM_ARCH_CONFIG_GUID](#)
Global ID for the [FSPM_ARCH_CONFIG_PPI](#).

14.27.1 Detailed Description

Header file for FSP-M Arch Config PPI.

Copyright

INTEL CONFIDENTIAL Copyright (c) 2018 - 2019, Intel Corporation. All rights reserved.

This program and the accompanying materials are licensed and made available under the terms and conditions of the BSD License which accompanies this distribution. The full text of the license may be found at <http://opensource.org/licenses/bsd-license.php>

THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

14.28 GbeConfig.h File Reference

Gigabit Ethernet policy.

Classes

- struct [GBE_CONFIG](#)
PCH intergrated GBE controller configuration settings.

14.28.1 Detailed Description

Gigabit Ethernet policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2018 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.29 GnaConfig.h File Reference

Policy definition for GNA Config Block.

Classes

- struct [GNA_CONFIG](#)
GNA config block for configuring GNA.

14.29.1 Detailed Description

Policy definition for GNA Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2019 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.30 GpioConfig.h File Reference

Header file for GpioConfig structure used by GPIO library.

Classes

- struct [GPIO_CONFIG](#)
GPIO configuration structure used for pin programming.

Macros

- `#define B_GPIO_INT_CONFIG_INT_SOURCE_MASK 0x1F`
Mask for GPIO_INT_CONFIG for interrupt source.
- `#define B_GPIO_INT_CONFIG_INT_TYPE_MASK 0xE0`
Mask for GPIO_INT_CONFIG for interrupt type.
- `#define B_GPIO_ELECTRICAL_CONFIG_TERMINATION_MASK 0x1F`
Mask for GPIO_ELECTRICAL_CONFIG for termination value.
- `#define B_GPIO_ELECTRICAL_CONFIG_1V8_TOLERANCE_MASK 0x60`
Mask for GPIO_ELECTRICAL_CONFIG for 1v8 tolerance setting.
- `#define B_GPIO_LOCK_CONFIG_PAD_CONF_LOCK_MASK 0x3`
Mask for GPIO_LOCK_CONFIG for Pad Configuration Lock.
- `#define B_GPIO_LOCK_CONFIG_OUTPUT_LOCK_MASK 0x5`
Mask for GPIO_LOCK_CONFIG for Pad Output Lock.
- `#define B_GPIO_OTHER_CONFIG_RXRAW_MASK 0x3`
Mask for GPIO_OTHER_CONFIG for RxRaw1 setting.

Typedefs

- typedef UINT32 [GPIO_PAD](#)
For any GpioPad usage in code use GPIO_PAD type.
- typedef UINT32 [GPIO_GROUP](#)
For any GpioGroup usage in code use GPIO_GROUP type.

Enumerations

- enum [GPIO_HARDWARE_DEFAULT](#)
- enum [GPIO_PAD_MODE](#)
GPIO Pad Mode Refer to GPIO documentation on native functions available for certain pad.
- enum [GPIO_HOSTSW_OWN](#)
Host Software Pad Ownership modes This setting affects GPIO interrupt status registers.
- enum [GPIO_DIRECTION](#)
GPIO Direction.
- enum [GPIO_OUTPUT_STATE](#)
GPIO Output State This field is relevant only if output is enabled.
- enum [GPIO_INT_CONFIG](#)
GPIO interrupt configuration This setting is applicable only if pad is in GPIO mode and has input enabled.
- enum [GPIO_RESET_CONFIG](#)
GPIO Power Configuration GPIO_RESET_CONFIG allows to set GPIO Reset type (PADCFG_DW0.PadRstCfg) which will be used to reset certain GPIO settings.
- enum [GPIO_ELECTRICAL_CONFIG](#)
GPIO Electrical Configuration Set GPIO termination and Pad Tolerance (applicable only for some pads) Field from GpioTermNone to GpioTermNative can be OR'ed with GpioTolerance1v8.
- enum [GPIO_LOCK_CONFIG](#)
GPIO LockConfiguration Set GPIO configuration lock and output state lock.
- enum [GPIO_OTHER_CONFIG](#)
Other GPIO Configuration GPIO_OTHER_CONFIG is used for less often settings and for future extensions Supported settings:

14.30.1 Detailed Description

Header file for GpioConfig structure used by GPIO library.

Copyright

INTEL CONFIDENTIAL Copyright 2014 - 2017 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.30.2 Enumeration Type Documentation

14.30.2.1 GPIO_DIRECTION

enum [GPIO_DIRECTION](#)

GPIO Direction.

Enumerator

GpioDirDefault	Leave pad direction setting unmodified.
GpioDirInOut	Set pad for both output and input.
GpioDirInInvOut	Set pad for both output and input with inversion.
GpioDirIn	Set pad for input only.
GpioDirInInv	Set pad for input with inversion.
GpioDirOut	Set pad for output only.
GpioDirNone	Disable both output and input.

Definition at line 167 of file GpioConfig.h.

14.30.2.2 GPIO_ELECTRICAL_CONFIG

enum [GPIO_ELECTRICAL_CONFIG](#)

GPIO Electrical Configuration Set GPIO termination and Pad Tolerance (applicable only for some pads) Field from GpioTermNone to GpioTermNative can be OR'ed with GpioTolerance1v8.

Enumerator

GpioTermDefault	Leave termination setting unmodified.
GpioTermNone	none
GpioTermWpd5K	5kOhm weak pull-down
GpioTermWpd20K	20kOhm weak pull-down
GpioTermWpu1K	1kOhm weak pull-up
GpioTermWpu2K	2kOhm weak pull-up
GpioTermWpu5K	5kOhm weak pull-up
GpioTermWpu20K	20kOhm weak pull-up
GpioTermWpu1K2K	1kOhm & 2kOhm weak pull-up
GpioTermNative	Native function controls pads termination This setting is applicable only to some native modes. Please check EDS to determine which native functionality can control pads termination
GpioNoTolerance1v8	Disable 1.8V pad tolerance.
GpioTolerance1v8	Enable 1.8V pad tolerance.

Definition at line 296 of file GpioConfig.h.

14.30.2.3 GPIO_HARDWARE_DEFAULT

enum [GPIO_HARDWARE_DEFAULT](#)

Enumerator

GpioHardwareDefault	Leave setting unmodified.
---------------------	---------------------------

Definition at line 118 of file GpioConfig.h.

14.30.2.4 GPIO_HOSTSW_OWN

enum [GPIO_HOSTSW_OWN](#)

Host Software Pad Ownership modes This setting affects GPIO interrupt status registers.

Depending on chosen ownership some GPIO Interrupt status register get updated and other masked. Please refer to EDS for HOSTSW_OWN register description.

Enumerator

GpioHostOwnDefault	Leave ownership value unmodified.
GpioHostOwnAcpi	Set HOST ownership to ACPI. Use this setting if pad is not going to be used by GPIO OS driver. If GPIO is configured to generate SCI/SMI/NMI then this setting must be used for interrupts to work
GpioHostOwnGpio	Set HOST ownership to GPIO Driver mode. Use this setting only if GPIO pad should be controlled by GPIO OS Driver. GPIO OS Driver will be able to control the pad if appropriate entry in ACPI exists (refer to ACPI specification for GpioIo and GpioInt descriptors)

Definition at line 146 of file GpioConfig.h.

14.30.2.5 GPIO_INT_CONFIG

enum [GPIO_INT_CONFIG](#)

GPIO interrupt configuration This setting is applicable only if pad is in GPIO mode and has input enabled.

GPIO_INT_CONFIG allows to choose which interrupt is generated (IOxAPIC/SCI/SMI/NMI) and how it is triggered (edge or level). Refer to PADCFG_DW0 register description in EDS for details on this settings. Field from Gpio↔IntNmi to GpioIntApic can be OR'ed with GpioIntLevel to GpioIntBothEdge to describe an interrupt e.g. GpioIntApic

| GpioIntLevel If GPIO is set to cause an SCI then also GPI_GPE_EN is enabled for this pad. If GPIO is set to cause an NMI then also GPI_NMI_EN is enabled for this pad. Not all GPIO are capable of generating an SMI or NMI interrupt. When routing GPIO to cause an IOxAPIC interrupt care must be taken, as this interrupt cannot be shared and its IRQn number is not configurable. Refer to EDS for GPIO pads IRQ numbers (PADCFG_DW1.IntSel) If GPIO is under GPIO OS driver control and appropriate ACPI GpioInt descriptor exist then use only trigger type setting (from GpioIntLevel to GpioIntBothEdge). This type of GPIO Driver interrupt doesn't have any additional routing setting required to be set by BIOS. Interrupt is handled by GPIO OS Driver.

Enumerator

GpioIntDefault	Leave value of interrupt routing unmodified.
GpioIntDis	Disable IOxAPIC/SCI/SMI/NMI interrupt generation.
GpioIntNmi	Enable NMI interrupt only.
GpioIntSmi	Enable SMI interrupt only.
GpioIntSci	Enable SCI interrupt only.
GpioIntApic	Enable IOxAPIC interrupt only.
GpioIntLevel	Set interrupt as level triggered.
GpioIntEdge	Set interrupt as edge triggered (type of edge depends on input inversion)
GpioIntLvlEdgDis	Disable interrupt trigger.
GpioIntBothEdge	Set interrupt as both edge triggered.

Definition at line 207 of file GpioConfig.h.

14.30.2.6 GPIO_LOCK_CONFIG

enum [GPIO_LOCK_CONFIG](#)

GPIO LockConfiguration Set GPIO configuration lock and output state lock.

GpioLockPadConfig and GpioLockOutputState can be OR'ed. Lock settings reset is in Powergood domain. Care must be taken when using this setting as fields it locks may be reset by a different signal and can be controllable by what is in GPIO_RESET_CONFIG (PADCFG_DW0.PadRstCfg). GPIO library provides functions which allow to unlock a GPIO pad.

Enumerator

GpioLockDefault	Leave lock setting unmodified.
GpioPadConfigLock	Lock Pad Configuration.
GpioOutputStateLock	Lock GPIO pad output value.

Definition at line 329 of file GpioConfig.h.

14.30.2.7 GPIO_OTHER_CONFIG

enum [GPIO_OTHER_CONFIG](#)

Other GPIO Configuration GPIO_OTHER_CONFIG is used for less often settings and for future extensions Supported settings:

- RX raw override to '1' - allows to override input value to '1' This setting is applicable only if in input mode (both in GPIO and native usage). The override takes place at the internal pad state directly from buffer and before the RXINV.

Enumerator

GpioRxRaw1Default	Use default input override value.
GpioRxRaw1Dis	Don't override input.
GpioRxRaw1En	Override input to '1'.

Definition at line 346 of file GpioConfig.h.

14.30.2.8 GPIO_OUTPUT_STATE

enum [GPIO_OUTPUT_STATE](#)

GPIO Output State This field is relevant only if output is enabled.

Enumerator

GpioOutDefault	Leave output value unmodified.
GpioOutLow	Set output to low.
GpioOutHigh	Set output to high.

Definition at line 181 of file GpioConfig.h.

14.30.2.9 GPIO_PAD_MODE

enum [GPIO_PAD_MODE](#)

GPIO Pad Mode Refer to GPIO documentation on native functions available for certain pad.

If GPIO is set to one of NativeX modes then following settings are not applicable and can be skipped:

- Interrupt related settings
- Host Software Ownership
- Output/Input enabling/disabling
- Output lock

Definition at line 132 of file GpioConfig.h.

14.30.2.10 GPIO_RESET_CONFIG

enum [GPIO_RESET_CONFIG](#)

GPIO Power Configuration [GPIO_RESET_CONFIG](#) allows to set GPIO Reset type ([PADCFG_DW0.PadRstCfg](#)) which will be used to reset certain GPIO settings.

Refer to EDS for settings that are controllable by [PadRstCfg](#).

Enumerator

GpioResetDefault	Leave value of pad reset unmodified.
GpioResetPwrGood	Deprecated settings. Maintained only for compatibility. GPP: RSMRST; GPD: DSW_PWROK; (PadRstCfg = 00b = "Powergood")
GpioResetDeep	Deep GPIO Reset (PadRstCfg = 01b = "Deep GPIO Reset")
GpioResetNormal	GPIO Reset (PadRstCfg = 10b = "GPIO Reset")
GpioResetResume	GPP: Reserved; GPD: RSMRST; (PadRstCfg = 11b = "Resume Reset")
GpioResumeReset	<p>New GPIO reset configuration options. Resume Reset (RSMRST) GPP: PadRstCfg = 00b = "Powergood" GPD: PadRstCfg = 11b = "Resume Reset" Pad setting will reset on:</p> <ul style="list-style-type: none"> • DeepSx transition • G3 Pad settings will not reset on: • S3/S4/S5 transition • Warm/Cold/Global reset
GpioHostDeepReset	<p>Host Deep Reset PadRstCfg = 01b = "Deep GPIO Reset" Pad settings will reset on:</p> <ul style="list-style-type: none"> • Warm/Cold/Global reset • DeepSx transition • G3 Pad settings will not reset on: • S3/S4/S5 transition
GpioPlatformReset	<p>Platform Reset (PLTRST) PadRstCfg = 10b = "GPIO Reset" Pad settings will reset on:</p> <ul style="list-style-type: none"> • S3/S4/S5 transition • Warm/Cold/Global reset • DeepSx transition • G3
GpioDswReset	<p>Deep Sleep Well Reset (DSW_PWROK) GPP: not applicable GPD: PadRstCfg = 00b = "Powergood" Pad settings will reset on:</p> <ul style="list-style-type: none"> • G3 Pad settings will not reset on: • S3/S4/S5 transition • Warm/Cold/Global reset • DeepSx transition

Definition at line 229 of file GpioConfig.h.

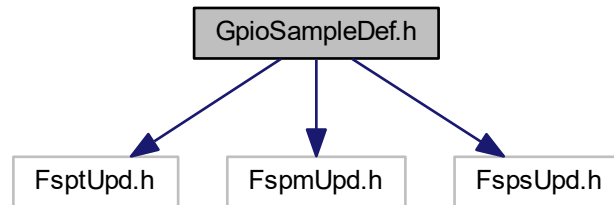
14.31 GpioSampleDef.h File Reference

Copyright (c) 2015 - 2021, Intel Corporation.

```
#include <FsptUpd.h>
#include <FspmUpd.h>
```

```
#include <FspsUpd.h>
```

Include dependency graph for GpioSampleDef.h:



14.31.1 Detailed Description

Copyright (c) 2015 - 2021, Intel Corporation.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of Intel Corporation nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This file is automatically generated. Please do NOT modify !!!

14.32 GraphicsConfig.h File Reference

Policy definition for Internal Graphics Config Block.

Classes

- struct [DDI_CONFIGURATION](#)
This structure configures the Native GPIOs for DDI port per VBT settings.
- struct [GRAPHICS_PEI_PREMEM_CONFIG](#)
This Configuration block is to configure GT related PreMem data/variables.
- struct [GRAPHICS_PEI_CONFIG](#)
This configuration block is to configure IGD related variables used in PostMem PEI.
- struct [GRAPHICS_DXE_CONFIG](#)
This configuration block is to configure IGD related variables used in DXE.

14.32.1 Detailed Description

Policy definition for Internal Graphics Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2018 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

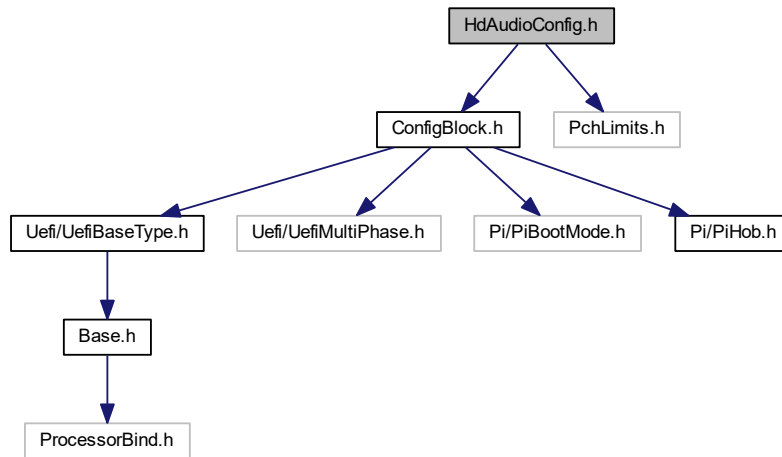
14.33 HdAudioConfig.h File Reference

HDAUDIO policy.

```
#include <ConfigBlock.h>
```

```
#include <PchLimits.h>
```

Include dependency graph for HdAudioConfig.h:



Classes

- struct [HDA_VERB_TABLE_HEADER](#)
Azalia verb table header Every verb table should contain this defined header and followed by azalia verb commands.
- struct [HDA_LINK_HDA](#)
HD Audio Link Policies.
- struct [HDA_LINK_DMIC](#)
HD Audio DMIC Interface Policies.
- struct [HDA_LINK_SSP](#)
HD Audio SSP Interface Policies.
- struct [HDA_LINK_SNDW](#)
HD Audio SNDW Interface Policies.
- struct [HDAUDIO_CONFIG](#)
This structure contains the policies which are related to HD Audio device (cAVS).
- struct [HDAUDIO_PREMEM_CONFIG](#)
This structure contains the premem policies which are related to HD Audio device (cAVS).
- struct [HDAUDIO_DXE_CONFIG](#)
This structure contains the DXE policies which are related to HD Audio device (cAVS).

Macros

- #define [HDAUDIO_VERB_TABLE_VIDDID](#)(Vid, Did) (UINT32)((UINT16)Vid | ((UINT16)Did << 16))
The PCH_HDAUDIO_CONFIG block describes the expected configuration of the Intel HD Audio feature.
- #define [HDAUDIO_VERB_TABLE_INIT](#)(Vid, Did, Rid, Sdi, ...)
Use this macro to create HDAUDIO_VERB_TABLE and populate size automatically.

14.33.1 Detailed Description

HDAUDIO policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2023 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.34 HybridGraphicsConfig.h File Reference

Hybrid Graphics policy definitions.

Classes

- struct [CPU_PCIE_GPIO_INFO](#)
CPU PCIe GPIO Data Structure.
- struct [CPU_PCIE_RTD3_GPIO](#)
CPU PCIe RTD3 GPIO Data Structure.
- struct [HYBRID_GRAPHICS_CONFIG](#)
This Configuration block configures CPU PCI Express 0/1/2 RTD3 GPIOs & Root Port.

Enumerations

- enum [GPIO_SUPPORT](#)
GPIO Support.

14.34.1 Detailed Description

Hybrid Graphics policy definitions.

Copyright

INTEL CONFIDENTIAL Copyright 2014 - 2019 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

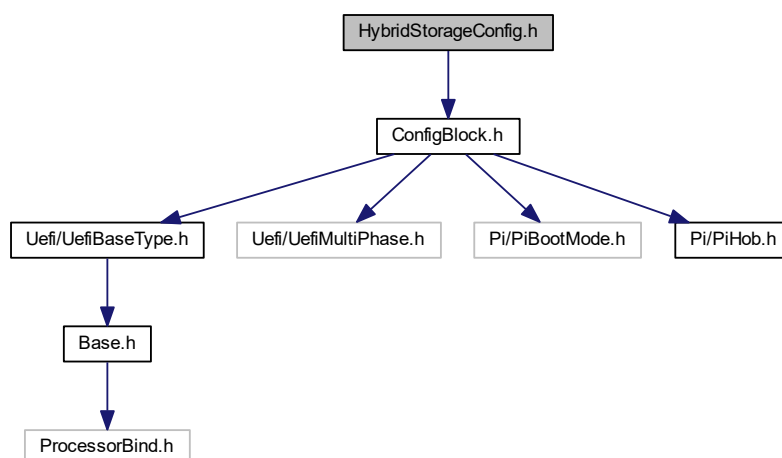
Specification Reference:

14.35 HybridStorageConfig.h File Reference

Hybrid Storage policy.

```
#include <ConfigBlock.h>
```

Include dependency graph for HybridStorageConfig.h:



Classes

- struct [HYBRID_STORAGE_CONFIG](#)

The [HYBRID_STORAGE_CONFIG](#) block describes the expected configuration for Hybrid Storage device.

14.35.1 Detailed Description

Hybrid Storage policy.

Copyright

INTEL CONFIDENTIAL Copyright 2019 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

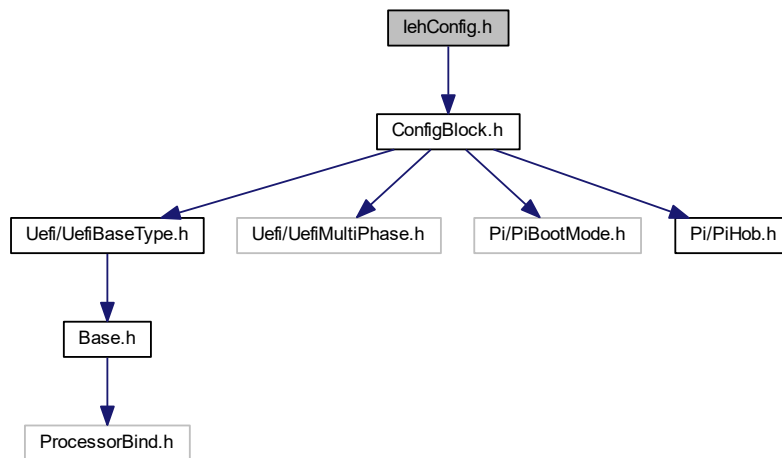
Specification Reference:

14.36 lehConfig.h File Reference

Integrated Error Handler policy.


```
#include <ConfigBlock.h>
```

Include dependency graph for IehConfig.h:



Classes

- struct [IEH_CONFIG](#)

The [IEH_CONFIG](#) block describes the expected configuration of the PCH Integrated Error Handler.

14.36.1 Detailed Description

Integrated Error Handler policy.

Copyright

INTEL CONFIDENTIAL Copyright 2018 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.37 InterruptConfig.h File Reference

Interrupt policy.

Classes

- struct [PCH_DEVICE_INTERRUPT_CONFIG](#)
The [PCH_DEVICE_INTERRUPT_CONFIG](#) block describes interrupt pin, IRQ and interrupt mode for PCH device.
- struct [PCH_INTERRUPT_CONFIG](#)
The [PCH_INTERRUPT_CONFIG](#) block describes interrupt settings for PCH.

Macros

- #define [PCH_MAX_DEVICE_INTERRUPT_CONFIG](#) 128
Number of all PCH devices.
- #define [PCH_MAX_PXRC_CONFIG](#) 8
Number of PXRC registers in ITSS.
- #define [PCH_MAX_ITSS_IPC_REGS](#) 4
Number of IPC registers in ITSS.
- #define [PCH_MAX_ITSS_IRQ_NUM](#) 120
Maximum number of IRQs.

Enumerations

- enum [PCH_INT_PIN](#)

14.37.1 Detailed Description

Interrupt policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.37.2 Enumeration Type Documentation

14.37.2.1 PCH_INT_PIN

enum [PCH_INT_PIN](#)

Enumerator

PchNoInt	No Interrupt Pin.
----------	-------------------

Definition at line 46 of file InterruptConfig.h.

14.38 IoApicConfig.h File Reference

IoApic policy.

Classes

- struct [PCH_IOAPIC_CONFIG](#)

The [PCH_IOAPIC_CONFIG](#) block describes the expected configuration of the PCH IO APIC, it's optional and PCH code would ignore it if the BdfValid bit is not TRUE.

14.38.1 Detailed Description

IoApic policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

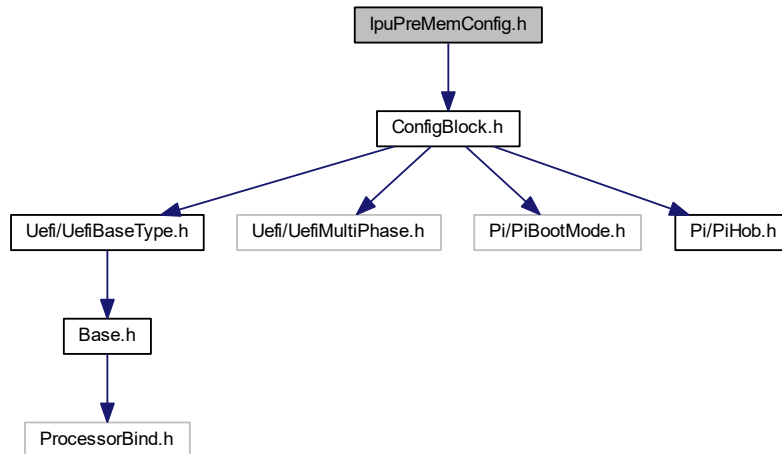
Specification Reference:

14.39 IpuPreMemConfig.h File Reference

IPU policy definitions.

```
#include <ConfigBlock.h>
```

Include dependency graph for IpuPreMemConfig.h:



Classes

- struct [IPU_PREMEM_CONFIG](#)
IPU PreMem configuration
Revision 1:

14.39.1 Detailed Description

IPU policy definitions.

Copyright

INTEL CONFIDENTIAL Copyright 2017 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.40 IshConfig.h File Reference

ISH policy.

Classes

- struct [ISH_GPIO_CONFIG](#)
ISH GPIO settings.
- struct [ISH_SPI_PIN_CONFIG](#)
SPI signals settings.
- struct [ISH_UART_PIN_CONFIG](#)
UART signals settings.
- struct [ISH_I2C_PIN_CONFIG](#)
I2C signals settings.
- struct [ISH_SPI](#)
Struct contains GPIO pins assigned and signal settings of SPI.
- struct [ISH_UART](#)
Struct contains GPIO pins assigned and signal settings of UART.
- struct [ISH_I2C](#)
Struct contains GPIO pins assigned and signal settings of I2C.
- struct [ISH_GP](#)
Struct contains GPIO pins assigned and signal settings of GP.
- struct [ISH_CONFIG](#)
The [ISH_CONFIG](#) block describes Integrated Sensor Hub device.
- struct [ISH_I3C_CONFIG](#)
The [ISH_I3C_CONFIG](#) block describes I3C in ISH device.
- struct [ISH_PREMEM_CONFIG](#)
Premem Policy for Integrated Sensor Hub device.

14.40.1 Detailed Description

ISH policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.41 LockDownConfig.h File Reference

Lock down policy.

Classes

- struct [PCH_LOCK_DOWN_CONFIG](#)

The [PCH_LOCK_DOWN_CONFIG](#) block describes the expected configuration of the PCH for security requirement.

14.41.1 Detailed Description

Lock down policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.42 MemoryConfig.h File Reference

Policy definition of Memory Config Block.

Classes

- struct [SPD_DATA_BUFFER](#)
SPD Data Buffer.
- struct [SA_MEMORY_DQDQS_MAPPING](#)
DqDqs Mapping.
- struct [SA_MEMORY_RCOMP](#)
Rcomp Policies.
- struct [SPD_OFFSET_TABLE](#)
SPD Offset Table.
- struct [SA_ADDRESS_DECODE](#)
SA memory address decode.
- struct [SA_FUNCTION_CALLS](#)
Function calls into the SA.
- struct [SA_MEMORY_FUNCTIONS](#)
Function calls into the MRC.
- struct [MEMORY_CONFIGURATION](#)
Memory Configuration The contents of this structure are CRC'd by the MRC for option change detection.
- struct [MEMORY_CONFIG_NO_CRC](#)
Memory Configuration The contents of this structure are not CRC'd by the MRC for option change detection.

Macros

- #define [MRC_MAX_RCOMP_TARGETS](#) 5
MEMORY_CONFIGURATION interface definitions.
- #define [MEM_CFG_MAX_CONTROLLERS](#) 2
Memory SubSystem Definitions.
- #define [PEI_MR_SMRAM_ABSEG_MASK](#) 0x01
SMRAM Memory Range.

Typedefs

- typedef UINT8(* [SA_IO_READ_8](#)) (UINTN IoAddress)
CPU I/O port 8-bit read.
- typedef UINT16(* [SA_IO_READ_16](#)) (UINTN IoAddress)
CPU I/O port 16-bit read.
- typedef UINT32(* [SA_IO_READ_32](#)) (UINTN IoAddress)
CPU I/O port 32-bit read.
- typedef UINT8(* [SA_IO_WRITE_8](#)) (UINTN IoAddress, UINT8 Value)
CPU I/O port 8-bit write.
- typedef UINT16(* [SA_IO_WRITE_16](#)) (UINTN IoAddress, UINT16 Value)
CPU I/O port 16-bit write.
- typedef UINT32(* [SA_IO_WRITE_32](#)) (UINTN IoAddress, UINT32 Value)
CPU I/O port 32-bit write.
- typedef UINT8(* [SA_MMIO_READ_8](#)) (UINTN Address)
Memory Mapped I/O port 8-bit read.
- typedef UINT16(* [SA_MMIO_READ_16](#)) (UINTN Address)
Memory Mapped I/O port 16-bit read.
- typedef UINT32(* [SA_MMIO_READ_32](#)) (UINTN Address)
Memory Mapped I/O port 32-bit read.
- typedef UINT64(* [SA_MMIO_READ_64](#)) (UINTN Address)
Memory Mapped I/O port 64-bit read.
- typedef UINT8(* [SA_MMIO_WRITE_8](#)) (UINTN Address, UINT8 Value)
Memory Mapped I/O port 8-bit write.
- typedef UINT16(* [SA_MMIO_WRITE_16](#)) (UINTN Address, UINT16 Value)
Memory Mapped I/O port 16-bit write.
- typedef UINT32(* [SA_MMIO_WRITE_32](#)) (UINTN Address, UINT32 Value)
Memory Mapped I/O port 32-bit write.
- typedef UINT64(* [SA_MMIO_WRITE_64](#)) (UINTN Address, UINT64 Value)
Memory Mapped I/O port 64-bit write.
- typedef UINT8(* [SA_SMBUS_READ_8](#)) (UINTN Address, RETURN_STATUS *Status)
Smbus 8-bit read.
- typedef UINT16(* [SA_SMBUS_READ_16](#)) (UINTN Address, RETURN_STATUS *Status)
Smbus 16-bit read.
- typedef UINT8(* [SA_SMBUS_WRITE_8](#)) (UINTN Address, UINT8 Value, RETURN_STATUS *Status)
Smbus 8-bit write.
- typedef UINT16(* [SA_SMBUS_WRITE_16](#)) (UINTN Address, UINT16 Value, RETURN_STATUS *Status)
Smbus 16-bit write.
- typedef UINT32(* [SA_GET_PCI_DEVICE_ADDRESS](#)) (UINT8 Bus, UINT8 Device, UINT8 Function, UINT8 Offset)
Get PCI device address.
- typedef UINT32(* [SA_GET_PCIE_DEVICE_ADDRESS](#)) (UINT8 Bus, UINT8 Device, UINT8 Function, UI↔NT8 Offset)
Get PCI express device address.
- typedef VOID(* [SA_GET_RTC_TIME](#)) (UINT8 *Second, UINT8 *Minute, UINT8 *Hour, UINT8 *Day, UINT8 *Month, UINT16 *Year)
Get the current time value.
- typedef UINT64(* [SA_GET_CPU_TIME](#)) (VOID)
The current CPU time in milliseconds.
- typedef VOID (*[SA_MEMORY_COPY](#)) (VOID *Destination, CONST VOID *Source, UINTN NumBytes)
Perform byte copy operation.

- typedef **VOID** (* **SA_MEMORY_SET_BYTE**) (**VOID** *Buffer, UINTN NumBytes, UINT8 Value)
Perform byte initialization operation.
- typedef **VOID** (* **SA_MEMORY_SET_WORD**) (**VOID** *Buffer, UINTN NumWords, UINT16 Value)
Perform word initialization operation.
- typedef **VOID** (* **SA_MEMORY_SET_DWORD**) (**VOID** *Buffer, UINTN NumDwords, UINT32 Value)
Perform dword initialization operation.
- typedef **UINT64**(* **SA_LEFT_SHIFT_64**) (**UINT64** Data, UINTN NumBits)
Left shift the 64-bit data value by specified number of bits.
- typedef **UINT64**(* **SA_RIGHT_SHIFT_64**) (**UINT64** Data, UINTN NumBits)
Right shift the 64-bit data value by specified number of bits.
- typedef **UINT64**(* **SA_MULT_U64_U32**) (**UINT64** Multiplicand, **UINT32** Multiplier)
Multiply a 64-bit data value by a 32-bit data value.
- typedef **UINT64**(* **SA_DIV_U64_U64**) (**UINT64** Dividend, **UINT64** Divisor, **UINT64** *Remainder)
Divide a 64-bit data value by a 64-bit data value.
- typedef **BOOLEAN**(* **SA_GET_SPD_DATA**) (**SPD_BOOT_MODE** BootMode, **UINT8** SpdAddress, **UINT8** *Buffer, **UINT8** *Ddr3Table, **UINT32** Ddr3TableSize, **UINT8** *Ddr4Table, **UINT32** Ddr4TableSize, **UINT8** *LpddrTable, **UINT32** LpddrTableSize)
Read the SPD data over the SMBus, at the given SmBus SPD address and copy the data to the data structure.
- typedef **BOOLEAN**(* **SA_GET_RANDOM_NUMBER**) (**UINT32** *Rand)
Get the next random 32-bit number.
- typedef **EFI_STATUS**(* **SA_CPU_MAILBOX_READ**) (**UINT32** Type, **UINT32** Command, **UINT32** *Value, **UINT32** *Status)
Perform a CPU mailbox read.
- typedef **EFI_STATUS**(* **SA_CPU_MAILBOX_WRITE**) (**UINT32** Type, **UINT32** Command, **UINT32** Value, **UINT32** *Status)
Perform a CPU mailbox write.
- typedef **UINT32**(* **SA_GET_MEMORY_VDD**) (**VOID** *GlobalData, **UINT32** DefaultVdd)
Get the current memory voltage (VDD).
- typedef **UINT32**(* **SA_SET_MEMORY_VDD**) (**VOID** *GlobalData, **UINT32** DefaultVdd, **UINT32** Value)
Set the memory voltage (VDD) to the given value.
- typedef **UINT32**(* **SA_SET_MEMORY_PMIC_VOLTAGE**) (**VOID** *GlobalData, **UINT8** SpdAddress, **UINT32** Vdd, **UINT32** Vddq, **UINT32** Vpp)
Set DDR5 memory voltages (VDD, VDDQ, VPP) to the given values.
- typedef **UINT32**(* **SA_CHECKPOINT**) (**VOID** *GlobalData, **UINT32** CheckPoint, **VOID** *Scratch)
Check point that is called at various points in the MRC.
- typedef **VOID**(* **SA_DEBUG_HOOK**) (**VOID** *GlobalData, **UINT16** DisplayDebugNumber)
Typically used to display to the I/O port 80h.
- typedef **UINT8**(* **SA_CHANNEL_EXIST**) (**VOID** *Outputs, **UINT8** Channel)
Returns whether Channel is or is not present.
- typedef **INT32**(* **SA_PRINTF**) (**VOID** *Debug, **UINT32** Level, char *Format,...)
Print to output stream/device.
- typedef **VOID**(* **SA_DEBUG_PRINT**) (**VOID** *String)
Output a string to the debug stream/device.
- typedef **UINT32**(* **SA_CHANGE_MARGIN**) (**VOID** *GlobalData, **UINT8** Param, **INT32** Value0, **INT32** Value1, **UINT8** EnMultiCast, **UINT8** Channel, **UINT8** RankIn, **UINT8** Byte, **UINT8** BitIn, **UINT8** UpdateMrcData, **UINT8** SkipWait, **UINT32** RegFileParam)
Change the margin.
- typedef **UINT8**(* **SA_SIGN_EXTEND**) (**UINT8** Value, **UINT8** OldMsb, **UINT8** NewMsb)
Sign extends OldMSB to NewMSB Bits (Eg: Bit 6 to Bit 7).
- typedef **VOID**(* **SA_SHIFT_PI_COMMAND_TRAIN**) (**VOID** *GlobalData, **UINT8** Channel, **UINT8** Iteration, **UINT8** RankMask, **UINT8** GroupMask, **INT32** NewValue, **UINT8** UpdateHost)
Move CMD/CTL/CLK/CKE PIs during training.

- typedef [VOID](#)(* [SA_UPDATE_VREF](#)) ([VOID](#) *GlobalData, [UINT8](#) Channel, [UINT8](#) RankMask, [UINT16](#) DeviceMask, [UINT8](#) VrefType, [INT32](#) Offset, [BOOLEAN](#) UpdateMrcData, [BOOLEAN](#) PDAmode, [BOOLEAN](#) SkipWait)
Update the Vref value and wait until it is stable.
- typedef [UINT8](#)(* [SA_GET_RTC_CMOS](#)) ([UINT8](#) Location)
Get the current value of the specified RTC CMOS location.
- typedef [UINT64](#)(* [SA_MSR_READ_64](#)) ([UINT32](#) Location)
Get the current value of the specified MSR location.
- typedef [UINT64](#)(* [SA_MSR_WRITE_64](#)) ([UINT32](#) Location, [UINT64](#) Data)
Set the current value of the specified MSR location.
- typedef [VOID](#)(* [SA_MRC_RETURN_FROM_SMC](#)) ([VOID](#) *GlobalData, [UINT32](#) MrcStatus)
Hook function after returning from MrcStartMemoryConfiguration()
- typedef [VOID](#)(* [SA_MRC_DRAM_RESET](#)) ([UINT32](#) PciEBaseAddress, [UINT32](#) ResetValue)
Assert or deassert DRAM_RESET# pin; this is used in JEDEC Reset.
- typedef [VOID](#)(* [SA_DELAY_NS](#)) ([VOID](#) *GlobalData, [UINT32](#) DelayNs)
Delay (stall) for the given amount of nanoseconds.

Enumerations

- enum [SA_SPD](#)
SA SPD profile selections.
- enum [SPD_BOOT_MODE](#)
Define the boot modes used by the SPD read function.

14.42.1 Detailed Description

Policy definition of Memory Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2014 - 2023 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.42.2 Enumeration Type Documentation

14.42.2.1 SA_SPD

enum [SA_SPD](#)

SA SPD profile selections.

Enumerator

DEFAULT_SPD	Standard DIMM profile select.
CUSTOM_PROFILE	XMP enthusiast settings: User specifies various override values.
XMP_PROFILE_1	XMP enthusiast settings: XMP profile #1.
XMP_PROFILE_2	XMP enthusiast settings: XMP profile #2.
XMP_PROFILE_3	XMP enthusiast settings: XMP profile #3.
XMP_USER_PROFILE↔ _4	XMP enthusiast settings: XMP user profile #4.
XMP_USER_PROFILE↔ _5	XMP enthusiast settings: XMP user profile #5.
SPDProfileMax	Ensures SA_SPD is UINT8.

Definition at line 118 of file MemoryConfig.h.

14.42.2.2 SPD_BOOT_MODE

enum [SPD_BOOT_MODE](#)

Define the boot modes used by the SPD read function.

Enumerator

SpdCold	Cold boot.
SpdWarm	Warm boot.
SpdS3	S3 resume.
SpdFast	Fast boot.
SpdBootModeMax	Delimiter.

Definition at line 132 of file MemoryConfig.h.

14.43 MePeiConfig.h File Reference

ME config block for PEI phase.

Classes

- struct [ME_PEI_PREMEM_CONFIG](#)
ME Pei Pre-Memory Configuration Structure.
- struct [ME_PEI_CONFIG](#)
ME Pei Post-Memory Configuration Structure.

14.43.1 Detailed Description

ME config block for PEI phase.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

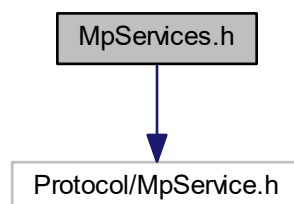
Specification Reference:

14.44 MpServices.h File Reference

This file declares UEFI PI Multi-processor PPI.

```
#include <Protocol/MpService.h>
```

Include dependency graph for MpServices.h:



Classes

- struct [_EFI_PEI_MP_SERVICES_PPI](#)

This PPI is installed by some platform or chipset-specific PEIM that abstracts handling multiprocessor support.

Typedefs

- typedef [EFI_STATUS](#)(* [EFI_PEI_MP_SERVICES_GET_NUMBER_OF_PROCESSORS](#)) (IN CONST [EFI_PEI_SERVICES](#) **PeiServices, IN [EFI_PEI_MP_SERVICES_PPI](#) *This, OUT UINTN *NumberOfProcessors, OUT UINTN *NumberOfEnabledProcessors)
Get the number of CPU's.
- typedef [EFI_STATUS](#)(* [EFI_PEI_MP_SERVICES_GET_PROCESSOR_INFO](#)) (IN CONST [EFI_PEI_SERVICES](#) **PeiServices, IN [EFI_PEI_MP_SERVICES_PPI](#) *This, IN UINTN ProcessorNumber, OUT [EFI_PROCESSOR_INFORMATION](#) *ProcessorInfoBuffer)
Get information on a specific CPU.
- typedef [EFI_STATUS](#)(* [EFI_PEI_MP_SERVICES_STARTUP_ALL_APS](#)) (IN CONST [EFI_PEI_SERVICES](#) **PeiServices, IN [EFI_PEI_MP_SERVICES_PPI](#) *This, IN [EFI_AP_PROCEDURE](#) Procedure, IN BOOLEAN SingleThread, IN UINTN TimeoutInMicroSeconds, IN VOID *ProcedureArgument [OPTIONAL](#))
Activate all of the application processors.
- typedef [EFI_STATUS](#)(* [EFI_PEI_MP_SERVICES_STARTUP_THIS_AP](#)) (IN CONST [EFI_PEI_SERVICES](#) **PeiServices, IN [EFI_PEI_MP_SERVICES_PPI](#) *This, IN [EFI_AP_PROCEDURE](#) Procedure, IN UINTN ProcessorNumber, IN UINTN TimeoutInMicroSeconds, IN VOID *ProcedureArgument [OPTIONAL](#))
Activate a specific application processor.
- typedef [EFI_STATUS](#)(* [EFI_PEI_MP_SERVICES_SWITCH_BSP](#)) (IN CONST [EFI_PEI_SERVICES](#) **PeiServices, IN [EFI_PEI_MP_SERVICES_PPI](#) *This, IN UINTN ProcessorNumber, IN BOOLEAN EnableOldBSP)
Switch the boot strap processor.
- typedef [EFI_STATUS](#)(* [EFI_PEI_MP_SERVICES_ENABLEDISABLEAP](#)) (IN CONST [EFI_PEI_SERVICES](#) **PeiServices, IN [EFI_PEI_MP_SERVICES_PPI](#) *This, IN UINTN ProcessorNumber, IN BOOLEAN EnableAP, IN UINT32 *HealthFlag [OPTIONAL](#))
Enable or disable an application processor.
- typedef [EFI_STATUS](#)(* [EFI_PEI_MP_SERVICES_WHOAMI](#)) (IN CONST [EFI_PEI_SERVICES](#) **PeiServices, IN [EFI_PEI_MP_SERVICES_PPI](#) *This, OUT UINTN *ProcessorNumber)
Identify the currently executing processor.

14.44.1 Detailed Description

This file declares UEFI PI Multi-processor PPI.

This PPI is installed by some platform or chipset-specific PEIM that abstracts handling multiprocessor support.

Copyright (c) 2015 - 2017, Intel Corporation. All rights reserved.
SPDX-License-Identifier: BSD-2-Clause-Patent

Revision Reference:

This PPI is introduced in PI Version 1.4.

14.44.2 Typedef Documentation

14.44.2.1 EFI_PEI_MP_SERVICES_ENABLEDISABLEAP

```
typedef EFI_STATUS ( * EFI_PEI_MP_SERVICES_ENABLEDISABLEAP ) ( IN CONST EFI_PEI_SERVICES **PeiServices, IN EFI_PEI_MP_SERVICES_PPI *This, IN UINTN ProcessorNumber, IN BOOLEAN EnableAP, IN  
UINT32 *HealthFlag OPTIONAL)
```

Enable or disable an application processor.

Parameters

in	<i>PeiServices</i>	An indirect pointer to the PEI Services Table published by the PEI Foundation.
in	<i>This</i>	A pointer to the EFI_PEI_MP_SERVICES_PPI instance.
in	<i>ProcessorNumber</i>	The handle number of the AP. The range is from 0 to the total number of logical processors minus 1. The total number of logical processors can be retrieved by <code>EFI_PEI_MP_SERVICES_PPI.GetNumberOfProcessors()</code> .
in	<i>EnableAP</i>	Specifies the new state for the processor for enabled, FALSE for disabled.
in	<i>HealthFlag</i>	If not NULL, a pointer to a value that specifies the new health status of the AP. This flag corresponds to <code>StatusFlag</code> defined in <code>EFI_PEI_MP_SERVICES_PPI.GetProcessorInfo()</code> . Only the <code>PROCESSOR_HEALTH_STATUS_BIT</code> is used. All other bits are ignored. If it is NULL, this parameter is ignored.

Return values

<i>EFI_SUCCESS</i>	The specified AP was enabled or disabled successfully.
<i>EFI_UNSUPPORTED</i>	Enabling or disabling an AP cannot be completed prior to this service returning.
<i>EFI_UNSUPPORTED</i>	Enabling or disabling an AP is not supported.
<i>EFI_DEVICE_ERROR</i>	The calling processor is an AP.
<i>EFI_NOT_FOUND</i>	Processor with the handle specified by <code>ProcessorNumber</code> does not exist.
<i>EFI_INVALID_PARAMETER</i>	<code>ProcessorNumber</code> specifies the BSP.

Definition at line 229 of file MpServices.h.

14.44.2.2 EFI_PEI_MP_SERVICES_GET_NUMBER_OF_PROCESSORS

```
typedef EFI_STATUS( * EFI_PEI_MP_SERVICES_GET_NUMBER_OF_PROCESSORS)(IN CONST EFI_PEI_SERVICES **PeiServices, IN EFI_PEI_MP_SERVICES_PPI *This, OUT UINTN *NumberOfProcessors, OUT UINTN *NumberOfEnabledProcessors)
```

Get the number of CPU's.

Parameters

in	<i>PeiServices</i>	An indirect pointer to the PEI Services Table published by the PEI Foundation.
in	<i>This</i>	Pointer to this instance of the PPI.
out	<i>NumberOfProcessors</i>	Pointer to the total number of logical processors in the system, including the BSP and disabled APs.
out	<i>NumberOfEnabledProcessors</i>	Number of processors in the system that are enabled.

Return values

<i>EFI_SUCCESS</i>	The number of logical processors and enabled logical processors was retrieved.
<i>EFI_DEVICE_ERROR</i>	The calling processor is an AP.
<i>EFI_INVALID_PARAMETER</i>	<code>NumberOfProcessors</code> is NULL. <code>NumberOfEnabledProcessors</code> is NULL.

Definition at line 44 of file MpServices.h.

14.44.2.3 EFI_PEI_MP_SERVICES_GET_PROCESSOR_INFO

```
typedef EFI_STATUS( * EFI_PEI_MP_SERVICES_GET_PROCESSOR_INFO) (IN CONST EFI_PEI_SERVICES **PeiServices, IN EFI_PEI_MP_SERVICES_PPI *This, IN UINTN ProcessorNumber, OUT EFI_PROCESSOR_INFORMATION *ProcessorInfoBuffer)
```

Get information on a specific CPU.

Parameters

in	<i>PeiServices</i>	An indirect pointer to the PEI Services Table published by the PEI Foundation.
in	<i>This</i>	Pointer to this instance of the PPI.
in	<i>ProcessorNumber</i>	Pointer to the total number of logical processors in the system, including the BSP and disabled APs.
out	<i>ProcessorInfoBuffer</i>	Number of processors in the system that are enabled.

Return values

<i>EFI_SUCCESS</i>	Processor information was returned.
<i>EFI_DEVICE_ERROR</i>	The calling processor is an AP.
<i>EFI_INVALID_PARAMETER</i>	ProcessorInfoBuffer is NULL.
<i>EFI_NOT_FOUND</i>	The processor with the handle specified by ProcessorNumber does not exist in the platform.

Definition at line 69 of file MpServices.h.

14.44.2.4 EFI_PEI_MP_SERVICES_STARTUP_ALL_APS

```
typedef EFI_STATUS( * EFI_PEI_MP_SERVICES_STARTUP_ALL_APS) (IN CONST EFI_PEI_SERVICES **PeiServices, IN EFI_PEI_MP_SERVICES_PPI *This, IN EFI_AP_PROCEDURE Procedure, IN BOOLEAN SingleThread, IN UINTN TimeoutInMicroSeconds, IN VOID *ProcedureArgument OPTIONAL)
```

Activate all of the application processors.

Parameters

in	<i>PeiServices</i>	An indirect pointer to the PEI Services Table published by the PEI Foundation.
in	<i>This</i>	A pointer to the EFI_PEI_MP_SERVICES_PPI instance.
in	<i>Procedure</i>	A pointer to the function to be run on enabled APs of the system.
in	<i>SingleThread</i>	If TRUE, then all the enabled APs execute the function specified by Procedure one by one, in ascending order of processor handle number. If FALSE, then all the enabled APs execute the function specified by Procedure simultaneously.

Parameters

in	<i>TimeoutInMicroSeconds</i>	Indicates the time limit in microseconds for APs to return from Procedure, for blocking mode only. Zero means infinity. If the timeout expires before all APs return from Procedure, then Procedure on the failed APs is terminated. All enabled APs are available for next function assigned by <code>EFI_PEI_MP_SERVICES_PPI.StartupAllAPs()</code> or <code>EFI_PEI_MP_SERVICES_PPI.StartupThisAP()</code> . If the timeout expires in blocking mode, BSP returns <code>EFI_TIMEOUT</code> .
in	<i>ProcedureArgument</i>	The parameter passed into Procedure for all APs.

Return values

<i>EFI_SUCCESS</i>	In blocking mode, all APs have finished before the timeout expired.
<i>EFI_DEVICE_ERROR</i>	Caller processor is AP.
<i>EFI_NOT_STARTED</i>	No enabled APs exist in the system.
<i>EFI_NOT_READY</i>	Any enabled APs are busy.
<i>EFI_TIMEOUT</i>	In blocking mode, the timeout expired before all enabled APs have finished.
<i>EFI_INVALID_PARAMETER</i>	Procedure is NULL.

Definition at line 112 of file MpServices.h.

14.44.2.5 EFI_PEI_MP_SERVICES_STARTUP_THIS_AP

```
typedef EFI_STATUS( * EFI_PEI_MP_SERVICES_STARTUP_THIS_AP)(IN CONST EFI_PEI_SERVICES **PeiServices, IN EFI_PEI_MP_SERVICES_PPI *This, IN EFI_AP_PROCEDURE Procedure, IN UINTN ProcessorNumber, IN UINTN TimeoutInMicroSeconds, IN VOID *ProcedureArgument OPTIONAL)
```

Activate a specific application processor.

Parameters

in	<i>PeiServices</i>	An indirect pointer to the PEI Services Table published by the PEI Foundation.
in	<i>This</i>	A pointer to the <code>EFI_PEI_MP_SERVICES_PPI</code> instance.
in	<i>Procedure</i>	A pointer to the function to be run on enabled APs of the system.
in	<i>ProcessorNumber</i>	The handle number of the AP. The range is from 0 to the total number of logical processors minus 1. The total number of logical processors can be retrieved by <code>EFI_PEI_MP_SERVICES_PPI.GetNumberOfProcessors()</code> .
in	<i>TimeoutInMicroSeconds</i>	Indicates the time limit in microseconds for APs to return from Procedure, for blocking mode only. Zero means infinity. If the timeout expires before all APs return from Procedure, then Procedure on the failed APs is terminated. All enabled APs are available for next function assigned by <code>EFI_PEI_MP_SERVICES_PPI.StartupAllAPs()</code> or <code>EFI_PEI_MP_SERVICES_PPI.StartupThisAP()</code> . If the timeout expires in blocking mode, BSP returns <code>EFI_TIMEOUT</code> .
in	<i>ProcedureArgument</i>	The parameter passed into Procedure for all APs.

Return values

<i>EFI_SUCCESS</i>	In blocking mode, specified AP finished before the timeout expires.
<i>EFI_DEVICE_ERROR</i>	The calling processor is an AP.
<i>EFI_TIMEOUT</i>	In blocking mode, the timeout expired before the specified AP has finished.
<i>EFI_NOT_FOUND</i>	The processor with the handle specified by ProcessorNumber does not exist.
<i>EFI_INVALID_PARAMETER</i>	ProcessorNumber specifies the BSP or disabled AP.
<i>EFI_INVALID_PARAMETER</i>	Procedure is NULL.

Definition at line 157 of file MpServices.h.

14.44.2.6 EFI_PEI_MP_SERVICES_SWITCH_BSP

```
typedef EFI_STATUS( * EFI_PEI_MP_SERVICES_SWITCH_BSP) (IN CONST EFI_PEI_SERVICES **PeiServices,
IN EFI_PEI_MP_SERVICES_PPI *This, IN UINTN ProcessorNumber, IN BOOLEAN EnableOldBSP)
```

Switch the boot strap processor.

Parameters

in	<i>PeiServices</i>	An indirect pointer to the PEI Services Table published by the PEI Foundation.
in	<i>This</i>	A pointer to the EFI_PEI_MP_SERVICES_PPI instance.
in	<i>ProcessorNumber</i>	The handle number of the AP. The range is from 0 to the total number of logical processors minus 1. The total number of logical processors can be retrieved by EFI_PEI_MP_SERVICES_PPI.GetNumberOfProcessors().
in	<i>EnableOldBSP</i>	If TRUE, then the old BSP will be listed as an enabled AP. Otherwise, it will be disabled.

Return values

<i>EFI_SUCCESS</i>	BSP successfully switched.
<i>EFI_UNSUPPORTED</i>	Switching the BSP cannot be completed prior to this service returning.
<i>EFI_UNSUPPORTED</i>	Switching the BSP is not supported.
<i>EFI_DEVICE_ERROR</i>	The calling processor is an AP.
<i>EFI_NOT_FOUND</i>	The processor with the handle specified by ProcessorNumber does not exist.
<i>EFI_INVALID_PARAMETER</i>	ProcessorNumber specifies the current BSP or a disabled AP.
<i>EFI_NOT_READY</i>	The specified AP is busy.

Definition at line 192 of file MpServices.h.

14.44.2.7 EFI_PEI_MP_SERVICES_WHOAMI

```
typedef EFI_STATUS( * EFI_PEI_MP_SERVICES_WHOAMI) (IN CONST EFI_PEI_SERVICES **PeiServices, IN
EFI_PEI_MP_SERVICES_PPI *This, OUT UINTN *ProcessorNumber)
```

Identify the currently executing processor.

Parameters

in	<i>PeiServices</i>	An indirect pointer to the PEI Services Table published by the PEI Foundation.
in	<i>This</i>	A pointer to the EFI_PEI_MP_SERVICES_PPI instance.
out	<i>ProcessorNumber</i>	The handle number of the AP. The range is from 0 to the total number of logical processors minus 1. The total number of logical processors can be retrieved by <code>EFI_PEI_MP_SERVICES_PPI.GetNumberOfProcessors()</code> .

Return values

<i>EFI_SUCCESS</i>	The current processor handle number was returned in <i>ProcessorNumber</i> .
<i>EFI_INVALID_PARAMETER</i>	<i>ProcessorNumber</i> is NULL.

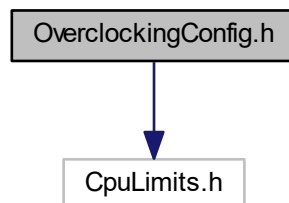
Definition at line 254 of file MpServices.h.

14.45 OverclockingConfig.h File Reference

Overclocking Config Block.

```
#include <CpuLimits.h>
```

Include dependency graph for OverclockingConfig.h:



Classes

- struct [OVERCLOCKING_PREMEM_CONFIG](#)
Overclocking Configuration Structure.

14.45.1 Detailed Description

Overclocking Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2018 - 2022 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.46 P2sbConfig.h File Reference

P2sb policy.

Classes

- struct [PCH_P2SB_CONFIG](#)

This structure contains the policies which are related to P2SB device.

14.46.1 Detailed Description

P2sb policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.47 PchDmiConfig.h File Reference

DMI policy.

Classes

- struct [PCH_DMI_EQ_PARAM](#)
Lane specific Dmi Gen3, Gen4 equalization parameters.
- struct [PCH_DMI_CONFIG](#)
The [PCH_DMI_CONFIG](#) block describes the expected configuration of the PCH for DMI.

14.47.1 Detailed Description

DMI policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.48 PchGeneralConfig.h File Reference

PCH General policy.

Classes

- struct [PCH_GENERAL_CONFIG](#)
*PCH General Configuration **Revision 1**: - Initial version.*
- struct [PCH_GENERAL_PREMEM_CONFIG](#)
*PCH General Pre-Memory Configuration **Revision 1**: - Initial version.*

Enumerations

- enum [PCH_RESERVED_PAGE_ROUTE](#)

14.48.1 Detailed Description

PCH General policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.48.2 Enumeration Type Documentation

14.48.2.1 PCH_RESERVED_PAGE_ROUTE

enum [PCH_RESERVED_PAGE_ROUTE](#)

Enumerator

PchReservedPageToLpc	Port 80h cycles are sent to LPC.
PchReservedPageToPcie	Port 80h cycles are sent to PCIe.

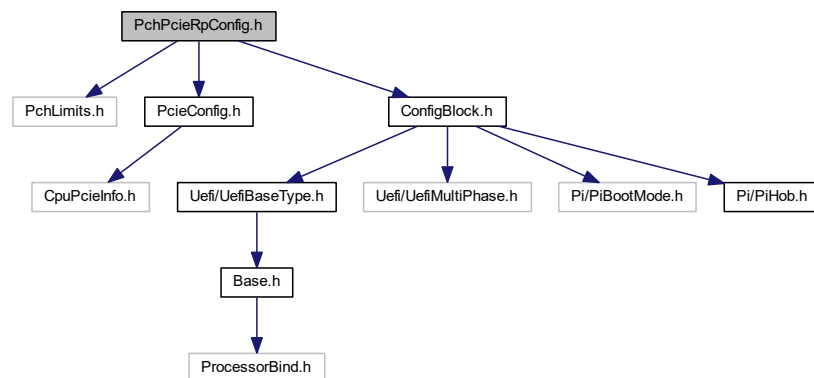
Definition at line 46 of file PchGeneralConfig.h.

14.49 PchPcieRpConfig.h File Reference

PCH Pcie root port policy.

```
#include <PchLimits.h>
#include <PcieConfig.h>
#include <ConfigBlock.h>
```

Include dependency graph for PchPcieRpConfig.h:



Classes

- struct [PCH_PCIE_CLOCK](#)
[PCH_PCIE_CLOCK](#) describes PCIe source clock generated by PCH.
- struct [PCH_PCIE_ROOT_PORT_CONFIG](#)
The [PCH_PCIE_ROOT_PORT_CONFIG](#) describe the feature and capability of each PCH PCIe root port.
- struct [PCH_PCIE_CONFIG](#)
*The [PCH_PCIE_CONFIG](#) block describes the expected configuration of the PCH PCI Express controllers **Revision 1**:*
- struct [PCH_PCIE_RP_PREMEM_CONFIG](#)
*The [PCH_PCIE_RP_PREMEM_CONFIG](#) block describes early configuration of the PCH PCI Express controllers **Revision 1**:*
- struct [PCIE_RP_DXE_CONFIG](#)
The [PCIE_RP_DXE_CONFIG](#) block describes the expected configuration of the PCH PCI Express controllers in DXE phase.

Enumerations

- enum [PCH_PCIE_ASPM_CONTROL](#)
The values before AutoConfig match the setting of PCI Express Base Specification 1.1, please be careful for adding new feature.
- enum [PCH_PCIE_L1SUBSTATES_CONTROL](#)
Refer to PCH EDS for the PCH implementation values corresponding to below PCI-E spec defined ranges.
- enum [PCH_PCIE_CLOCK_USAGE](#)

14.49.1 Detailed Description

PCH Pcie root port policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.49.2 Enumeration Type Documentation

14.49.2.1 PCH_PCIE_CLOCK_USAGE

enum [PCH_PCIE_CLOCK_USAGE](#)

Enumerator

PchClockUsageCpuPcie0	Quantity of PCH and CPU PCIe ports, as well as their encoding in this enum, may change between silicon generations and series. Do not assume that PCH port 0 will be always encoded by 0. Instead, it is recommended to use (PchClockUsagePchPcie0 + PchPortIndex) style to be forward-compatible
PchClockUsageUnspecified	In use for a purpose not listed above.

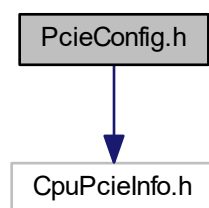
Definition at line 134 of file PchPcieRpConfig.h.

14.50 PcieConfig.h File Reference

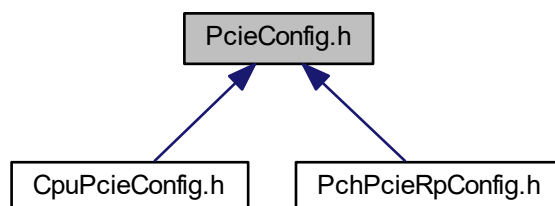
PCIe Config Block.

```
#include <CpuPcieInfo.h>
```

Include dependency graph for PcieConfig.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [PCIE_EQ_PARAM](#)
Represent lane specific PCIe Gen3 equalization parameters.
- struct [PCIE_LINK_EQ_PLATFORM_SETTINGS](#)
PCIe Link EQ Platform Settings.
- struct [PCIE_COMMON_CONFIG](#)
PCIe Common Config.
- struct [PCIE_DEVICE_OVERRIDE](#)
PCIe device table entry entry.

Enumerations

- enum [PCIE_FORM_FACTOR](#)
Specifies the form factor that the slot implements.
- enum [PCIE_LINK_EQ_METHOD](#)
- enum [PCIE_LINK_EQ_MODE](#)

14.50.1 Detailed Description

PCIe Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2018 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.50.2 Enumeration Type Documentation

14.50.2.1 PCIE_FORM_FACTOR

enum [PCIE_FORM_FACTOR](#)

Specifies the form factor that the slot implements.

For custom form factors that do not require any special handling please set PcieFormFactorOther.

Definition at line 103 of file PcieConfig.h.

14.50.2.2 PCIE_LINK_EQ_METHOD

enum [PCIE_LINK_EQ_METHOD](#)

Enumerator

PcieLinkHardwareEq	Hardware is responsible for performing coefficient/preset search.
PcieLinkFixedEq	No coefficient/preset search is performed. Fixed values are used.

Definition at line 114 of file PcieConfig.h.

14.50.2.3 PCIE_LINK_EQ_MODE

enum [PCIE_LINK_EQ_MODE](#)

Enumerator

PcieLinkEqPresetMode	Use presets during PCIe link equalization.
PcieLinkEqCoefficientMode	Use coefficients during PCIe link equalization.

Definition at line 119 of file PcieConfig.h.

14.51 PciePreMemConfig.h File Reference

PCIe Config Block PreMem.

Classes

- struct [PCIE_IMR_CONFIG](#)
PCIe IMR Config.
- struct [PCIE_PREMEM_CONFIG](#)
*PCIe Pre-Memory Configuration **Revision 1:** - Initial version.*

14.51.1 Detailed Description

PCIe Config Block PreMem.

Copyright

INTEL CONFIDENTIAL Copyright 2018 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

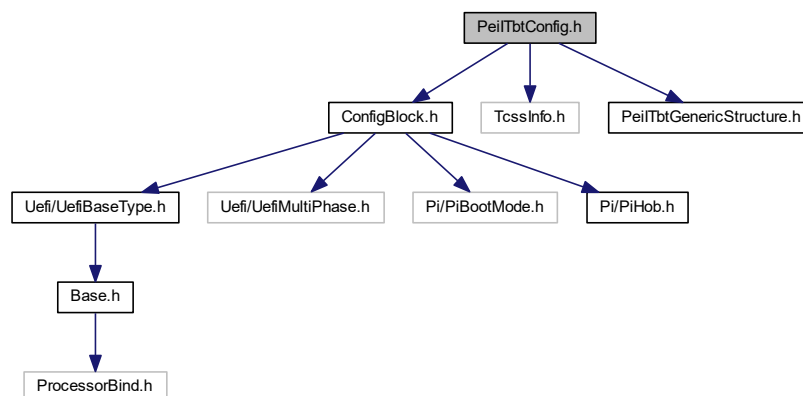
Specification Reference:

14.52 PeiTbtConfig.h File Reference

Header file for TBT PEI Policy.

```
#include <ConfigBlock.h>
#include <TcssInfo.h>
#include <PeiTbtGenericStructure.h>
```

Include dependency graph for PeiTbtConfig.h:



Classes

- struct [_PEI_ITBT_CONFIG](#)
ITBT PEI configuration
Revision 1:

Typedefs

- typedef struct [_PEI_ITBT_CONFIG](#) [PEI_ITBT_CONFIG](#)
ITBT PEI configuration
Revision 1:

14.52.1 Detailed Description

Header file for TBT PEI Policy.

Copyright

INTEL CONFIDENTIAL Copyright 2017 - 2019 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.52.2 Typedef Documentation

14.52.2.1 PEI_ITBT_CONFIG

```
typedef struct _PEI_ITBT_CONFIG PEI_ITBT_CONFIG
```

ITBT PEI configuration

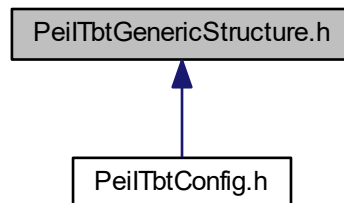
Revision 1:

- Initial version. **Revision 2:**
- Add ITbtPcieTunnelingForUsb4, deprecated ITbtSecurityLevel.

14.53 PeiTbtGenericStructure.h File Reference

ITBT Policy definition to be referred in both PEI and DXE phase.

This graph shows which files directly or indirectly include this file:



Classes

- struct [_ITBT_ROOTPORT_CONFIG](#)
iTBT RootPort Data Structure
- struct [_ITBT_GENERIC_CONFIG](#)
ITBT Controller Data Structure to be used cross to RP and controller to be shared by CONFIG_BLOCK and HOB.

Typedefs

- typedef struct [_ITBT_ROOTPORT_CONFIG](#) [ITBT_ROOTPORT_CONFIG](#)
iTBT RootPort Data Structure
- typedef struct [_ITBT_GENERIC_CONFIG](#) [ITBT_GENERIC_CONFIG](#)
ITBT Controller Data Structure to be used cross to RP and controller to be shared by CONFIG_BLOCK and HOB.

14.53.1 Detailed Description

ITBT Policy definition to be referred in both PEI and DXE phase.

Copyright

INTEL CONFIDENTIAL Copyright 2017 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.54 PeiPreMemSiDefaultPolicy.h File Reference

This file defines the function to initialize default silicon policy PPI.

Classes

- struct [_PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI](#)

This PPI provides function to install default silicon policy.

Typedefs

- typedef [EFI_STATUS](#)(* [PEI_PREMEM_POLICY_INIT](#)) ([VOID](#))

Initialize and install default silicon policy PPI.

14.54.1 Detailed Description

This file defines the function to initialize default silicon policy PPI.

Copyright

INTEL CONFIDENTIAL Copyright 2019 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.55 PeiSiDefaultPolicy.h File Reference

This file defines the function to initialize default silicon policy PPI.

Classes

- struct [_PEI_SI_DEFAULT_POLICY_INIT_PPI](#)

This PPI provides function to install default silicon policy.

Typedefs

- typedef [EFI_STATUS](#)(* [PEI_POLICY_INIT](#)) ([VOID](#))

Initialize and install default silicon policy PPI.

14.55.1 Detailed Description

This file defines the function to initialize default silicon policy PPI.

Copyright

INTEL CONFIDENTIAL Copyright 2019 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.56 PiHob.h File Reference

HOB related definitions in PI.

This graph shows which files directly or indirectly include this file:



Classes

- struct [EFI_HOB_GENERIC_HEADER](#)
Describes the format and size of the data inside the HOB.
- struct [EFI_HOB_HANDOFF_INFO_TABLE](#)
Contains general state information used by the HOB producer phase.
- struct [EFI_HOB_MEMORY_ALLOCATION_HEADER](#)
[EFI_HOB_MEMORY_ALLOCATION_HEADER](#) describes the various attributes of the logical memory allocation.
- struct [EFI_HOB_MEMORY_ALLOCATION](#)
Describes all memory ranges used during the HOB producer phase that exist outside the HOB list.
- struct [EFI_HOB_MEMORY_ALLOCATION_STACK](#)

Describes the memory stack that is produced by the HOB producer phase and upon which all post-memory-installed executable content in the HOB producer phase is executing.

- struct [EFI_HOB_MEMORY_ALLOCATION_BSP_STORE](#)

Defines the location of the boot-strap processor (BSP) BSPStore ("Backing Store Pointer Store").

- struct [EFI_HOB_MEMORY_ALLOCATION_MODULE](#)

Defines the location and entry point of the HOB consumer phase.

- struct [EFI_HOB_RESOURCE_DESCRIPTOR](#)

Describes the resource properties of all fixed, nonrelocatable resource ranges found on the processor host bus during the HOB producer phase.

- struct [EFI_HOB_GUID_TYPE](#)

Allows writers of executable content in the HOB producer phase to maintain and manage HOBs with specific [GUID](#).

- struct [EFI_HOB_FIRMWARE_VOLUME](#)

Details the location of firmware volumes that contain firmware files.

- struct [EFI_HOB_FIRMWARE_VOLUME2](#)

Details the location of a firmware volume that was extracted from a file within another firmware volume.

- struct [EFI_HOB_FIRMWARE_VOLUME3](#)

Details the location of a firmware volume that was extracted from a file within another firmware volume.

- struct [EFI_HOB_CPU](#)

Describes processor information, such as address space and I/O space capabilities.

- struct [EFI_HOB_MEMORY_POOL](#)

Describes pool memory allocations.

- struct [EFI_HOB_UEFI_CAPSULE](#)

Each UEFI capsule HOB details the location of a UEFI capsule.

- union [EFI_PEI_HOB_POINTERS](#)

Union of all the possible HOB Types.

Macros

- #define [EFI_HOB_HANDOFF_TABLE_VERSION](#) 0x0009

Value of version in [EFI_HOB_HANDOFF_INFO_TABLE](#).

Typedefs

- typedef UINT32 [EFI_RESOURCE_TYPE](#)

The resource type.

- typedef UINT32 [EFI_RESOURCE_ATTRIBUTE_TYPE](#)

A type of recount attribute type.

14.56.1 Detailed Description

HOB related definitions in PI.

Copyright (c) 2006 - 2017, Intel Corporation. All rights reserved.

SPDX-License-Identifier: BSD-2-Clause-Patent

Revision Reference:

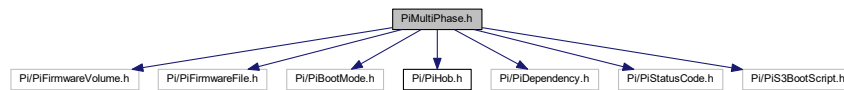
PI Version 1.6

14.57 PiMultiPhase.h File Reference

Include file matches things in PI for multiple module types.

```
#include <Pi/PiFirmwareVolume.h>
#include <Pi/PiFirmwareFile.h>
#include <Pi/PiBootMode.h>
#include <Pi/PiHob.h>
#include <Pi/PiDependency.h>
#include <Pi/PiStatusCode.h>
#include <Pi/PiS3BootScript.h>
```

Include dependency graph for PiMultiPhase.h:



Classes

- struct [EFI_MMram_DESCRIPTOR](#)
Structure describing a MMram region and its accessibility attributes.
- struct [_EFI_MM_RESERVED_MMram_REGION](#)
Structure describing a MMram region which cannot be used for the MMram heap.

Macros

- #define [DXE_ERROR](#)(StatusCode) (MAX_BIT | (MAX_BIT >> 2) | StatusCode)
Produces an error code in the range reserved for use by the Platform Initialization Architecture Specification.
- #define [EFI_REQUEST_UNLOAD_IMAGE_DXE_ERROR](#) (1)
If this value is returned by an EFI image, then the image should be unloaded.
- #define [EFI_NOT_AVAILABLE_YET_DXE_ERROR](#) (2)
If this value is returned by an API, it means the capability is not yet installed/available/ready to use.
- #define [PI_ENCODE_WARNING](#)(a) ((MAX_BIT >> 2) | (a))
Success and warning codes reserved for use by PI.
- #define [PI_ENCODE_ERROR](#)(a) (MAX_BIT | (MAX_BIT >> 2) | (a))
Error codes reserved for use by PI.
- #define [EFI_INTERRUPT_PENDING_PI_ENCODE_ERROR](#) (0)
Return status codes defined in SMM CIS.
- #define [EFI_MMram_OPEN](#) 0x00000001
MMram states and capabilities.
- #define [EFI_AUTH_STATUS_PLATFORM_OVERRIDE](#) 0x01
Bitmask of values for Authentication Status.

Typedefs

- typedef struct [_EFI_MM_RESERVED_MMRAM_REGION](#) [EFI_MM_RESERVED_MMRAM_REGION](#)
Structure describing a MMRAM region which cannot be used for the MMRAM heap.
- typedef [VOID](#)(* [EFI_AP_PROCEDURE](#)) ([IN OUT VOID](#) *Buffer)
The function prototype for invoking a function on an Application Processor.
- typedef [EFI_STATUS](#)(* [EFI_AP_PROCEDURE2](#)) ([IN VOID](#) *ProcedureArgument)
The function prototype for invoking a function on an Application Processor.

14.57.1 Detailed Description

Include file matches things in PI for multiple module types.

Copyright (c) 2006 - 2018, Intel Corporation. All rights reserved.
 SPDX-License-Identifier: BSD-2-Clause-Patent

Revision Reference:

These elements are defined in UEFI Platform Initialization Specification 1.2.

14.57.2 Macro Definition Documentation

14.57.2.1 DXE_ERROR

```
#define DXE_ERROR(  
    StatusCode ) (MAX_BIT | (MAX_BIT >> 2) | StatusCode)
```

Produces an error code in the range reserved for use by the Platform Initialization Architecture Specification.

The supported 32-bit range is 0xA0000000-0xBFFFFFFF The supported 64-bit range is 0xA000000000000000-0xBFFFFFFF00000000

Parameters

<i>StatusCode</i>	The status code value to convert into a warning code. <i>StatusCode</i> must be in the range 0x00000000..0xFFFFFFFF.
-------------------	--

Returns

The value specified by *StatusCode* in the PI reserved range.

Definition at line 36 of file PiMultiPhase.h.

14.57.2.2 EFI_AUTH_STATUS_PLATFORM_OVERRIDE

```
#define EFI_AUTH_STATUS_PLATFORM_OVERRIDE 0x01
```

Bitmask of values for Authentication Status.

Authentication Status is returned from `EFI_GUIDED_SECTION_EXTRACTION_PROTOCOL` and the `EFI_PEI_GUIDED_SECTION_EXTRACTION_PPI`

xx00 Image was not signed. xxx1 Platform security policy override. Assumes the same meaning as 0010 (the image was signed, the signature was tested, and the signature passed authentication test). 0010 Image was signed, the signature was tested, and the signature passed authentication test. 0110 Image was signed and the signature was not tested. 1010 Image was signed, the signature was tested, and the signature failed the authentication test.

Definition at line 84 of file `PiMultiPhase.h`.

14.57.2.3 PI_ENCODE_ERROR

```
#define PI_ENCODE_ERROR(  
    a ) (MAX_BIT | (MAX_BIT >> 2) | (a))
```

Error codes reserved for use by PI.

Supported 32-bit range is 0xa0000000-0xbfffffff. Supported 64-bit range is 0xa000000000000000-0xbfffffffffffffff.

Definition at line 61 of file `PiMultiPhase.h`.

14.57.2.4 PI_ENCODE_WARNING

```
#define PI_ENCODE_WARNING(  
    a ) ((MAX_BIT >> 2) | (a))
```

Success and warning codes reserved for use by PI.

Supported 32-bit range is 0x20000000-0x3fffffff. Supported 64-bit range is 0x2000000000000000-0x3fffffffffffffff.

Definition at line 54 of file `PiMultiPhase.h`.

14.57.3 Typedef Documentation

14.57.3.1 EFI_AP_PROCEDURE

```
typedef VOID ( * EFI_AP_PROCEDURE) (IN OUT VOID *Buffer)
```

The function prototype for invoking a function on an Application Processor.

This definition is used by the UEFI MP Serices Protocol, and the PI SMM System Table.

Parameters

in, out	<i>Buffer</i>	The pointer to private data buffer.
---------	---------------	-------------------------------------

Definition at line 190 of file PiMultiPhase.h.

14.57.3.2 EFI_AP_PROCEDURE2

```
typedef EFI_STATUS( * EFI_AP_PROCEDURE2) (IN VOID *ProcedureArgument)
```

The function prototype for invoking a function on an Application Processor.

This definition is used by the UEFI MM MP Serices Protocol.

Parameters

in	<i>ProcedureArgument</i>	The pointer to private data buffer.
----	--------------------------	-------------------------------------

Return values

<i>EFI_SUCCESS</i>	Excutive the procedure successfully
--------------------	-------------------------------------

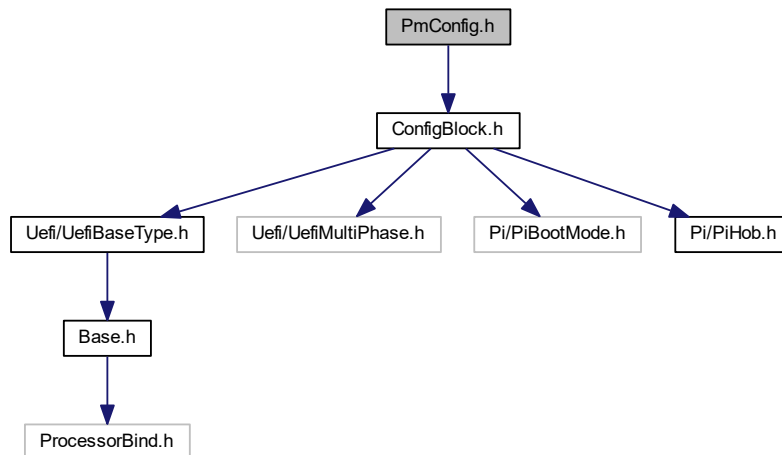
Definition at line 206 of file PiMultiPhase.h.

14.58 PmConfig.h File Reference

Power Management policy.

```
#include <ConfigBlock.h>
```

Include dependency graph for PmConfig.h:



Classes

- struct [PCH_WAKE_CONFIG](#)
This structure allows to customize PCH wake up capability from S5 or DeepSx by WOL, LAN, PCIE wake events.
- union [PMC_LPM_S0IX_SUB_STATE_EN](#)
Low Power Mode Enable config.
- union [PMC_GLOBAL_RESET_MASK](#)
Description of Global Reset Trigger/Event Mask register.
- struct [PCH_PM_CONFIG](#)
The [PCH_PM_CONFIG](#) block describes expected miscellaneous power management settings.

Enumerations

- enum [PCH_SLP_S4_MIN_ASSERT](#)

14.58.1 Detailed Description

Power Management policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2022 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.58.2 Enumeration Type Documentation

14.58.2.1 PCH_SLP_S4_MIN_ASSERT

enum [PCH_SLP_S4_MIN_ASSERT](#)

Enumerator

PchSlpS4PchTime	The time defined in PCH EDS Power Sequencing and Reset Signal Timings table.
-----------------	--

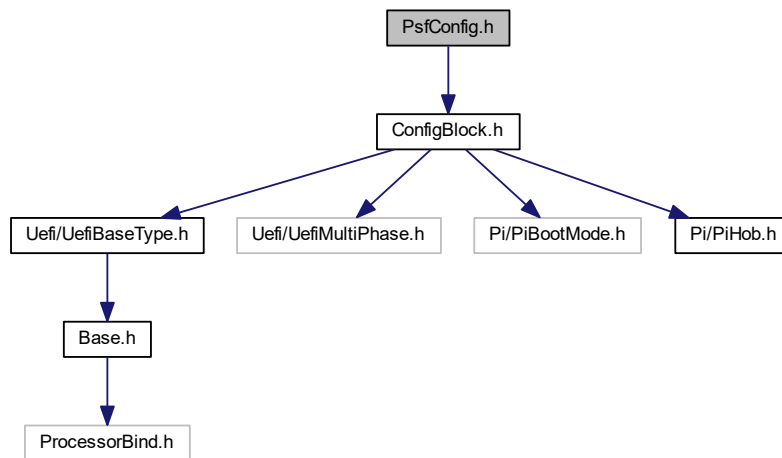
Definition at line 80 of file PmConfig.h.

14.59 PsfConfig.h File Reference

Primary Sideband Fabric policy.


```
#include <ConfigBlock.h>
```

Include dependency graph for PsfConfig.h:



Classes

- struct [PSF_CONFIG](#)

The [PSF_CONFIG](#) block describes the expected configuration of the Primary Sideband Fabric.

14.59.1 Detailed Description

Primary Sideband Fabric policy.

Copyright

INTEL CONFIDENTIAL Copyright 2018 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

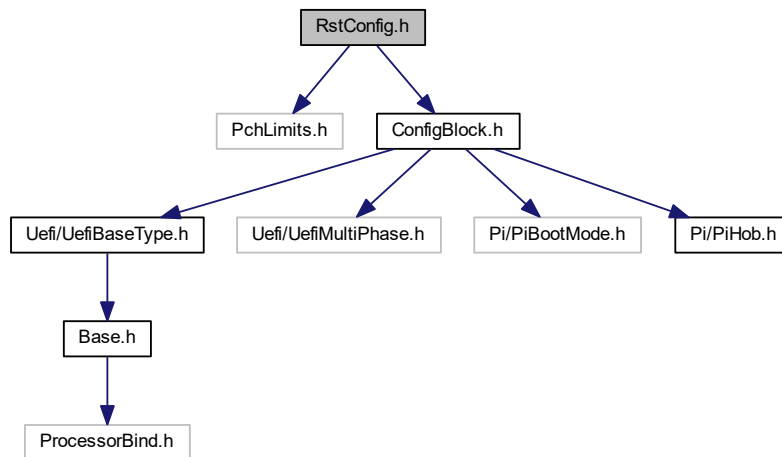
Specification Reference:

14.60 RstConfig.h File Reference

Rst policy.

```
#include <PchLimits.h>
#include <ConfigBlock.h>
```

Include dependency graph for RstConfig.h:



Classes

- struct [RST_HARDWARE_REMAPPED_STORAGE_CONFIG](#)

This structure describes the details of Intel RST for PCIe Storage remapping Note: In order to use this feature, Intel RST Driver is required.

- struct [RST_CONFIG](#)

Rapid Storage Technology settings.

14.60.1 Detailed Description

Rst policy.

Copyright

INTEL CONFIDENTIAL Copyright 2018 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.61 RtcConfig.h File Reference

RTC policy.

Classes

- struct [RTC_CONFIG](#)

The [RTC_CONFIG](#) block describes the expected configuration of RTC configuration.

14.61.1 Detailed Description

RTC policy.

Copyright

INTEL CONFIDENTIAL Copyright 2018 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.62 SaMiscPeiConfig.h File Reference

Policy details for miscellaneous configuration in System Agent.

Classes

- struct [SA_MISC_PEI_CONFIG](#)

This configuration block is to configure SA Miscellaneous variables during PEI Post-Mem.

14.62.1 Detailed Description

Policy details for miscellaneous configuration in System Agent.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2019 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.63 SaMiscPeiPreMemConfig.h File Reference

Policy details for miscellaneous configuration in System Agent.

Classes

- struct [SA_MISC_PEI_PREMEM_CONFIG](#)

This configuration block is to configure SA Miscellaneous variables during PEI Pre-Mem phase like programming different System Agent BARs, TsegSize, MmioSize required etc.

14.63.1 Detailed Description

Policy details for miscellaneous configuration in System Agent.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2022 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

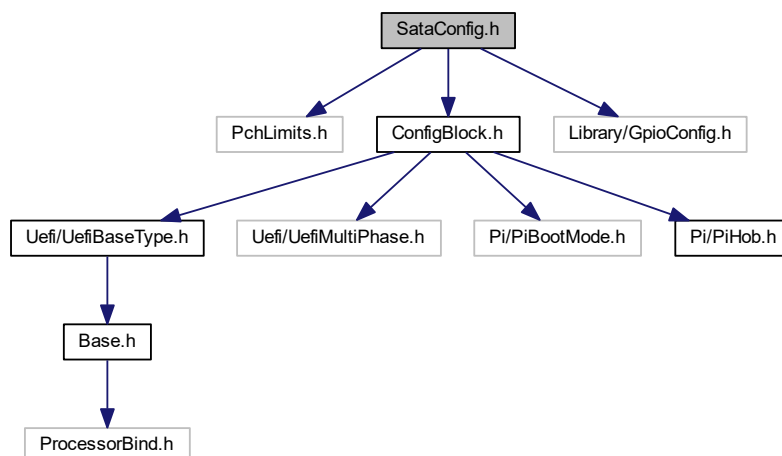
This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.64 SataConfig.h File Reference

Sata policy.

```
#include <PchLimits.h>
#include <ConfigBlock.h>
#include <Library/GpioConfig.h>
Include dependency graph for SataConfig.h:
```



Classes

- struct [PCH_SATA_PORT_CONFIG](#)

This structure configures the features, property, and capability for each SATA port.

- struct [SATA_THERMAL_THROTTLING](#)

This structure lists PCH supported SATA thermal throttling register setting for customization.

- struct [SATA_CONFIG](#)

The [SATA_CONFIG](#) block describes the expected configuration of the SATA controllers.

14.64.1 Detailed Description

Sata policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

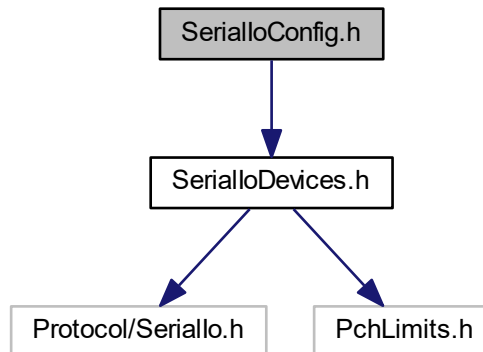
Specification Reference:

14.65 SerialIoConfig.h File Reference

Serial IO policy.

```
#include <SerialIoDevices.h>
```

Include dependency graph for SerialIoConfig.h:



Classes

- struct [SERIAL_IO_CONFIG](#)

The [SERIAL_IO_CONFIG](#) block provides the configurations to set the Serial IO controllers.

14.65.1 Detailed Description

Serial IO policy.

Copyright

INTEL CONFIDENTIAL Copyright 2016 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

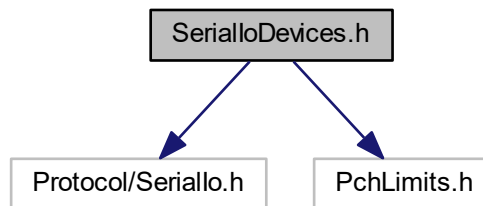
14.66 SerialIoDevices.h File Reference

Serial IO policy.

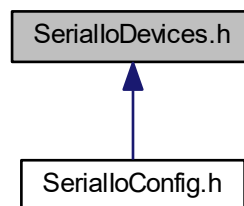
```
#include <Protocol/SerialIo.h>
```

```
#include <PchLimits.h>
```

Include dependency graph for SerialIoDevices.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [SPI_PIN_MUX](#)
SPI signals pin muxing settings.
- struct [SERIAL_IO_SPI_CONFIG](#)
The [SERIAL_IO_SPI_CONFIG](#) provides the configurations to set the Serial IO SPI controller.
- struct [SERIAL_IO_UART_ATTRIBUTES](#)
UART Settings.
- struct [UART_PIN_MUX](#)
UART signals pin muxing settings.
- struct [SERIAL_IO_UART_CONFIG](#)
Serial IO UART Controller Configuration.
- struct [I2C_PIN_MUX](#)
I2C signals pin muxing settings.
- struct [SERIAL_IO_I2C_CONFIG](#)
Serial IO I2C Controller Configuration.

Enumerations

- enum [SERIAL_IO_SPI_MODE](#)
Available working modes for SerialIo SPI Host Controller.
- enum [SERIAL_IO_CS_POLARITY](#)
Used to set Inactive/Idle polarity of Spi Chip Select.
- enum [SERIAL_IO_UART_MODE](#)
Available working modes for SerialIo UART Host Controller.
- enum [SERIAL_IO_UART_PG](#)
- enum [SERIAL_IO_I2C_MODE](#)
Available working modes for SerialIo I2C Host Controller.

14.66.1 Detailed Description

Serial IO policy.

Copyright

INTEL CONFIDENTIAL Copyright 2016 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.66.2 Enumeration Type Documentation

14.66.2.1 SERIAL_IO_I2C_MODE

enum [SERIAL_IO_I2C_MODE](#)

Available working modes for SerialIo I2C Host Controller.

```
0: SerialIoI2cDisabled;
- Device is placed in D3
- Gpio configuration is skipped
- PSF:
!important! If given device is Function 0 and other higher functions on given device
are enabled, PSF disabling is skipped. PSF default will remain and device PCI CFG Space will s
This is needed to allow PCI enumerator access functions above 0 in a multifunction device.
```

1: SerialIoI2cPci;

- Gpio pin configuration in native mode for each assigned pin
- Device will be enabled in PSF
- Only BAR0 will be enabled 2: SerialIoI2cHidden;
- Gpio pin configuration in native mode for each assigned pin
- Device disabled in the PSF
- Both BARs are enabled, BAR1 becomes devices Pci Config Space
- BAR0 assigned from the global PCH reserved memory range, reported as motherboard resource by SIRC

Note

If this controller is located at function 0 and it's mode is set to hidden it will not be visible in the PCI space.

Definition at line 217 of file SerialIoDevices.h.

14.66.2.2 SERIAL_IO_SPI_MODE

enum [SERIAL_IO_SPI_MODE](#)

Available working modes for SerialIo SPI Host Controller.

```
0: SerialIoSpiDisabled;
- Device is placed in D3
- Gpio configuration is skipped
- PSF:
!important! If given device is Function 0 and other higher functions on given device
are enabled, PSF disabling is skipped. PSF default will remain and device PCI CFG Space will s
This is needed to allow PCI enumerator access functions above 0 in a multifunction device.
```

1: SerialIoSpiPci;

- Gpio pin configuration in native mode for each assigned pin
- Device will be enabled in PSF
- Only BAR0 will be enabled 2: SerialIoSpiHidden;
- Gpio pin configuration in native mode for each assigned pin
- Device disabled in the PSF
- Both BARs are enabled, BAR1 becomes devices Pci Config Space
- BAR0 assigned from the global PCH reserved memory range, reported as motherboard resource by SIRC

Note

If this controller is located at function 0 and it's mode is set to hidden it will not be visible in the PCI space.

Definition at line 65 of file SerialIoDevices.h.

14.66.2.3 SERIAL_IO_UART_MODE

enum [SERIAL_IO_UART_MODE](#)

Available working modes for SerialIo UART Host Controller.

```
0: SerialIoUartDisabled;
- Device is placed in D3
- Gpio configuration is skipped
- PSF:
!important! If given device is Function 0 and other higher functions on given device
are enabled, PSF disabling is skipped. PSF default will remain and device PCI CFG Space will s
This is needed to allow PCI enumerator access functions above 0 in a multifunction device.
```

1: SerialIoUartPci;

- Designated for Serial IO UART OS driver usage
- Gpio pin configuration in native mode for each assigned pin
- Device will be enabled in PSF
- Only BAR0 will be enabled 2: SerialIoUartHidden;
- Designated for BIOS and/or DBG2 OS kernel debug
- Gpio pin configuration in native mode for each assigned pin
- Device disabled in the PSF
- Both BARs are enabled, BAR1 becomes devices Pci Config Space
- BAR0 assigned from the global PCH reserved memory range, reported as motherboard resource by SIRC

Note

If this controller is located at function 0 and it's mode is set to hidden it will not be visible in the PCI space. 3: SerialIoUartCom;

- Designated for 16550/PNP0501 compatible COM device
- Gpio pin configuration in native mode for each assigned pin
- Device disabled in the PSF
- Both BARs are enabled, BAR1 becomes devices Pci Config Space
- BAR0 assigned from the global PCH reserved memory range, reported as motherboard resource by SIRC 4: SerialIoUartSkipInit;
- Gpio configuration is skipped
- PSF configuration is skipped
- BAR assignemnt is skipped
- D-satate setting is skipped

Definition at line 140 of file SerialIoDevices.h.

14.66.2.4 SERIAL_IO_UART_PG

enum [SERIAL_IO_UART_PG](#)

Enumerator

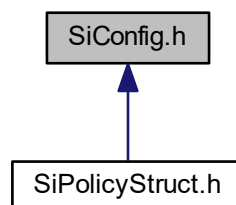
SerialIoUartPgDisabled	No _PS0/_PS3 support, device left in D0, after initialization.
SerialIoUartPgEnabled	In mode: SerialIoUartCom; _PS0/_PS3 that supports getting device out of reset In mode: SerialIoUartPci Keeps UART in D0 and assigns Fixed MMIO for SEC/PEI usage only.
SerialIoUartPgAuto	_PS0 and _PS3, detection through ACPI if device was initialized prior to first PG. If it was used (DBG2) PG is disabled,

Definition at line 183 of file SerialIoDevices.h.

14.67 SiConfig.h File Reference

Si Config Block.

This graph shows which files directly or indirectly include this file:



Classes

- struct [SI_CONFIG](#)

The Silicon Policy allows the platform code to publish a set of configuration information that the RC drivers will use to configure the silicon hardware.

- struct [SVID_SID_VALUE](#)

Subsystem Vendor ID / Subsystem ID.

14.67.1 Detailed Description

Si Config Block.

Copyright

INTEL CONFIDENTIAL Copyright 2016 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

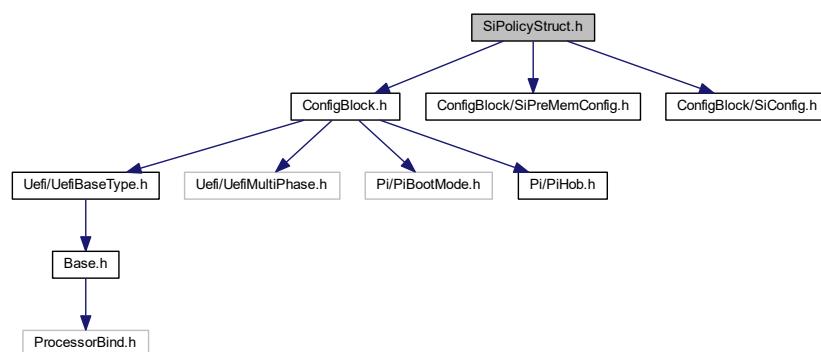
This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:**14.68 SiPolicyStruct.h File Reference**

Intel reference code configuration policies.

```
#include <ConfigBlock.h>
#include <ConfigBlock/SiPreMemConfig.h>
#include <ConfigBlock/SiConfig.h>
```

Include dependency graph for SiPolicyStruct.h:



Classes

- struct [_SI_PREMEM_POLICY_STRUCT](#)
SI Policy PPI in Pre-Mem
All SI config block change history will be listed here
- struct [_SI_POLICY_STRUCT](#)
SI Policy PPI
All SI config block change history will be listed here

Macros

- #define [SI_POLICY_REVISION](#) 1
Silicon Policy revision number Any change to this structure will result in an update in the revision number.
- #define [SI_PREMEM_POLICY_REVISION](#) 1
Silicon pre-memory Policy revision number Any change to this structure will result in an update in the revision number.

14.68.1 Detailed Description

Intel reference code configuration policies.

Copyright

INTEL CONFIDENTIAL Copyright 2014 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.68.2 Macro Definition Documentation

14.68.2.1 SI_POLICY_REVISION

```
#define SI_POLICY_REVISION 1
```

Silicon Policy revision number Any change to this structure will result in an update in the revision number.

This member specifies the revision of the Silicon Policy. This field is used to indicate change to the policy structure.

Revision 1:

- Initial version.

Definition at line 52 of file SiPolicyStruct.h.

14.68.2.2 SI_PREMEM_POLICY_REVISION

```
#define SI_PREMEM_POLICY_REVISION 1
```

Silicon pre-memory Policy revision number Any change to this structure will result in an update in the revision number.

Revision 1:

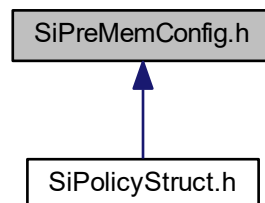
- Initial version.

Definition at line 61 of file SiPolicyStruct.h.

14.69 SiPreMemConfig.h File Reference

Si Config Block PreMem.

This graph shows which files directly or indirectly include this file:



Classes

- struct [SI_PREMEM_CONFIG](#)

The Silicon PreMem Policy allows the platform code to publish a set of configuration information that the RC drivers will use to configure the silicon hardware.

14.69.1 Detailed Description

Si Config Block PreMem.

Copyright

INTEL CONFIDENTIAL Copyright 2017 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.70 SmbusConfig.h File Reference

Smbus policy.

Classes

- struct [PCH_SMBUS_PREMEM_CONFIG](#)

The SMBUS_CONFIG block lists the reserved addresses for non-ARP capable devices in the platform.

14.70.1 Detailed Description

Smbus policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

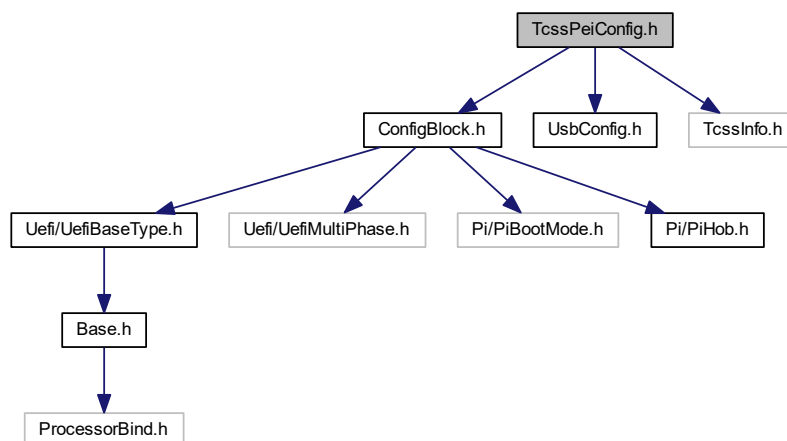
Specification Reference:

14.71 TcssPeiConfig.h File Reference

TCSS PEI policy.

```
#include <ConfigBlock.h>
#include <UsbConfig.h>
#include <TcssInfo.h>
```

Include dependency graph for TcssPeiConfig.h:



Classes

- struct [IOM_AUX_ORI_PAD_CONFIG](#)
The [IOM_AUX_ORI_PAD_CONFIG](#) describes IOM TypeC port map GPIO pin.
- struct [IOM_INTERFACE_CONFIG](#)
The [IOM_EC_INTERFACE_CONFIG](#) block describes interaction between BIOS and IOM-EC.
- struct [PMC_INTERFACE_CONFIG](#)
The [PMC_INTERFACE_CONFIG](#) block describes interaction between BIOS and PMC.
- struct [SA_XDCI_IRQ_INT_CONFIG](#)
The SA XDCI INT Pin and IRQ number.
- struct [TCSS_PCIE_PORT_POLICY](#)
The [TCSS_PCIE_PORT_POLICY](#) block describes PCIe settings for TCSS.
- struct [TCSS_PCIE_PEI_POLICY](#)
[TCSS_PCIE_PEI_POLICY](#) describes PCIe port settings for TCSS.
- struct [TCSS_IOM_ORI_OVERRIDE](#)
The [TCSS_IOM_PEI_CONFIG](#) block describes IOM Aux/HSL override settings for TCSS.
- struct [TCSS_IOM_PEI_CONFIG](#)
The [TCSS_IOM_PEI_CONFIG](#) block describes IOM settings for TCSS.
- struct [TCSS_MISC_PEI_CONFIG](#)
The [TCSS_MISC_PEI_CONFIG](#) block describes MISC settings for TCSS.
- struct [TCSS_PEI_CONFIG](#)
The [TCSS_PEI_CONFIG](#) block describes TCSS settings for SA.

14.71.1 Detailed Description

TCSS PEI policy.

Copyright

INTEL CONFIDENTIAL Copyright 2016 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

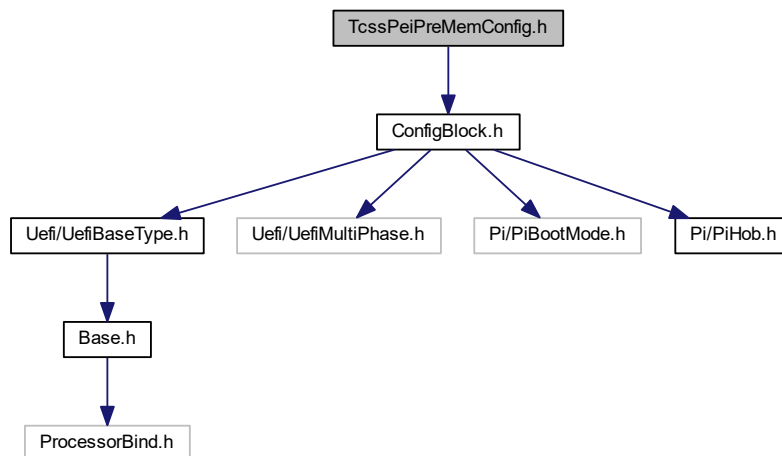
Specification Reference:

14.72 TcssPeiPreMemConfig.h File Reference

TCSS PEI PreMem policy.

```
#include <ConfigBlock.h>
```

Include dependency graph for TcssPeiPreMemConfig.h:



Classes

- union [TCSS_DEVEN_PEI_PREMEM_CONFIG](#)
The [TCSS_DEVEN_PEI_PREMEM_CONFIG](#) block describes Device Enable settings for TCSS.
- struct [TCSS_USBTC_PEI_PERMEM_CONFIG](#)
The [TCSS_USBTC_PEI_PERMEM_CONFIG](#) block describes IOM settings for TCSS.
- struct [TCSS_MISC_PEI_PREMEM_CONFIG](#)
The [TCSS_MISC_PEI_PREMEM_CONFIG](#) block describes MISC settings for TCSS.
- struct [TCSS_PEI_PREMEM_CONFIG](#)
This configuration block describes TCSS settings.

14.72.1 Detailed Description

TCSS PEI PreMem policy.

Copyright

INTEL CONFIDENTIAL Copyright 2016 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and

treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

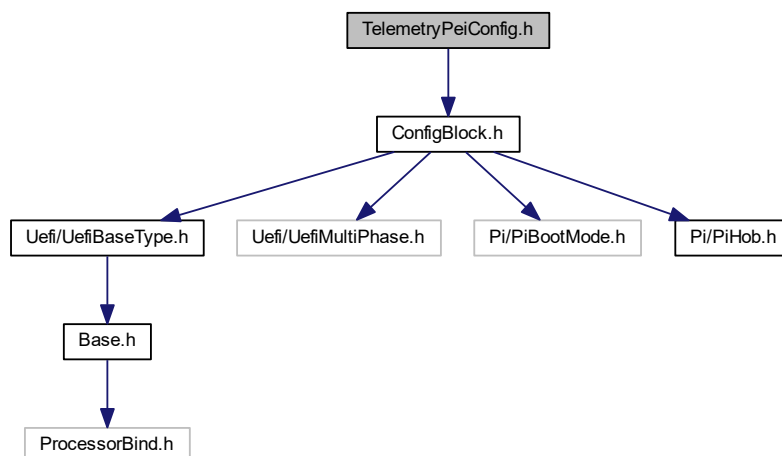
Specification Reference:

14.73 TelemetryPeiConfig.h File Reference

Configurations for Telemetry.

```
#include <ConfigBlock.h>
```

Include dependency graph for TelemetryPeiConfig.h:



Classes

- struct [TELEMETRY_PEI_PREMEM_CONFIG](#)
This configuration block describes Telemetry settings in PreMem.
- struct [TELEMETRY_PEI_CONFIG](#)
This configuration block describes Telemetry settings in PostMem.

14.73.1 Detailed Description

Configurations for Telemetry.

Copyright

INTEL CONFIDENTIAL Copyright 2019 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.74 ThcConfig.h File Reference

Touch Host Controller policy.

Classes

- struct [THC_HID_OVER_SPI](#)
THC Hid Over SPI mode related settings.
- struct [THC_PORT](#)
Port Configuration structure required for each Port that THC might use.
- struct [THC_CONFIG](#)
THC_CONFIG block provides the configurations for Touch Host Controllers.

Enumerations

- enum [THC_PORT_ASSIGNMENT](#)
Available Port Assignments.
- enum [THC_MODE](#)
Available modes.

14.74.1 Detailed Description

Touch Host Controller policy.

Copyright

INTEL CONFIDENTIAL Copyright 2017 - 2022 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.74.2 Enumeration Type Documentation

14.74.2.1 THC_MODE

enum [THC_MODE](#)

Available modes.

Enumerator

Thc	Intel THC protocol.
HidOverSpi	Hid Over SPI protocol.

Definition at line 57 of file ThcConfig.h.

14.74.2.2 THC_PORT_ASSIGNMENT

enum [THC_PORT_ASSIGNMENT](#)

Available Port Assignments.

Enumerator

ThcAssignmentNone	None of the avaialbe controllers assigned.
ThcAssignmentThc0	Port assigned to THC0.
ThcAssignmentThc1	Port assigned to THC1.

Definition at line 48 of file ThcConfig.h.

14.75 ThermalConfig.h File Reference

Thermal policy.

Classes

- struct [THERMAL_THROTTLE_LEVELS](#)
This structure lists PCH supported throttling register setting for custimization.
- struct [DMI_HW_WIDTH_CONTROL](#)
This structure allows to customize DMI HW Autonomous Width Control for Thermal and Mechanical spec design.
- struct [TS_GPIO_PIN_SETTING](#)
This structure configures PCH memory throttling thermal sensor GPIO PIN settings.
- struct [PCH_MEMORY_THROTTLING](#)
This structure supports an external memory thermal sensor (TS-on-DIMM or TS-on-Board).
- struct [THERMAL_CONFIG](#)
The [THERMAL_CONFIG](#) block describes the expected configuration of the Thermal IP block.

14.75.1 Detailed Description

Thermal policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2018 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

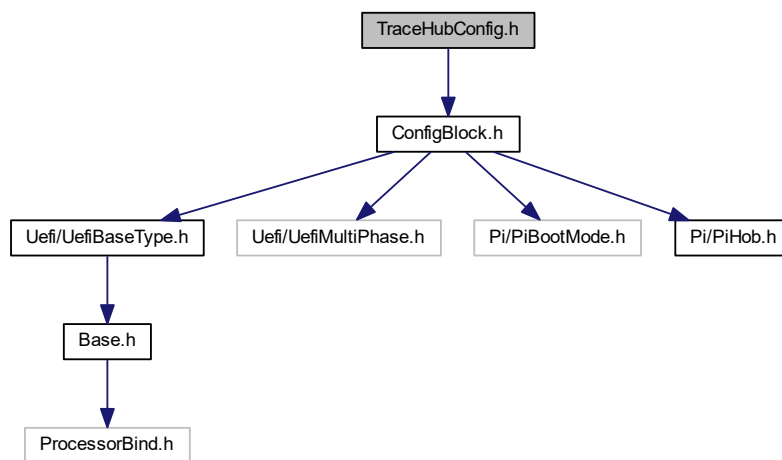
Specification Reference:

14.76 TraceHubConfig.h File Reference

Configurations for TraceHub.

```
#include <ConfigBlock.h>
```

Include dependency graph for TraceHubConfig.h:



Classes

- struct [TRACE_HUB_CONFIG](#)
TRACE_HUB_CONFIG block describes TraceHub settings.
- struct [CPU_TRACE_HUB_PREMEM_CONFIG](#)
*CPU Trace Hub PreMem Configuration Contains Trace Hub settings for CPU side tracing **Revision 1**: - Initial version.*
- struct [PCH_TRACE_HUB_PREMEM_CONFIG](#)
*PCH Trace Hub PreMem Configuration Contains Trace Hub settings for PCH side tracing **Revision 1**: - Initial version.*

Enumerations

- enum [TRACE_HUB_ENABLE_MODE](#)
The TRACE_HUB_ENABLE_MODE describes TraceHub mode of operation.
- enum [TRACE_BUFFER_SIZE](#)
The TRACE_BUFFER_SIZE describes the desired TraceHub buffer size.

14.76.1 Detailed Description

Configurations for TraceHub.

Copyright

INTEL CONFIDENTIAL Copyright 2018 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.77 TsnConfig.h File Reference

TSN Config policy.

Classes

- struct [TSN_CONFIG](#)

The [TSN_CONFIG](#) block describes policies related to Time Sensitive Networking(TSN)

14.77.1 Detailed Description

TSN Config policy.

Copyright

INTEL CONFIDENTIAL Copyright 2019 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

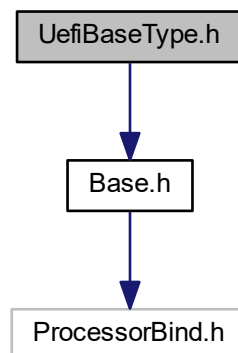
This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:**14.78 UefiBaseType.h File Reference**

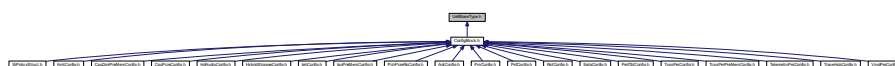
Defines data types and constants introduced in UEFI.

```
#include <Base.h>
```

Include dependency graph for UefiBaseType.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [EFI_TIME](#)
EFI Time Abstraction: Year: 1900 - 9999 Month: 1 - 12 Day: 1 - 31 Hour: 0 - 23 Minute: 0 - 59 Second: 0 - 59 Nanosecond: 0 - 999,999,999 TimeZone: -1440 to 1440 or 2047.
- struct [EFI_MAC_ADDRESS](#)
32-byte buffer containing a network Media Access Control address.
- union [EFI_IP_ADDRESS](#)
16-byte buffer aligned on a 4-byte boundary.

Macros

- #define [EFIERR\(_a\) ENCODE_ERROR\(_a\)](#)
Define macro to encode the status code.
- #define [EFI_SIZE_TO_PAGES](#)(Size) (((Size) >> EFI_PAGE_SHIFT) + (((Size) & EFI_PAGE_MASK) ? 1 : 0))
Macro that converts a size, in bytes, to a number of EFI_PAGESs.
- #define [EFI_PAGES_TO_SIZE](#)(Pages) ((Pages) << EFI_PAGE_SHIFT)
Macro that converts a number of EFI_PAGES to a size in bytes.
- #define [EFI_IMAGE_MACHINE_IA32](#) 0x014C
PE32+ Machine type for IA32 UEFI images.
- #define [EFI_IMAGE_MACHINE_IA64](#) 0x0200
PE32+ Machine type for IA64 UEFI images.
- #define [EFI_IMAGE_MACHINE_EBC](#) 0x0EBC
PE32+ Machine type for EBC UEFI images.
- #define [EFI_IMAGE_MACHINE_X64](#) 0x8664
PE32+ Machine type for X64 UEFI images.
- #define [EFI_IMAGE_MACHINE_ARMTHUMB_MIXED](#) 0x01C2
PE32+ Machine type for ARM mixed ARM and Thumb/Thumb2 images.
- #define [EFI_IMAGE_MACHINE_AARCH64](#) 0xAA64
PE32+ Machine type for AARCH64 A64 images.
- #define [EFI_IMAGE_MACHINE_RISCV32](#) 0x5032
PE32+ Machine type for RISC-V 32/64/128.

- #define [EFI_SUCCESS](#) RETURN_SUCCESS
Enumeration of EFI_STATUS.

- #define [EFI_NETWORK_UNREACHABLE](#) EFIERR(100)
ICMP error definitions.

- #define [EFI_CONNECTION_FIN](#) EFIERR(104)
Tcp connection status definitions.

Typedefs

- typedef [GUID](#) [EFI_GUID](#)
128-bit buffer containing a unique identifier value.
- typedef [RETURN_STATUS](#) [EFI_STATUS](#)
Function return status for EFI API.
- typedef [VOID](#) * [EFI_HANDLE](#)
A collection of related interfaces.
- typedef [VOID](#) * [EFI_EVENT](#)
Handle to an event structure.
- typedef [UINTN](#) [EFI_TPL](#)
Task priority level.
- typedef [UINT64](#) [EFI_LBA](#)
Logical block address.
- typedef [UINT64](#) [EFI_PHYSICAL_ADDRESS](#)
64-bit physical memory address.
- typedef [UINT64](#) [EFI_VIRTUAL_ADDRESS](#)
64-bit virtual memory address.
- typedef [IPv4_ADDRESS](#) [EFI_IPv4_ADDRESS](#)
4-byte buffer.
- typedef [IPv6_ADDRESS](#) [EFI_IPv6_ADDRESS](#)
16-byte buffer.

14.78.1 Detailed Description

Defines data types and constants introduced in UEFI.

Copyright (c) 2006 - 2021, Intel Corporation. All rights reserved.

Portions copyright (c) 2011 - 2016, ARM Ltd. All rights reserved.

Copyright (c) 2020, Hewlett Packard Enterprise Development LP. All rights reserved.

SPDX-License-Identifier: BSD-2-Clause-Patent

14.78.2 Macro Definition Documentation

14.78.2.1 [EFI_PAGES_TO_SIZE](#)

```
#define EFI_PAGES_TO_SIZE(  
    Pages ) ((Pages) << EFI_PAGE_SHIFT)
```

Macro that converts a number of [EFI_PAGES](#) to a size in bytes.

Parameters

<i>Pages</i>	The number of EFI_PAGES . This parameter is assumed to be type UINTN . Passing in a parameter that is larger than UINTN may produce unexpected results.
--------------	---

Returns

The number of bytes associated with the number of EFI_PAGES specified by Pages.

Definition at line 210 of file UefiBaseType.h.

14.78.2.2 EFI_SIZE_TO_PAGES

```
#define EFI_SIZE_TO_PAGES(  
    Size ) (((Size) >> EFI_PAGE_SHIFT) + (((Size) & EFI_PAGE_MASK) ? 1 : 0))
```

Macro that converts a size, in bytes, to a number of EFI_PAGESs.

Parameters

<i>Size</i>	A size in bytes. This parameter is assumed to be type UINTN. Passing in a parameter that is larger than UINTN may produce unexpected results.
-------------	---

Returns

The number of EFI_PAGESs associated with the number of bytes specified by Size.

Definition at line 197 of file UefiBaseType.h.

14.78.3 Typedef Documentation**14.78.3.1 EFI_IPv4_ADDRESS**

```
typedef IPv4_ADDRESS EFI_IPv4_ADDRESS
```

4-byte buffer.

An IPv4 internet protocol address.

Definition at line 84 of file UefiBaseType.h.

14.78.3.2 EFI_IPv6_ADDRESS

```
typedef IPv6_ADDRESS EFI_IPv6_ADDRESS
```

16-byte buffer.

An IPv6 internet protocol address.

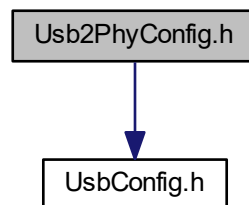
Definition at line 89 of file UefiBaseType.h.

14.79 Usb2PhyConfig.h File Reference

USB2 PHY configuration policy.

```
#include <UsbConfig.h>
```

Include dependency graph for Usb2PhyConfig.h:



Classes

- struct [USB2_PHY_PARAMETERS](#)
This structure configures per USB2 AFE settings.
- struct [USB2_PHY_CONFIG](#)
This structure holds info on how to tune electrical parameters of USB2 ports based on board layout.

14.79.1 Detailed Description

USB2 PHY configuration policy.

Copyright

INTEL CONFIDENTIAL Copyright 2018 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

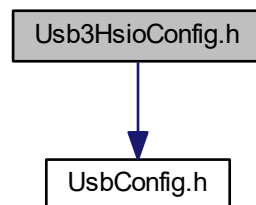
Specification Reference:

14.80 Usb3HsioConfig.h File Reference

USB3 Mod PHY configuration policy.

```
#include <UsbConfig.h>
```

Include dependency graph for Usb3HsioConfig.h:



Classes

- struct [HSIO_PARAMETERS](#)
This structure describes USB3 Port N configuration parameters.
- struct [USB3_HSIO_CONFIG](#)
Structure for holding USB3 tuning parameters.

14.80.1 Detailed Description

USB3 Mod PHY configuration policy.

Copyright

INTEL CONFIDENTIAL Copyright 2018 - 2019 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

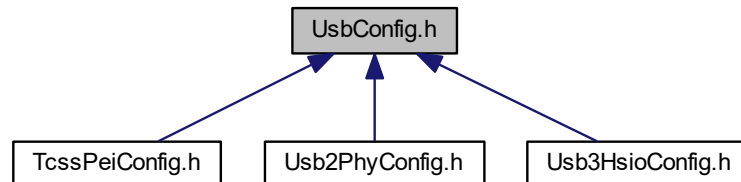
This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.81 UsbConfig.h File Reference

Common USB policy shared between PCH and CPU Contains general features settings for xHCI and xDCI.

This graph shows which files directly or indirectly include this file:



Classes

- struct [USB2_PORT_CONFIG](#)

This structure configures per USB2.0 port settings like enabling and overcurrent protection.

- struct [USB3_PORT_CONFIG](#)

This structure configures per USB3.x port settings like enabling and overcurrent protection.

- struct [XDCI_CONFIG](#)

The [XDCI_CONFIG](#) block describes the configurations of the xDCI Usb Device controller.

- struct [USB_CONFIG](#)

This member describes the expected configuration of the USB controller, Platform modules may need to refer Setup options, schematic, BIOS specification to update this field.

Macros

- `#define` [USB_OC_MAX_PINS](#) 16

Total OC pins number (both physical and virtual)

14.81.1 Detailed Description

Common USB policy shared between PCH and CPU Contains general features settings for xHCI and xDCI.

Copyright

INTEL CONFIDENTIAL Copyright 2017 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

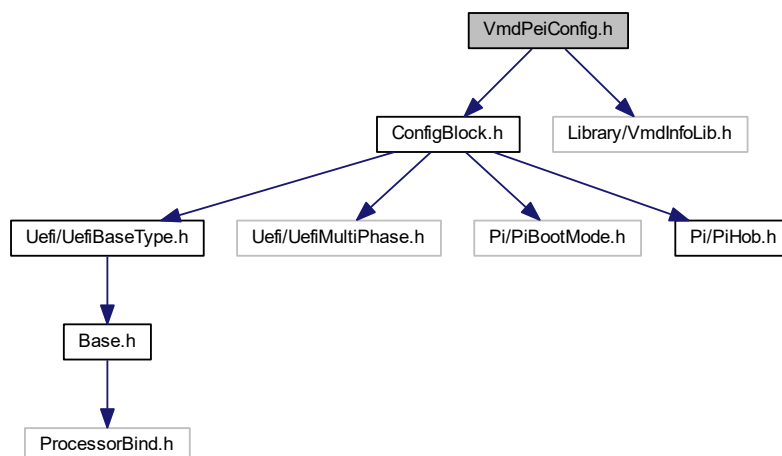
This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:**14.82 VmdPeiConfig.h File Reference**

VMD PEI policy.

```
#include <ConfigBlock.h>
#include <Library/VmdInfoLib.h>
```

Include dependency graph for VmdPeiConfig.h:



Classes

- struct [VMD_PEI_CONFIG](#)

This configuration block is to configure VMD related variables used in PostMem PEI.

14.82.1 Detailed Description

VMD PEI policy.

Copyright

INTEL CONFIDENTIAL Copyright 2017 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

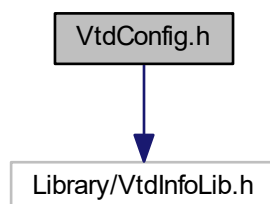
Specification Reference:

14.83 VtdConfig.h File Reference

VT-d policy definitions.

```
#include <Library/VtdInfoLib.h>
```

Include dependency graph for VtdConfig.h:



Classes

- struct [VTD_CONFIG](#)
The data elements should be initialized by a Platform Module.
- struct [VTD_DXE_CONFIG](#)
The data structure is for VT-d driver initialization in DXE
Revision 1:

Typedefs

- typedef [EFI_STATUS](#)(* [VTD_ENABLE_DMA_BUFFER](#)) ([IN](#) [VTD_ENGINE_INDEX](#) VtdEngineIndex)
Enable DMA buffer in VTd.

14.83.1 Detailed Description

VT-d policy definitions.

Copyright

INTEL CONFIDENTIAL Copyright 2014 - 2021 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

14.83.2 Typedef Documentation

14.83.2.1 VTD_ENABLE_DMA_BUFFER

```
typedef EFI\_STATUS ( * VTD\_ENABLE\_DMA\_BUFFER) (IN VTD\_ENGINE\_INDEX VtdEngineIndex)
```

Enable DMA buffer in VTd.

Parameters

in	<i>VtdEngineIndex</i>	VTd engine index.
----	-----------------------	-------------------

Return values

<i>EFI_SUCCESS</i>	Enable successfully.
<i>EFI_INVALID_PARAMETER</i>	Input parameters are invalid.
<i>EFI_UNSUPPORTED</i>	VTd base is zero.
<i>EFI_OUT_OF_RESOURCES</i>	There is no additional space in the PPI database.

Definition at line 59 of file VtdConfig.h.

14.84 WatchDogConfig.h File Reference

WatchDog policy.

Classes

- struct [PCH_WDT_PREMEM_CONFIG](#)

This policy clears status bits and disable watchdog, then lock the WDT registers.

14.84.1 Detailed Description

WatchDog policy.

Copyright

INTEL CONFIDENTIAL Copyright 2015 - 2020 Intel Corporation.

The source code contained or described herein and all documents related to the source code ("Material") are owned by Intel Corporation or its suppliers or licensors. Title to the Material remains with Intel Corporation or its suppliers and licensors. The Material may contain trade secrets and proprietary and confidential information of Intel Corporation and its suppliers and licensors, and is protected by worldwide copyright and trade secret laws and treaty provisions. No part of the Material may be used, copied, reproduced, modified, published, uploaded, posted, transmitted, distributed, or disclosed in any way without Intel's prior express written permission.

No license under any patent, copyright, trade secret or other intellectual property right is granted to or conferred upon you by disclosure or delivery of the Materials, either expressly, by implication, inducement, estoppel or otherwise. Any license under such intellectual property rights must be express and approved by Intel in writing.

Unless otherwise agreed by Intel in writing, you may not remove or alter this notice or any other notice embedded in Materials by Intel or Intel's suppliers or licensors in any way.

This file contains an 'Intel Peripheral Driver' and is uniquely identified as "Intel Reference Module" and is licensed for Intel CPUs and chipsets under the terms of your license agreement with Intel or your vendor. This file may be modified by the user, subject to additional terms of the license agreement.

Specification Reference:

Index

- [_BASE_INT_SIZE_OF](#)
 - [Base.h, 393](#)
- [_CONFIG_BLOCK, 59](#)
- [_CONFIG_BLOCK_HEADER, 60](#)
- [_CONFIG_BLOCK_TABLE_STRUCT, 61](#)
- [_EFI_MM_RESERVED_MMRAM_REGION, 62](#)
 - [MmramReservedSize, 62](#)
 - [MmramReservedStart, 62](#)
- [_EFI_PEI_MP_SERVICES_PPI, 63](#)
- [_INT_SIZE_OF](#)
 - [Base.h, 393](#)
- [_ITBT_GENERIC_CONFIG, 63](#)
 - [ITbtForcePowerOnTimeoutInMs, 64](#)
- [_ITBT_ROOTPORT_CONFIG, 64](#)
- [_LIST_ENTRY, 65](#)
- [_PEI_ITBT_CONFIG, 66](#)
- [_PEI_PREMEM_SI_DEFAULT_POLICY_INIT_PPI, 67](#)
- [_PEI_SI_DEFAULT_POLICY_INIT_PPI, 67](#)
- [_PPM_CUSTOM_CTDTP_TABLE, 68](#)
- [_SI_POLICY_STRUCT, 68](#)
- [_SI_PREMEM_POLICY_STRUCT, 70](#)
- [ABS](#)
 - [Base.h, 393](#)
- [AcpiL6dPmeHandling](#)
 - [PCH_GENERAL_CONFIG, 265](#)
- [AcSplitLock](#)
 - [CPU_CONFIG, 86](#)
- [ActiveCoreCount](#)
 - [CPU_CONFIG_LIB_PREMEM_CONFIG, 91](#)
- [ActiveCoreCount1](#)
 - [CPU_CONFIG_LIB_PREMEM_CONFIG, 91](#)
- [ActiveSmallCoreCount](#)
 - [CPU_CONFIG_LIB_PREMEM_CONFIG, 91](#)
- [ActiveSocNorthAtomCoreCount](#)
 - [CPU_CONFIG_LIB_PREMEM_CONFIG, 92](#)
- [AddConfigBlock](#)
 - [ConfigBlockLib.h, 412](#)
- [ADR_CONFIG, 71](#)
- [ADR_SOURCE_ENABLE, 72](#)
- [AdrConfig.h, 385](#)
- [AesEnable](#)
 - [CPU_CONFIG, 86](#)
- [AetEnabled](#)
 - [TRACE_HUB_CONFIG, 366](#)
- [ALIGN_POINTER](#)
 - [Base.h, 394](#)
- [ALIGN_VALUE](#)
 - [Base.h, 394](#)
- [ALIGN_VARIABLE](#)
 - [Base.h, 395](#)
- [AMT_DXE_CONFIG, 73](#)
 - [AmtbxHotkeyPressed, 74](#)
 - [AmtbxSelectionScreen, 74](#)
 - [CiraTimeout, 74](#)
 - [CpuReplacementTimeout, 75](#)
 - [HideUnConfigureMeConfirm, 75](#)
 - [MebxDebugMsg, 75](#)
 - [MebxGraphicsMode, 75](#)
 - [MebxNonUiTextMode, 76](#)
 - [MebxUiTextMode, 76](#)
 - [OemResolutionSettingsRsvd, 76](#)
- [AMT_PEI_CONFIG, 77](#)
 - [AmtEnabled, 78](#)
 - [AmtKvmEnabled, 78](#)
 - [ManageabilityMode, 78](#)
 - [RemoteAssistance, 78](#)
 - [WatchDogEnabled, 79](#)
 - [WatchDogTimerBios, 79](#)
 - [WatchDogTimerOs, 79](#)
- [AMT_REPORT_ERROR](#)
 - [AmtConfig.h, 387](#)
- [AmtbxHotkeyPressed](#)
 - [AMT_DXE_CONFIG, 74](#)
- [AmtbxSelectionScreen](#)
 - [AMT_DXE_CONFIG, 74](#)
- [AmtConfig.h, 386](#)
 - [AMT_REPORT_ERROR, 387](#)
- [AmtEnabled](#)
 - [AMT_PEI_CONFIG, 78](#)
- [AmtKvmEnabled](#)
 - [AMT_PEI_CONFIG, 78](#)
- [ANALYZER_NORETURN](#)
 - [Base.h, 395](#)
- [ANALYZER_UNREACHABLE](#)
 - [Base.h, 395](#)
- [ARRAY_SIZE](#)
 - [Base.h, 395](#)
- [AtomClusterRatio](#)
 - [OVERCLOCKING_PREMEM_CONFIG, 244](#)
- [AtomL2MaxOcRatio](#)
 - [OVERCLOCKING_PREMEM_CONFIG, 244](#)
- [AtomL2VoltageAdaptive](#)
 - [OVERCLOCKING_PREMEM_CONFIG, 244](#)
- [AtomL2VoltageMode](#)
 - [OVERCLOCKING_PREMEM_CONFIG, 245](#)
- [AtomL2VoltageOffset](#)
 - [OVERCLOCKING_PREMEM_CONFIG, 245](#)
- [AtomL2VoltageOverride](#)

- OVERCLOCKING_PREMEM_CONFIG, 245
- AtomTurboRatioLimitNumCore
 - CPU_POWER_MGMT_BASIC_CONFIG, 114
- AtomTurboRatioLimitRatio
 - CPU_POWER_MGMT_BASIC_CONFIG, 114
- AudioLinkDmic
 - HDAUDIO_PREMEM_CONFIG, 197
- AudioLinkHda
 - HDAUDIO_PREMEM_CONFIG, 197
- AudioLinkSndw
 - HDAUDIO_PREMEM_CONFIG, 198
- AudioLinkSsp
 - HDAUDIO_PREMEM_CONFIG, 198
- Avx2VoltageScaleFactor
 - OVERCLOCKING_PREMEM_CONFIG, 245
- Avx3Disable
 - CPU_CONFIG, 87
- Avx512VoltageScaleFactor
 - OVERCLOCKING_PREMEM_CONFIG, 246
- AvxDisable
 - CPU_CONFIG, 87
- Base.h, 388
 - _BASE_INT_SIZE_OF, 393
 - _INT_SIZE_OF, 393
 - ABS, 393
 - ALIGN_POINTER, 394
 - ALIGN_VALUE, 394
 - ALIGN_VARIABLE, 395
 - ANALYZER_NORETURN, 395
 - ANALYZER_UNREACHABLE, 395
 - ARRAY_SIZE, 395
 - BASE_ARG, 396
 - BASE_CR, 396
 - BASE_LIST, 406
 - ENCODE_ERROR, 397
 - ENCODE_WARNING, 397
 - FALSE, 398
 - MAX, 398
 - MIN, 399
 - NORETURN, 399
 - OFFSET_OF, 399
 - RETURN_ADDRESS, 400
 - RETURN_BUFFER_TOO_SMALL, 400
 - RETURN_ERROR, 400
 - RETURNS_TWICE, 401
 - SIGNATURE_16, 401
 - SIGNATURE_32, 401
 - SIGNATURE_64, 402
 - STATIC_ASSERT, 403
 - TRUE, 403
 - UNREACHABLE, 403
 - VA_ARG, 403
 - VA_COPY, 405
 - VA_END, 405
 - VA_LIST, 406
 - VA_START, 405
- BASE_ARG
 - Base.h, 396
- BASE_CR
 - Base.h, 396
- BASE_LIST
 - Base.h, 406
- BaseAddress
 - EFI_HOB_UEFI_CAPSULE, 165
- BclkSource
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 92
- BIOS_GUARD_CONFIG, 80
- BiosGuard
 - CPU_SECURITY_PREMEM_CONFIG, 139
- BiosGuardConfig.h, 407
 - PLATFORM_SEND_EC_COMMAND, 408
- BiosInterface
 - PCH_LOCK_DOWN_CONFIG, 273
- BiosInterfaceLock
 - RTC_CONFIG, 320
- BiosLock
 - PCH_LOCK_DOWN_CONFIG, 273
- BmeMasterSlaveEnabled
 - PCH_ESPI_CONFIG, 262
- BootFrequency
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 92
 - CPU_POWER_MGMT_BASIC_CONFIG, 114
- BtAudioOffload
 - CNVI_CONFIG, 82
- C10DynamicThresholdAdjustment
 - PCH_PM_CONFIG, 284
- C3AutoDemotion
 - CPU_POWER_MGMT_TEST_CONFIG, 129
- C3UnDemotion
 - CPU_POWER_MGMT_TEST_CONFIG, 129
- CdClock
 - GRAPHICS_PEI_CONFIG, 187
- ChHashInterleaveBit
 - MEMORY_CONFIGURATION, 236
- CiraTimeout
 - AMT_DXE_CONFIG, 74
- ClkReqMsgEnable
 - CPU_PCIE_RP_PREMEM_CONFIG, 108
- ClkReqMsgEnableRp
 - CPU_PCIE_RP_PREMEM_CONFIG, 108
- CNVI_CONFIG, 81
 - BtAudioOffload, 82
 - Mode, 82
- CNVI_PIN_MUX, 83
- CNVI_PREMEM_CONFIG, 84
- CnviConfig.h, 409
- ComplianceTestMode
 - PCIE_COMMON_CONFIG, 297
- ComputeDieSscEnable
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 93
- ComputeDieSscValue
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 93
- ConfigBlock.h, 410
- ConfigBlockLib.h, 411
 - AddConfigBlock, 412
 - CreateConfigBlockTable, 412

- GetConfigBlock, [413](#)
- ConfigTdpLevel
 - CPU_POWER_MGMT_TEST_CONFIG, [129](#)
- CoreMaxOcRatio
 - OVERCLOCKING_PREMEM_CONFIG, [246](#)
- CoreVfConfigScope
 - OVERCLOCKING_PREMEM_CONFIG, [246](#)
- CoreVfPointOffset
 - OVERCLOCKING_PREMEM_CONFIG, [247](#)
- CoreVfPointOffsetMode
 - OVERCLOCKING_PREMEM_CONFIG, [247](#)
- CoreVoltageAdaptive
 - OVERCLOCKING_PREMEM_CONFIG, [247](#)
- CoreVoltageMode
 - OVERCLOCKING_PREMEM_CONFIG, [247](#)
- CoreVoltageOffset
 - OVERCLOCKING_PREMEM_CONFIG, [248](#)
- CoreVoltageOverride
 - OVERCLOCKING_PREMEM_CONFIG, [248](#)
- CPU_CONFIG, [85](#)
 - AcSplitLock, [86](#)
 - AesEnable, [86](#)
 - Avx3Disable, [87](#)
 - AvxDisable, [87](#)
 - PpinSupport, [87](#)
 - SmbiosType4MaxSpeedOverride, [88](#)
 - TxtEnable, [88](#)
 - X2ApicSupport, [88](#)
- CPU_CONFIG_LIB_PREMEM_CONFIG, [89](#)
 - ActiveCoreCount, [91](#)
 - ActiveCoreCount1, [91](#)
 - ActiveSmallCoreCount, [91](#)
 - ActiveSocNorthAtomCoreCount, [92](#)
 - BclkSource, [92](#)
 - BootFrequency, [92](#)
 - ComputeDieSscEnable, [93](#)
 - ComputeDieSscValue, [93](#)
 - CpuRatio, [93](#)
 - CrashLogEnable, [94](#)
 - CrashLogGprs, [94](#)
 - FClkFrequency, [94](#)
 - InfDfxPwrEnable, [94](#)
 - PeciC10Reset, [95](#)
 - PeciSxReset, [95](#)
 - SocDieSscEnable, [95](#)
 - SocDieSscValue, [96](#)
 - TmeEnable, [96](#)
 - VmxEnable, [96](#)
- CPU_DMI_EQ_PARAM, [97](#)
- CPU_DMI_PREMEM_CONFIG, [98](#)
 - DmiGen3EqPh2Enable, [100](#)
 - DmiGen3EqPh3Method, [100](#)
 - DmiGen3ProgramStaticEq, [100](#)
 - DmiGen3RxCtlePeaking, [100](#)
 - DmiMaxLinkSpeed, [101](#)
- CPU_PCIE_CONFIG, [101](#)
 - EqPh3LaneParam, [102](#)
 - PcieDeviceOverrideTablePtr, [103](#)
 - PegGen3ProgramStaticEq, [103](#)
 - PegGen4ProgramStaticEq, [103](#)
 - SetSecuredRegisterLock, [103](#)
- CPU_PCIE_CONFIG_REVISION
 - CpuPcieConfig.h, [418](#)
- CPU_PCIE_EQ_LANE_PARAM, [104](#)
- CPU_PCIE_EQ_METHOD
 - CpuPcieConfig.h, [418](#)
- CPU_PCIE_GPIO_INFO, [105](#)
- CPU_PCIE_ROOT_PORT_CONFIG, [105](#)
 - Gen4EqPh3Method, [106](#)
- CPU_PCIE_RP_PREMEM_CONFIG, [107](#)
 - ClkReqMsgEnable, [108](#)
 - ClkReqMsgEnableRp, [108](#)
 - LinkDownGpios, [108](#)
 - PcieSpeed, [109](#)
 - RpEnabledMask, [109](#)
 - SlotImplemented, [109](#)
- CPU_PCIE_RTD3_GPIO, [110](#)
- CPU_POWER_MGMT_BASIC_CONFIG, [111](#)
 - AtomTurboRatioLimitNumCore, [114](#)
 - AtomTurboRatioLimitRatio, [114](#)
 - BootFrequency, [114](#)
 - DualTauBoost, [115](#)
 - EightCoreRatioLimit, [115](#)
 - EnableEpbPeciOverride, [115](#)
 - EnableFastMsrHwpReq, [116](#)
 - EnableHwpAutoEppGrouping, [116](#)
 - EnableHwpAutoPerCorePstate, [116](#)
 - EnableHwpScalabilityTracking, [116](#)
 - Enableltbm, [117](#)
 - EnableltbmDriver, [117](#)
 - EnablePerCorePState, [117](#)
 - FiveCoreRatioLimit, [117](#)
 - FourCoreRatioLimit, [118](#)
 - HdcControl, [118](#)
 - Hwp, [118](#)
 - OneCoreRatioLimit, [119](#)
 - PowerLimit1, [119](#)
 - PowerLimit1Time, [119](#)
 - PowerLimit2Power, [120](#)
 - PowerLimit3, [120](#)
 - PowerLimit4, [120](#)
 - SevenCoreRatioLimit, [120](#)
 - SixCoreRatioLimit, [121](#)
 - TccActivationOffset, [121](#)
 - TccOffsetClamp, [121](#)
 - TccOffsetTimeWindowForRatl, [122](#)
 - ThreeCoreRatioLimit, [122](#)
 - TurboRatioLimitNumCore, [122](#)
 - TurboRatioLimitRatio, [122](#)
 - TwoCoreRatioLimit, [123](#)
- CPU_POWER_MGMT_CUSTOM_CONFIG, [123](#)
- CPU_POWER_MGMT_PSYS_CONFIG, [124](#)
- CPU_POWER_MGMT_TEST_CONFIG, [126](#)
 - C3AutoDemotion, [129](#)
 - C3UnDemotion, [129](#)
 - ConfigTdpLevel, [129](#)

- CstateLatencyControl0TimeUnit, 129
- CustomPowerUnit, 130
- PpmlrmSetting, 130
- CPU_POWER_MGMT_VR_CONFIG, 131
 - EnableFastVmode, 135
 - FivrRfiFrequency, 135
 - FivrSpreadSpectrum, 135
 - IccLimit, 136
 - TdcTimeWindow, 136
 - VccInDemotionCapacitanceInUf, 136
 - VccInDemotionEnable, 136
 - VccInDemotionQuiescentPowerInMw, 137
 - VrPowerDeliveryDesign, 137
 - VsysAssertionDeglitchMantissa, 137
 - VsysFullScale, 137
- CPU_SECURITY_PREMEM_CONFIG, 138
 - BiosGuard, 139
 - EnableC6Dram, 140
 - EnableSgx, 140
 - Txt, 140
- CPU_TEST_CONFIG, 141
 - ProcessorTraceMemSize, 142
- CPU_TRACE_HUB_PREMEM_CONFIG, 143
- CPU_TXT_PREMEM_CONFIG, 144
- CpuC10GatePinEnable
 - PCH_PM_CONFIG, 284
- CpuConfig.h, 413
- CpuConfigLibPreMemConfig.h, 414
- CpuDmiPreMemConfig.h, 415
- CpuPcieConfig.h, 416
 - CPU_PCIE_CONFIG_REVISION, 418
 - CPU_PCIE_EQ_METHOD, 418
 - CpuPcieEqDefault, 418
 - CpuPcieEqHardware, 418
 - CpuPcieEqStaticCoeff, 418
- CpuPcieEqDefault
 - CpuPcieConfig.h, 418
- CpuPcieEqHardware
 - CpuPcieConfig.h, 418
- CpuPcieEqStaticCoeff
 - CpuPcieConfig.h, 418
- CpuPowerMgmtBasicConfig.h, 419
- CpuPowerMgmtCustomConfig.h, 419
 - MAX_CUSTOM_CTDTP_ENTRIES, 421
- CpuPowerMgmtPsysConfig.h, 421
- CpuPowerMgmtTestConfig.h, 422
 - CUSTOM_POWER_UNIT, 423
 - PowerUnit125MilliWatts, 423
 - PowerUnitWatts, 423
- CpuPowerMgmtVrConfig.h, 423
 - MAX_NUM_VRS, 424
- CpuRatio
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 93
- CpuReplacementTimeout
 - AMT_DXE_CONFIG, 75
- CpuSecurityPreMemConfig.h, 424
- CpuStart
 - EFI_MMRAM_DESCRIPTOR, 167
- CpuTestConfig.h, 425
- CpuTxtConfig.h, 426
- CrashLogEnable
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 94
- CrashLogGprs
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 94
- CreateConfigBlockTable
 - ConfigBlockLib.h, 412
- Crid
 - PCH_GENERAL_CONFIG, 265
- CsmFlag
 - SI_CONFIG, 342
- CstateLatencyControl0TimeUnit
 - CPU_POWER_MGMT_TEST_CONFIG, 129
- CUSTOM_POWER_UNIT
 - CpuPowerMgmtTestConfig.h, 423
- CUSTOM_PROFILE
 - MemoryConfig.h, 459
- CustomizedSsid
 - SI_CONFIG, 342
- CustomizedSvid
 - SI_CONFIG, 342
- CustomPowerUnit
 - CPU_POWER_MGMT_TEST_CONFIG, 130
- DCC
 - MEMORY_CONFIGURATION, 236
- DciClkEnable
 - PCH_DCI_PREMEM_CONFIG, 257
- DciConfig.h, 427
- DciDbcMode
 - PCH_DCI_PREMEM_CONFIG, 257
- DciEn
 - PCH_DCI_PREMEM_CONFIG, 257
- DciUsb3TypecUfpDbg
 - PCH_DCI_PREMEM_CONFIG, 258
- DDI_CONFIGURATION, 145
- DEFAULT_SPD
 - MemoryConfig.h, 459
- DeviceResetDelay
 - RST_HARDWARE_REMAPPED_STORAGE_CONFIG, 318
- Direction
 - GPIO_CONFIG, 180
- DisableDsxAcPresentPulldown
 - PCH_PM_CONFIG, 284
- DisableDynamicTccoldHandshake
 - PCH_GENERAL_PREMEM_CONFIG, 267
- DisableEnergyReport
 - PCH_PM_CONFIG, 285
- DisableNativePowerButton
 - PCH_PM_CONFIG, 285
- DisablePerAtomMask
 - OVERCLOCKING_PREMEM_CONFIG, 248
- DisablePerCoreMask
 - OVERCLOCKING_PREMEM_CONFIG, 248
- DMI_HW_WIDTH_CONTROL, 146
- DmiGen3EqPh2Enable
 - CPU_DMI_PREMEM_CONFIG, 100

- PCIE_PEI_PREMEM_CONFIG, 306
- DmiGen3EqPh3Method
 - CPU_DMI_PREMEM_CONFIG, 100
 - PCIE_PEI_PREMEM_CONFIG, 306
- DmiGen3ProgramStaticEq
 - CPU_DMI_PREMEM_CONFIG, 100
 - PCIE_PEI_PREMEM_CONFIG, 307
- DmiGen3RxCtlePeaking
 - CPU_DMI_PREMEM_CONFIG, 100
 - PCIE_PEI_PREMEM_CONFIG, 307
- DmiHaAWC
 - THERMAL_CONFIG, 363
- DmiMaxLinkSpeed
 - CPU_DMI_PREMEM_CONFIG, 101
 - PCIE_PEI_PREMEM_CONFIG, 307
- DmiPowerReduction
 - PCH_DMI_CONFIG, 260
- DualTauBoost
 - CPU_POWER_MGMT_BASIC_CONFIG, 115
- DXE_ERROR
 - PiMultiPhase.h, 484
- ECT
 - MEMORY_CONFIGURATION, 236
- EFI_AP_PROCEDURE
 - PiMultiPhase.h, 485
- EFI_AP_PROCEDURE2
 - PiMultiPhase.h, 486
- EFI_AUTH_STATUS_PLATFORM_OVERRIDE
 - PiMultiPhase.h, 484
- EFI_HOB_CPU, 147
 - Header, 148
- EFI_HOB_FIRMWARE_VOLUME, 148
 - Header, 149
- EFI_HOB_FIRMWARE_VOLUME2, 149
 - Header, 150
- EFI_HOB_FIRMWARE_VOLUME3, 151
 - ExtractedFv, 151
 - FileName, 152
 - FvName, 152
 - Header, 152
- EFI_HOB_GENERIC_HEADER, 153
- EFI_HOB_GUID_TYPE, 153
 - Header, 154
- EFI_HOB_HANDOFF_INFO_TABLE, 154
 - EfiMemoryTop, 155
 - Header, 155
 - Version, 155
- EFI_HOB_MEMORY_ALLOCATION, 156
 - Header, 157
- EFI_HOB_MEMORY_ALLOCATION_BSP_STORE, 157
 - Header, 158
- EFI_HOB_MEMORY_ALLOCATION_HEADER, 158
 - MemoryBaseAddress, 159
 - MemoryType, 159
 - Name, 159
- EFI_HOB_MEMORY_ALLOCATION_MODULE, 160
 - Header, 161
- EFI_HOB_MEMORY_ALLOCATION_STACK, 161
 - Header, 162
- EFI_HOB_MEMORY_POOL, 162
 - Header, 163
- EFI_HOB_RESOURCE_DESCRIPTOR, 163
 - Header, 164
 - Owner, 164
- EFI_HOB_UEFI_CAPSULE, 165
 - BaseAddress, 165
- EFI_IP_ADDRESS, 166
- EFI_IPv4_ADDRESS
 - UefiBaseType.h, 517
- EFI_IPv6_ADDRESS
 - UefiBaseType.h, 517
- EFI_MAC_ADDRESS, 167
- EFI_MMRAM_DESCRIPTOR, 167
 - CpuStart, 167
 - PhysicalStart, 168
 - RegionState, 168
- EFI_PAGES_TO_SIZE
 - UefiBaseType.h, 516
- EFI_PEI_HOB_POINTERS, 169
- EFI_PEI_MP_SERVICES_ENABLEDISABLEAP
 - MpServices.h, 461
- EFI_PEI_MP_SERVICES_GET_NUMBER_OF_PROCESSORS
 - MpServices.h, 463
- EFI_PEI_MP_SERVICES_GET_PROCESSOR_INFO
 - MpServices.h, 464
- EFI_PEI_MP_SERVICES_STARTUP_ALL_APS
 - MpServices.h, 464
- EFI_PEI_MP_SERVICES_STARTUP_THIS_AP
 - MpServices.h, 465
- EFI_PEI_MP_SERVICES_SWITCH_BSP
 - MpServices.h, 466
- EFI_PEI_MP_SERVICES_WHOAMI
 - MpServices.h, 466
- EFI_SIZE_TO_PAGES
 - UefiBaseType.h, 517
- EFI_TIME, 169
- EfiMemoryTop
 - EFI_HOB_HANDOFF_INFO_TABLE, 155
- EightCoreRatioLimit
 - CPU_POWER_MGMT_BASIC_CONFIG, 115
- ElectricalConfig
 - GPIO_CONFIG, 180
- Enable
 - GBE_CONFIG, 178
 - ISH_PREMEM_CONFIG, 214
 - PCH_MEMORY_THROTTLING, 275
 - PCH_SATA_PORT_CONFIG, 290
 - PCH_SMBUS_PREMEM_CONFIG, 292
 - RST_HARDWARE_REMAPPED_STORAGE_CONFIG, 318
 - SATA_CONFIG, 333
 - XDCI_CONFIG, 383
- Enable8254ClockGating
 - PCH_IOAPIC_CONFIG, 271
- Enable8254ClockGatingOnS3
 - PCH_IOAPIC_CONFIG, 271

- EnableC6Dram
 - CPU_SECURITY_PREMEM_CONFIG, 140
- EnableDmaBuffer
 - VTD_CONFIG, 381
- EnableEpbPeciOverride
 - CPU_POWER_MGMT_BASIC_CONFIG, 115
- EnableFastMsrHwpReq
 - CPU_POWER_MGMT_BASIC_CONFIG, 116
- EnableFastVmode
 - CPU_POWER_MGMT_VR_CONFIG, 135
- EnableHwpAutoEppGrouping
 - CPU_POWER_MGMT_BASIC_CONFIG, 116
- EnableHwpAutoPerCorePstate
 - CPU_POWER_MGMT_BASIC_CONFIG, 116
- EnableHwpScalabilityTracking
 - CPU_POWER_MGMT_BASIC_CONFIG, 116
- EnableItbm
 - CPU_POWER_MGMT_BASIC_CONFIG, 117
- EnableItbmDriver
 - CPU_POWER_MGMT_BASIC_CONFIG, 117
- EnableMode
 - TRACE_HUB_CONFIG, 366
- EnablePerCorePState
 - CPU_POWER_MGMT_BASIC_CONFIG, 117
- EnablePort8xhDecode
 - PCIE_COMMON_CONFIG, 297
- EnableSgx
 - CPU_SECURITY_PREMEM_CONFIG, 140
- ENCODE_ERROR
 - Base.h, 397
- ENCODE_WARNING
 - Base.h, 397
- EqPh23Bypass
 - PCIE_LINK_EQ_PLATFORM_SETTINGS, 303
- EqPh3Bypass
 - PCIE_LINK_EQ_PLATFORM_SETTINGS, 303
- EqPh3LaneParam
 - CPU_PCIE_CONFIG, 102
- EsataSpeedLimit
 - SATA_CONFIG, 333
- EspiConfig.h, 427
- ExtractedFv
 - EFI_HOB_FIRMWARE_VOLUME3, 151
- ExtSync
 - PCH_PCIE_ROOT_PORT_CONFIG, 279
- ExtVnnRailSx
 - PCH_FIVR_CONFIG, 264
- FALSE
 - Base.h, 398
- FastThrottleThreshold
 - OVERCLOCKING_PREMEM_CONFIG, 249
- FClkFrequency
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 94
- FileName
 - EFI_HOB_FIRMWARE_VOLUME3, 152
- FIRMWARE_VERSION, 170
- FIRMWARE_VERSION_INFO, 170
- FIRMWARE_VERSION_INFO_HOB, 171
- FirmwareVersionInfoHob.h, 428
- FirstDimmBitMask
 - MEMORY_CONFIGURATION, 236
 - SA_MISC_PEI_PREMEM_CONFIG, 329
- FirstDimmBitMaskEcc
 - MEMORY_CONFIGURATION, 237
 - SA_MISC_PEI_PREMEM_CONFIG, 329
- FiveCoreRatioLimit
 - CPU_POWER_MGMT_BASIC_CONFIG, 117
- FIVR_EXT_RAIL_CONFIG, 172
 - IccMax, 172
 - SupportedVoltageStates, 173
 - Voltage, 173
- FIVR_VCCIN_AUX_CONFIG, 173
 - LowToHighCurModeVolTranTime, 174
 - OffToHighCurModeVolTranTime, 174
 - RetToHighCurModeVolTranTime, 174
 - RetToLowCurModeVolTranTime, 175
- FivrConfig.h, 429
 - PCH_FIVR_CONFIG_REVISION, 430
- FivrRfiFrequency
 - CPU_POWER_MGMT_VR_CONFIG, 135
- FivrSpreadSpectrum
 - CPU_POWER_MGMT_VR_CONFIG, 135
- ForceLtrOverride
 - PCIE_DEVICE_OVERRIDE, 299
- FourCoreRatioLimit
 - CPU_POWER_MGMT_BASIC_CONFIG, 118
- FreqLimitMixedConfig_1R1R_16GB
 - MEMORY_CONFIGURATION, 237
- FreqLimitMixedConfig_1R1R_8GB
 - MEMORY_CONFIGURATION, 237
- FreqLimitMixedConfig_1R1R_8GB_16GB
 - MEMORY_CONFIGURATION, 237
- FreqLimitMixedConfig_2R2R
 - MEMORY_CONFIGURATION, 238
- FSP_ERROR_INFO_HOB, 175
- FspErrorInfo.h, 431
- FspFixedPcds.h, 431
- FspInfoHob.h, 432
- FSPM_ARCH_CONFIG_PPI, 176
- FspmArchConfigPpi.h, 433
- FvName
 - EFI_HOB_FIRMWARE_VOLUME3, 152
- GBE_CONFIG, 177
 - Enable, 178
- GbeConfig.h, 434
- Gen4EqPh3Method
 - CPU_PCIE_ROOT_PORT_CONFIG, 106
- GetConfigBlock
 - ConfigBlockLib.h, 413
- GlobalResetMasksOverride
 - PCH_PM_CONFIG, 285
- GlobalSmi
 - PCH_LOCK_DOWN_CONFIG, 274
- GNA_CONFIG, 178
 - GnaEnable, 179
- GnaConfig.h, 434

- GnaEnable
 - GNA_CONFIG, [179](#)
- GPIO_CONFIG, [179](#)
 - Direction, [180](#)
 - ElectricalConfig, [180](#)
 - HostSoftPadOwn, [181](#)
 - InterruptConfig, [181](#)
 - LockConfig, [181](#)
 - OutputState, [181](#)
 - PadMode, [182](#)
 - PowerConfig, [182](#)
- GPIO_DIRECTION
 - GpioConfig.h, [437](#)
- GPIO_ELECTRICAL_CONFIG
 - GpioConfig.h, [437](#)
- GPIO_HARDWARE_DEFAULT
 - GpioConfig.h, [438](#)
- GPIO_HOSTSW_OWN
 - GpioConfig.h, [438](#)
- GPIO_INT_CONFIG
 - GpioConfig.h, [438](#)
- GPIO_LOCK_CONFIG
 - GpioConfig.h, [439](#)
- GPIO_OTHER_CONFIG
 - GpioConfig.h, [439](#)
- GPIO_OUTPUT_STATE
 - GpioConfig.h, [440](#)
- GPIO_PAD_MODE
 - GpioConfig.h, [440](#)
- GPIO_RESET_CONFIG
 - GpioConfig.h, [440](#)
- GpioConfig.h, [435](#)
 - GPIO_DIRECTION, [437](#)
 - GPIO_ELECTRICAL_CONFIG, [437](#)
 - GPIO_HARDWARE_DEFAULT, [438](#)
 - GPIO_HOSTSW_OWN, [438](#)
 - GPIO_INT_CONFIG, [438](#)
 - GPIO_LOCK_CONFIG, [439](#)
 - GPIO_OTHER_CONFIG, [439](#)
 - GPIO_OUTPUT_STATE, [440](#)
 - GPIO_PAD_MODE, [440](#)
 - GPIO_RESET_CONFIG, [440](#)
 - GpioDirDefault, [437](#)
 - GpioDirIn, [437](#)
 - GpioDirInInv, [437](#)
 - GpioDirInInvOut, [437](#)
 - GpioDirInOut, [437](#)
 - GpioDirNone, [437](#)
 - GpioDirOut, [437](#)
 - GpioDswReset, [442](#)
 - GpioHardwareDefault, [438](#)
 - GpioHostDeepReset, [442](#)
 - GpioHostOwnAcpi, [438](#)
 - GpioHostOwnDefault, [438](#)
 - GpioHostOwnGpio, [438](#)
 - GpioIntApic, [439](#)
 - GpioIntBothEdge, [439](#)
 - GpioIntDefault, [439](#)
 - GpioIntDis, [439](#)
 - GpioIntEdge, [439](#)
 - GpioIntLevel, [439](#)
 - GpioIntLvIEdgDis, [439](#)
 - GpioIntNmi, [439](#)
 - GpioIntSci, [439](#)
 - GpioIntSmi, [439](#)
 - GpioLockDefault, [439](#)
 - GpioNoTolerance1v8, [437](#)
 - GpioOutDefault, [440](#)
 - GpioOutHigh, [440](#)
 - GpioOutLow, [440](#)
 - GpioOutputStateLock, [439](#)
 - GpioPadConfigLock, [439](#)
 - GpioPlatformReset, [442](#)
 - GpioResetDeep, [442](#)
 - GpioResetDefault, [442](#)
 - GpioResetNormal, [442](#)
 - GpioResetPwrGood, [442](#)
 - GpioResetResume, [442](#)
 - GpioResumeReset, [442](#)
 - GpioRxRaw1Default, [440](#)
 - GpioRxRaw1Dis, [440](#)
 - GpioRxRaw1En, [440](#)
 - GpioTermDefault, [437](#)
 - GpioTermNative, [437](#)
 - GpioTermNone, [437](#)
 - GpioTermWpd20K, [437](#)
 - GpioTermWpd5K, [437](#)
 - GpioTermWpu1K, [437](#)
 - GpioTermWpu1K2K, [437](#)
 - GpioTermWpu20K, [437](#)
 - GpioTermWpu2K, [437](#)
 - GpioTermWpu5K, [437](#)
 - GpioTolerance1v8, [437](#)
- GpioDirDefault
 - GpioConfig.h, [437](#)
- GpioDirIn
 - GpioConfig.h, [437](#)
- GpioDirInInv
 - GpioConfig.h, [437](#)
- GpioDirInInvOut
 - GpioConfig.h, [437](#)
- GpioDirInOut
 - GpioConfig.h, [437](#)
- GpioDirNone
 - GpioConfig.h, [437](#)
- GpioDirOut
 - GpioConfig.h, [437](#)
- GpioDswReset
 - GpioConfig.h, [442](#)
- GpioHardwareDefault
 - GpioConfig.h, [438](#)
- GpioHostDeepReset
 - GpioConfig.h, [442](#)
- GpioHostOwnAcpi
 - GpioConfig.h, [438](#)
- GpioHostOwnDefault

- GpioConfig.h, [438](#)
- GpioHostOwnGpio
 - GpioConfig.h, [438](#)
- GpioIntApic
 - GpioConfig.h, [439](#)
- GpioIntBothEdge
 - GpioConfig.h, [439](#)
- GpioIntDefault
 - GpioConfig.h, [439](#)
- GpioIntDis
 - GpioConfig.h, [439](#)
- GpioIntEdge
 - GpioConfig.h, [439](#)
- GpioIntLevel
 - GpioConfig.h, [439](#)
- GpioIntLvlEdgDis
 - GpioConfig.h, [439](#)
- GpioIntNmi
 - GpioConfig.h, [439](#)
- GpioIntSci
 - GpioConfig.h, [439](#)
- GpioIntSmi
 - GpioConfig.h, [439](#)
- GpioLockDefault
 - GpioConfig.h, [439](#)
- GpioNoTolerance1v8
 - GpioConfig.h, [437](#)
- GpioOutDefault
 - GpioConfig.h, [440](#)
- GpioOutHigh
 - GpioConfig.h, [440](#)
- GpioOutLow
 - GpioConfig.h, [440](#)
- GpioOutputStateLock
 - GpioConfig.h, [439](#)
- GpioOverride
 - PCH_GENERAL_PREMEM_CONFIG, [267](#)
- GpioPadConfigLock
 - GpioConfig.h, [439](#)
- GpioPlatformReset
 - GpioConfig.h, [442](#)
- GpioResetDeep
 - GpioConfig.h, [442](#)
- GpioResetDefault
 - GpioConfig.h, [442](#)
- GpioResetNormal
 - GpioConfig.h, [442](#)
- GpioResetPwrGood
 - GpioConfig.h, [442](#)
- GpioResetResume
 - GpioConfig.h, [442](#)
- GpioResumeReset
 - GpioConfig.h, [442](#)
- GpioRxRaw1Default
 - GpioConfig.h, [440](#)
- GpioRxRaw1Dis
 - GpioConfig.h, [440](#)
- GpioRxRaw1En
 - GpioConfig.h, [440](#)
- GpioSampleDef.h, [442](#)
- GpioTermDefault
 - GpioConfig.h, [437](#)
- GpioTermNative
 - GpioConfig.h, [437](#)
- GpioTermNone
 - GpioConfig.h, [437](#)
- GpioTermWpd20K
 - GpioConfig.h, [437](#)
- GpioTermWpd5K
 - GpioConfig.h, [437](#)
- GpioTermWpu1K
 - GpioConfig.h, [437](#)
- GpioTermWpu1K2K
 - GpioConfig.h, [437](#)
- GpioTermWpu20K
 - GpioConfig.h, [437](#)
- GpioTermWpu2K
 - GpioConfig.h, [437](#)
- GpioTermWpu5K
 - GpioConfig.h, [437](#)
- GpioTolerance1v8
 - GpioConfig.h, [437](#)
- GRAPHICS_DXE_CONFIG, [183](#)
 - IgdDvmtMemSize, [184](#)
- GRAPHICS_PEI_CONFIG, [185](#)
 - CdClock, [187](#)
- GRAPHICS_PEI_PREMEM_CONFIG, [187](#)
 - InternalGraphics, [189](#)
- GraphicsConfig.h, [443](#)
- GUID, [190](#)
- HDA_LINK_DMIC, [190](#)
- HDA_LINK_HDA, [191](#)
- HDA_LINK_SNDW, [191](#)
- HDA_LINK_SSP, [192](#)
- HDA_VERB_TABLE_HEADER, [192](#)
- HDAUDIO_CONFIG, [193](#)
 - VerbTableEntryNum, [194](#)
 - VerbTablePtr, [194](#)
- HDAUDIO_DXE_CONFIG, [195](#)
- HDAUDIO_PREMEM_CONFIG, [196](#)
 - AudioLinkDmic, [197](#)
 - AudioLinkHda, [197](#)
 - AudioLinkSndw, [198](#)
 - AudioLinkSsp, [198](#)
- HdAudioConfig.h, [444](#)
- HdcControl
 - CPU_POWER_MGMT_BASIC_CONFIG, [118](#)
- Header
 - EFI_HOB_CPU, [148](#)
 - EFI_HOB_FIRMWARE_VOLUME, [149](#)
 - EFI_HOB_FIRMWARE_VOLUME2, [150](#)
 - EFI_HOB_FIRMWARE_VOLUME3, [152](#)
 - EFI_HOB_GUID_TYPE, [154](#)
 - EFI_HOB_HANDOFF_INFO_TABLE, [155](#)
 - EFI_HOB_MEMORY_ALLOCATION, [157](#)

- EFI_HOB_MEMORY_ALLOCATION_BSP_STORE, 158
- EFI_HOB_MEMORY_ALLOCATION_MODULE, 161
- EFI_HOB_MEMORY_ALLOCATION_STACK, 162
- EFI_HOB_MEMORY_POOL, 163
- EFI_HOB_RESOURCE_DESCRIPTOR, 164
- PCH_SMBUS_PREMEM_CONFIG, 292
- Heci3Enabled
 - ME_PEI_CONFIG, 220
- HideUnConfigureMeConfirm
 - AMT_DXE_CONFIG, 75
- HidOverSpi
 - ThcConfig.h, 510
- HostSoftPadOwn
 - GPIO_CONFIG, 181
- HSIO_PARAMETERS, 198
 - HsioCtrlAdaptOffsetCfg, 200
- HsioCtrlAdaptOffsetCfg
 - HSIO_PARAMETERS, 200
- Hwp
 - CPU_POWER_MGMT_BASIC_CONFIG, 118
- HYBRID_GRAPHICS_CONFIG, 200
- HYBRID_STORAGE_CONFIG, 202
- HybridGraphicsConfig.h, 446
- HybridStorageConfig.h, 447
- I2C_PIN_MUX, 203
- lalcMax
 - OVERCLOCKING_PREMEM_CONFIG, 249
- lccLimit
 - CPU_POWER_MGMT_VR_CONFIG, 136
- lccMax
 - FIVR_EXT_RAIL_CONFIG, 172
- ledSize
 - SA_MISC_PEI_PREMEM_CONFIG, 329
- IEH_CONFIG, 203
- lehConfig.h, 448
- lgdDvmtMemSize
 - GRAPHICS_DXE_CONFIG, 184
- ImguClkOutEn
 - IPU_PREMEM_CONFIG, 207
- InfDfxPwrEnable
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 94
- InitPcieAspmAfterOprom
 - PCIE_PEI_PREMEM_CONFIG, 308
- InternalGraphics
 - GRAPHICS_PEI_PREMEM_CONFIG, 189
- InterruptConfig
 - GPIO_CONFIG, 181
- InterruptConfig.h, 450
 - PCH_INT_PIN, 451
 - PchNoInt, 451
- IoApicConfig.h, 451
- loeDebugEn
 - PCH_GENERAL_PREMEM_CONFIG, 268
- IOM_AUX_ORI_PAD_CONFIG, 204
- IOM_INTERFACE_CONFIG, 205
- IPU_PREMEM_CONFIG, 205
 - ImguClkOutEn, 207
 - IpuEnable, 207
 - IpuEnable
 - IPU_PREMEM_CONFIG, 207
 - IpuPreMemConfig.h, 452
 - IPv4_ADDRESS, 207
 - IPv6_ADDRESS, 208
 - ISH_CONFIG, 208
 - ISH_GP, 209
 - ISH_GPIO_CONFIG, 210
 - PadTermination, 210
 - PinMux, 210
 - ISH_I2C, 211
 - ISH_I2C_PIN_CONFIG, 212
 - ISH_I3C_CONFIG, 213
 - ISH_PREMEM_CONFIG, 214
 - Enable, 214
 - ISH_SPI, 215
 - ISH_SPI_PIN_CONFIG, 216
 - ISH_UART, 217
 - ISH_UART_PIN_CONFIG, 218
 - IshConfig.h, 453
 - ITbtForcePowerOnTimeoutInMs
 - _ITBT_GENERIC_CONFIG, 64
- KeepEarlyTrace
 - PCH_DCI_PREMEM_CONFIG, 258
- L1sCommonModeRestoreTime
 - PCIE_DEVICE_OVERRIDE, 299
- L1sTpowerOnScale
 - PCIE_DEVICE_OVERRIDE, 299
- L1sTpowerOnValue
 - PCIE_DEVICE_OVERRIDE, 299
- L1SubstatesCapMask
 - PCIE_DEVICE_OVERRIDE, 300
- L1SubstatesCapOffset
 - PCIE_DEVICE_OVERRIDE, 300
- LatchEventsC10Exit
 - PCH_PM_CONFIG, 285
- LegacyIoLowLatency
 - PCH_GENERAL_CONFIG, 265
- LgmrEnable
 - PCH_ESPI_CONFIG, 262
- LinkDownGpios
 - CPU_PCIE_RP_PREMEM_CONFIG, 108
- LocalTransmitterOverrideEnable
 - PCIE_LINK_EQ_PLATFORM_SETTINGS, 303
- LockConfig
 - GPIO_CONFIG, 181
- LockDownConfig.h, 454
- LowToHighCurModeVolTranTime
 - FIVR_VCCIN_AUX_CONFIG, 174
- LpmS0ixSubStateEnable
 - PCH_PM_CONFIG, 286
- LtrOverrideEnable
 - USB_CONFIG, 376
- ManageabilityMode

- AMT_PEI_CONFIG, 78
- MAX
 - Base.h, 398
- MAX_CUSTOM_CTDP_ENTRIES
 - CpuPowerMgmtCustomConfig.h, 421
- MAX_NUM_VRS
 - CpuPowerMgmtVrConfig.h, 424
- ME_PEI_CONFIG, 219
 - Heci3Enabled, 220
 - MeUnconfigOnRtcClear, 220
- ME_PEI_PREMEM_CONFIG, 221
 - SkipMbpHob, 223
- MebxDebugMsg
 - AMT_DXE_CONFIG, 75
- MebxGraphicsMode
 - AMT_DXE_CONFIG, 75
- MebxNonUiTextMode
 - AMT_DXE_CONFIG, 76
- MebxUiTextMode
 - AMT_DXE_CONFIG, 76
- MEMORY_CONFIG_NO_CRC, 223
 - SerialDebugLevel, 224
- MEMORY_CONFIGURATION, 225
 - ChHashInterleaveBit, 236
 - DCC, 236
 - ECT, 236
 - FirstDimmBitMask, 236
 - FirstDimmBitMaskEcc, 237
 - FreqLimitMixedConfig_1R1R_16GB, 237
 - FreqLimitMixedConfig_1R1R_8GB, 237
 - FreqLimitMixedConfig_1R1R_8GB_16GB, 237
 - FreqLimitMixedConfig_2R2R, 238
 - WRDSUDT, 238
- MemoryBaseAddress
 - EFI_HOB_MEMORY_ALLOCATION_HEADER, 159
- MemoryConfig.h, 455
 - CUSTOM_PROFILE, 459
 - DEFAULT_SPD, 459
 - SA_SPD, 459
 - SPD_BOOT_MODE, 459
 - SpdBootModeMax, 459
 - SpdCold, 459
 - SpdFast, 459
 - SPDProfileMax, 459
 - SpdS3, 459
 - SpdWarm, 459
 - XMP_PROFILE_1, 459
 - XMP_PROFILE_2, 459
 - XMP_PROFILE_3, 459
 - XMP_USER_PROFILE_4, 459
 - XMP_USER_PROFILE_5, 459
- MemoryLock
 - RTC_CONFIG, 320
- MemoryType
 - EFI_HOB_MEMORY_ALLOCATION_HEADER, 159
- MemReg0Size
- TRACE_HUB_CONFIG, 366
- MePeiConfig.h, 459
- MeUnconfigOnRtcClear
 - ME_PEI_CONFIG, 220
- MIN
 - Base.h, 399
- MmramReservedSize
 - _EFI_MM_RESERVED_MMRAM_REGION, 62
- MmramReservedStart
 - _EFI_MM_RESERVED_MMRAM_REGION, 62
- Mode
 - CNVI_CONFIG, 82
- ModPhySusPgEnable
 - PCH_PM_CONFIG, 286
- MpServices.h, 460
 - EFI_PEI_MP_SERVICES_ENABLEDISABLEAP, 461
 - EFI_PEI_MP_SERVICES_GET_NUMBER_OF_PROCESSORS, 463
 - EFI_PEI_MP_SERVICES_GET_PROCESSOR_INFO, 464
 - EFI_PEI_MP_SERVICES_STARTUP_ALL_APS, 464
 - EFI_PEI_MP_SERVICES_STARTUP_THIS_AP, 465
 - EFI_PEI_MP_SERVICES_SWITCH_BSP, 466
 - EFI_PEI_MP_SERVICES_WHOAMI, 466
- MvcEnabled
 - PCH_PCIE_ROOT_PORT_CONFIG, 280
- Name
 - EFI_HOB_MEMORY_ALLOCATION_HEADER, 159
- NonSnoopLatency
 - PCIE_DEVICE_OVERRIDE, 300
- NORETURN
 - Base.h, 399
- NumberOfSsidTableEntry
 - SI_CONFIG, 342
- OcSupport
 - OVERCLOCKING_PREMEM_CONFIG, 249
- OemResolutionSettingsRsvd
 - AMT_DXE_CONFIG, 76
- OFFSET_OF
 - Base.h, 399
- OffToHighCurModeVolTranTime
 - FIVR_VCCIN_AUX_CONFIG, 174
- OneCoreRatioLimit
 - CPU_POWER_MGMT_BASIC_CONFIG, 119
- OsidleEnable
 - PCH_PM_CONFIG, 286
- OutputState
 - GPIO_CONFIG, 181
- OVERCLOCKING_PREMEM_CONFIG, 239
 - AtomClusterRatio, 244
 - AtomL2MaxOcRatio, 244
 - AtomL2VoltageAdaptive, 244
 - AtomL2VoltageMode, 245

- AtomL2VoltageOffset, [245](#)
- AtomL2VoltageOverride, [245](#)
- Avx2VoltageScaleFactor, [245](#)
- Avx512VoltageScaleFactor, [246](#)
- CoreMaxOcRatio, [246](#)
- CoreVfConfigScope, [246](#)
- CoreVfPointOffset, [247](#)
- CoreVfPointOffsetMode, [247](#)
- CoreVoltageAdaptive, [247](#)
- CoreVoltageMode, [247](#)
- CoreVoltageOffset, [248](#)
- CoreVoltageOverride, [248](#)
- DisablePerAtomMask, [248](#)
- DisablePerCoreMask, [248](#)
- FastThrottleThreshold, [249](#)
- IalccMax, [249](#)
- OcSupport, [249](#)
- PerCoreHtDisable, [249](#)
- PerCoreVoltageOffset, [250](#)
- PllMaxBandingRatio, [250](#)
- PvdMode, [250](#)
- PvdRatioThreshold, [250](#)
- RingDownBin, [251](#)
- RingMaxOcRatio, [251](#)
- RingVfPointOffset, [251](#)
- RingVfPointOffsetMode, [251](#)
- RingVoltageAdaptive, [252](#)
- RingVoltageMode, [252](#)
- RingVoltageOffset, [252](#)
- RingVoltageOverride, [252](#)
- SaExtraTurboVoltage, [253](#)
- SaVoltageMode, [253](#)
- SaVoltageOverride, [253](#)
- TjMaxOffset, [253](#)
- TvbDownBinsTempThreshold0, [254](#)
- TvbDownBinsTempThreshold1, [254](#)
- TvbRatioClipping, [254](#)
- TvbTempThreshold0, [254](#)
- TvbTempThreshold1, [255](#)
- TvbVoltageOptimization, [255](#)
- OverclockingConfig.h, [467](#)
- OverCurrentEnable
 - USB_CONFIG, [376](#)
- OverCurrentPin
 - USB2_PORT_CONFIG, [372](#)
 - USB3_PORT_CONFIG, [374](#)
- Owner
 - EFI_HOB_RESOURCE_DESCRIPTOR, [164](#)
- P2sbConfig.h, [468](#)
- PadMode
 - GPIO_CONFIG, [182](#)
- PadTermination
 - ISH_GPIO_CONFIG, [210](#)
 - SERIAL_IO_I2C_CONFIG, [337](#)
- PCH_DCI_PREMEM_CONFIG, [256](#)
 - DciClkEnable, [257](#)
 - DciDbcMode, [257](#)
 - DciEn, [257](#)
 - DciUsb3TypecUfpDbg, [258](#)
 - KeepEarlyTrace, [258](#)
- PCH_DEVICE_INTERRUPT_CONFIG, [258](#)
- PCH_DMI_CONFIG, [259](#)
 - DmiPowerReduction, [260](#)
- PCH_DMI_EQ_PARAM, [261](#)
- PCH_ESPI_CONFIG, [261](#)
 - BmeMasterSlaveEnabled, [262](#)
 - LgmrEnable, [262](#)
- PCH_FIVR_CONFIG, [263](#)
 - ExtVnnRailSx, [264](#)
- PCH_FIVR_CONFIG_REVISION
 - FivrConfig.h, [430](#)
- PCH_GENERAL_CONFIG, [264](#)
 - AcpiL6dPmeHandling, [265](#)
 - Crid, [265](#)
 - LegacyIoLowLatency, [265](#)
- PCH_GENERAL_PREMEM_CONFIG, [266](#)
 - DisableDynamicTccoldHandshake, [267](#)
 - GpioOverride, [267](#)
 - IoDebugEn, [268](#)
 - PmodeClkEn, [268](#)
- PCH_INT_PIN
 - InterruptConfig.h, [451](#)
- PCH_INTERRUPT_CONFIG, [269](#)
- PCH_IOAPIC_CONFIG, [270](#)
 - Enable8254ClockGating, [271](#)
 - Enable8254ClockGatingOnS3, [271](#)
- PCH_LOCK_DOWN_CONFIG, [272](#)
 - BiosInterface, [273](#)
 - BiosLock, [273](#)
 - GlobalSmi, [274](#)
 - UnlockGpioPads, [274](#)
- PCH_MEMORY_THROTTLING, [275](#)
 - Enable, [275](#)
 - TsGpioPinSetting, [275](#)
- PCH_P2SB_CONFIG, [276](#)
 - SbAccessUnlock, [277](#)
- PCH_PCIE_CLOCK, [277](#)
- PCH_PCIE_CLOCK_USAGE
 - PchPcieRpConfig.h, [472](#)
- PCH_PCIE_CONFIG, [278](#)
- PCH_PCIE_ROOT_PORT_CONFIG, [279](#)
 - ExtSync, [279](#)
 - MvcEnabled, [280](#)
 - VppPort, [280](#)
- PCH_PCIE_RP_PREMEM_CONFIG, [281](#)
 - RpEnabledMask, [282](#)
- PCH_PM_CONFIG, [282](#)
 - C10DynamicThresholdAdjustment, [284](#)
 - CpuC10GatePinEnable, [284](#)
 - DisableDsxAcpPresentPulldown, [284](#)
 - DisableEnergyReport, [285](#)
 - DisableNativePowerButton, [285](#)
 - GlobalResetMasksOverride, [285](#)
 - LatchEventsC10Exit, [285](#)
 - LpmS0ixSubStateEnable, [286](#)
 - ModPhySusPgEnable, [286](#)

- OsIdleEnable, [286](#)
- PchPwrCycDur, [286](#)
- PciePIISsc, [287](#)
- PmcDbgMsgEn, [287](#)
- PsOnEnable, [287](#)
- PwrBtnOverridePeriod, [288](#)
- S0ixAutoDemotion, [288](#)
- SlpLanLowDc, [288](#)
- SlpStrchSusUp, [288](#)
- ThermTimerDelay, [289](#)
- Usb2PhySusPgEnable, [289](#)
- PCH_RESERVED_PAGE_ROUTE
 - PchGeneralConfig.h, [470](#)
- PCH_SATA_PORT_CONFIG, [289](#)
 - Enable, [290](#)
 - ZpOdd, [291](#)
- PCH_SLP_S4_MIN_ASSERT
 - PmConfig.h, [488](#)
- PCH_SMBUS_PREMEM_CONFIG, [291](#)
 - Enable, [292](#)
 - Header, [292](#)
 - SpdWriteDisable, [293](#)
- PCH_TRACE_HUB_PREMEM_CONFIG, [293](#)
- PCH_WAKE_CONFIG, [294](#)
 - PmeB0S5Dis, [295](#)
- PCH_WDT_PREMEM_CONFIG, [295](#)
- PchClockUsageCpuPcie0
 - PchPcieRpConfig.h, [473](#)
- PchClockUsageUnspecified
 - PchPcieRpConfig.h, [473](#)
- PchCrossThrottling
 - THERMAL_THROTTLE_LEVELS, [365](#)
- PchDmiConfig.h, [469](#)
- PchGeneralConfig.h, [469](#)
 - PCH_RESERVED_PAGE_ROUTE, [470](#)
 - PchReservedPageToLpc, [471](#)
 - PchReservedPageToPcie, [471](#)
- PchHotLevel
 - THERMAL_CONFIG, [363](#)
- PchNoInt
 - InterruptConfig.h, [451](#)
- PchPcieRpConfig.h, [471](#)
 - PCH_PCIE_CLOCK_USAGE, [472](#)
 - PchClockUsageCpuPcie0, [473](#)
 - PchClockUsageUnspecified, [473](#)
- PchPwrCycDur
 - PCH_PM_CONFIG, [286](#)
- PchReservedPageToLpc
 - PchGeneralConfig.h, [471](#)
- PchReservedPageToPcie
 - PchGeneralConfig.h, [471](#)
- PchSlpS4PchTime
 - PmConfig.h, [488](#)
- PCIE_COMMON_CONFIG, [296](#)
 - ComplianceTestMode, [297](#)
 - EnablePort8xhDecode, [297](#)
 - RpFunctionSwap, [297](#)
- PCIE_DEVICE_OVERRIDE, [298](#)
 - ForceLtrOverride, [299](#)
 - L1sCommonModeRestoreTime, [299](#)
 - L1sTpowerOnScale, [299](#)
 - L1sTpowerOnValue, [299](#)
 - L1SubstatesCapMask, [300](#)
 - L1SubstatesCapOffset, [300](#)
 - NonSnoopLatency, [300](#)
 - SnoopLatency, [300](#)
- PCIE_EQ_PARAM, [301](#)
- PCIE_FORM_FACTOR
 - PcieConfig.h, [474](#)
- PCIE_IMR_CONFIG, [301](#)
- PCIE_LINK_EQ_METHOD
 - PcieConfig.h, [475](#)
- PCIE_LINK_EQ_MODE
 - PcieConfig.h, [475](#)
- PCIE_LINK_EQ_PLATFORM_SETTINGS, [302](#)
 - EqPh23Bypass, [303](#)
 - EqPh3Bypass, [303](#)
 - LocalTransmitterOverrideEnable, [303](#)
 - Ph2LocalTransmitterOverridePreset, [304](#)
 - Ph3NumberOfPresetsOrCoefficients, [304](#)
- PCIE_PEI_PREMEM_CONFIG, [305](#)
 - DmiGen3EqPh2Enable, [306](#)
 - DmiGen3EqPh3Method, [306](#)
 - DmiGen3ProgramStaticEq, [307](#)
 - DmiGen3RxCtlePeaking, [307](#)
 - DmiMaxLinkSpeed, [307](#)
 - InitPcieAspmAfterOprom, [308](#)
- PCIE_PREMEM_CONFIG, [308](#)
- PCIE_RP_DXE_CONFIG, [309](#)
 - PcieDeviceOverrideTablePtr, [310](#)
- PcieConfig.h, [473](#)
 - PCIE_FORM_FACTOR, [474](#)
 - PCIE_LINK_EQ_METHOD, [475](#)
 - PCIE_LINK_EQ_MODE, [475](#)
 - PcieLinkEqCoefficientMode, [475](#)
 - PcieLinkEqPresetMode, [475](#)
 - PcieLinkFixedEq, [475](#)
 - PcieLinkHardwareEq, [475](#)
- PcieDeviceOverrideTablePtr
 - CPU_PCIE_CONFIG, [103](#)
 - PCIE_RP_DXE_CONFIG, [310](#)
- PcieLinkEqCoefficientMode
 - PcieConfig.h, [475](#)
- PcieLinkEqPresetMode
 - PcieConfig.h, [475](#)
- PcieLinkFixedEq
 - PcieConfig.h, [475](#)
- PcieLinkHardwareEq
 - PcieConfig.h, [475](#)
- PcieMultipleSegmentEnabled
 - TCSS_MISC_PEI_PREMEM_CONFIG, [351](#)
- PciePIISsc
 - PCH_PM_CONFIG, [287](#)
- PciePreMemConfig.h, [475](#)
- PcieSpeed
 - CPU_PCIE_RP_PREMEM_CONFIG, [109](#)

- PdoProgramming
 - USB_CONFIG, 376
- PeciC10Reset
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 95
- PeciSxReset
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 95
- PegGen3ProgramStaticEq
 - CPU_PCIE_CONFIG, 103
- PegGen4ProgramStaticEq
 - CPU_PCIE_CONFIG, 103
- PEI_ITBT_CONFIG
 - PeiItbtConfig.h, 477
- PeiItbtConfig.h, 476
 - PEI_ITBT_CONFIG, 477
- PeiItbtGenericStructure.h, 478
- PeiPreMemSiDefaultPolicy.h, 479
- PeiSiDefaultPolicy.h, 480
- PerCoreHtDisable
 - OVERCLOCKING_PREMEM_CONFIG, 249
- PerCoreVoltageOffset
 - OVERCLOCKING_PREMEM_CONFIG, 250
- Ph2LocalTransmitterOverridePreset
 - PCIE_LINK_EQ_PLATFORM_SETTINGS, 304
- Ph3NumberOfPresetsOrCoefficients
 - PCIE_LINK_EQ_PLATFORM_SETTINGS, 304
- PhysicalStart
 - EFI_MMRAM_DESCRIPTOR, 168
- PI_ENCODE_ERROR
 - PiMultiPhase.h, 485
- PI_ENCODE_WARNING
 - PiMultiPhase.h, 485
- PiHob.h, 481
- PiMultiPhase.h, 483
 - DXE_ERROR, 484
 - EFI_AP_PROCEDURE, 485
 - EFI_AP_PROCEDURE2, 486
 - EFI_AUTH_STATUS_PLATFORM_OVERRIDE, 484
 - PI_ENCODE_ERROR, 485
 - PI_ENCODE_WARNING, 485
- PinMux
 - ISH_GPIO_CONFIG, 210
- PLATFORM_SEND_EC_COMMAND
 - BiosGuardConfig.h, 408
- PlatformDebugConsent
 - SI_PREMEM_CONFIG, 345
- PIIMaxBandingRatio
 - OVERCLOCKING_PREMEM_CONFIG, 250
- PMC_GLOBAL_RESET_MASK, 311
- PMC_INTERFACE_CONFIG, 311
- PMC_LPM_S0IX_SUB_STATE_EN, 312
 - S0i2p2En, 312
 - S0i3p3En, 312
 - S0i3p4En, 313
- PmcDbgMsgEn
 - PCH_PM_CONFIG, 287
- PmConfig.h, 486
 - PCH_SLP_S4_MIN_ASSERT, 488
- PchSlpS4PchTime, 488
- PmeB0S5Dis
 - PCH_WAKE_CONFIG, 295
- PmodeClkEn
 - PCH_GENERAL_PREMEM_CONFIG, 268
- PmsyncEnable
 - TS_GPIO_PIN_SETTING, 367
- Port
 - USB2_PHY_CONFIG, 371
- PowerConfig
 - GPIO_CONFIG, 182
- PowerLimit1
 - CPU_POWER_MGMT_BASIC_CONFIG, 119
- PowerLimit1Time
 - CPU_POWER_MGMT_BASIC_CONFIG, 119
- PowerLimit2Power
 - CPU_POWER_MGMT_BASIC_CONFIG, 120
- PowerLimit3
 - CPU_POWER_MGMT_BASIC_CONFIG, 120
- PowerLimit4
 - CPU_POWER_MGMT_BASIC_CONFIG, 120
- PowerUnit125MilliWatts
 - CpuPowerMgmtTestConfig.h, 423
- PowerUnitWatts
 - CpuPowerMgmtTestConfig.h, 423
- PpinSupport
 - CPU_CONFIG, 87
- PPM_CUSTOM_RATIO_TABLE, 313
 - StateRatioMax16, 314
- PpmlrmSetting
 - CPU_POWER_MGMT_TEST_CONFIG, 130
- PreBootDmaMask
 - VTD_CONFIG, 381
- ProcessorTraceMemSize
 - CPU_TEST_CONFIG, 142
- PSF_CONFIG, 314
 - TccEnable, 315
- PsfConfig.h, 488
- PsOnEnable
 - PCH_PM_CONFIG, 287
- PvdMode
 - OVERCLOCKING_PREMEM_CONFIG, 250
- PvdRatioThreshold
 - OVERCLOCKING_PREMEM_CONFIG, 250
- PwrBtnOverridePeriod
 - PCH_PM_CONFIG, 288
- RaidDeviceId
 - SATA_CONFIG, 333
- RegionState
 - EFI_MMRAM_DESCRIPTOR, 168
- RemoteAssistance
 - AMT_PEI_CONFIG, 78
- RetToHighCurModeVolTranTime
 - FIVR_VCCIN_AUX_CONFIG, 174
- RetToLowCurModeVolTranTime
 - FIVR_VCCIN_AUX_CONFIG, 175
- RETURN_ADDRESS
 - Base.h, 400

- RETURN_BUFFER_TOO_SMALL
 - Base.h, [400](#)
- RETURN_ERROR
 - Base.h, [400](#)
- RETURNS_TWICE
 - Base.h, [401](#)
- RingDownBin
 - OVERCLOCKING_PREMEM_CONFIG, [251](#)
- RingMaxOcRatio
 - OVERCLOCKING_PREMEM_CONFIG, [251](#)
- RingVfPointOffset
 - OVERCLOCKING_PREMEM_CONFIG, [251](#)
- RingVfPointOffsetMode
 - OVERCLOCKING_PREMEM_CONFIG, [251](#)
- RingVoltageAdaptive
 - OVERCLOCKING_PREMEM_CONFIG, [252](#)
- RingVoltageMode
 - OVERCLOCKING_PREMEM_CONFIG, [252](#)
- RingVoltageOffset
 - OVERCLOCKING_PREMEM_CONFIG, [252](#)
- RingVoltageOverride
 - OVERCLOCKING_PREMEM_CONFIG, [252](#)
- RpEnabledMask
 - CPU_PCIE_RP_PREMEM_CONFIG, [109](#)
 - PCH_PCIE_RP_PREMEM_CONFIG, [282](#)
- RpFunctionSwap
 - PCIE_COMMON_CONFIG, [297](#)
- RST_CONFIG, [316](#)
- RST_HARDWARE_REMAPPED_STORAGE_CONFIG, [318](#)
 - DeviceResetDelay, [318](#)
 - Enable, [318](#)
- RstConfig.h, [490](#)
- Rsvd
 - TCSS_MISC_PEI_PREMEM_CONFIG, [351](#)
- RTC_CONFIG, [319](#)
 - BiosInterfaceLock, [320](#)
 - MemoryLock, [320](#)
- RtcConfig.h, [491](#)
- S0i2p2En
 - PMC_LPM_S0IX_SUB_STATE_EN, [312](#)
- S0i3p3En
 - PMC_LPM_S0IX_SUB_STATE_EN, [312](#)
- S0i3p4En
 - PMC_LPM_S0IX_SUB_STATE_EN, [313](#)
- S0ixAutoDemotion
 - PCH_PM_CONFIG, [288](#)
- SA_ADDRESS_DECODE, [321](#)
- SA_FUNCTION_CALLS, [321](#)
- SA_MEMORY_DQDQS_MAPPING, [324](#)
- SA_MEMORY_FUNCTIONS, [324](#)
- SA_MEMORY_RCOMP, [325](#)
- SA_MISC_PEI_CONFIG, [326](#)
- SA_MISC_PEI_PREMEM_CONFIG, [327](#)
 - FirstDimmBitMask, [329](#)
 - FirstDimmBitMaskEcc, [329](#)
 - IedSize, [329](#)
 - ScanExtGfxForLegacyOpRom, [330](#)
 - SpdAddressTable, [330](#)
 - TsegSize, [330](#)
- SA_SPD
 - MemoryConfig.h, [459](#)
- SA_XDCI_IRQ_INT_CONFIG, [331](#)
- SaExtraTurboVoltage
 - OVERCLOCKING_PREMEM_CONFIG, [253](#)
- SaMiscPeiConfig.h, [492](#)
- SaMiscPeiPreMemConfig.h, [492](#)
- SATA_CONFIG, [332](#)
 - Enable, [333](#)
 - EsataSpeedLimit, [333](#)
 - RaidDeviceId, [333](#)
 - SataMode, [333](#)
 - SpeedLimit, [334](#)
 - ThermalThrottling, [334](#)
- SATA_THERMAL_THROTTLING, [334](#)
- SataConfig.h, [493](#)
- SataMode
 - SATA_CONFIG, [333](#)
- SaVoltageMode
 - OVERCLOCKING_PREMEM_CONFIG, [253](#)
- SaVoltageOverride
 - OVERCLOCKING_PREMEM_CONFIG, [253](#)
- SbAccessUnlock
 - PCH_P2SB_CONFIG, [277](#)
- ScanExtGfxForLegacyOpRom
 - SA_MISC_PEI_PREMEM_CONFIG, [330](#)
- SERIAL_IO_CONFIG, [336](#)
- SERIAL_IO_I2C_CONFIG, [337](#)
 - PadTermination, [337](#)
- SERIAL_IO_I2C_MODE
 - SerialIoDevices.h, [497](#)
- SERIAL_IO_SPI_CONFIG, [338](#)
- SERIAL_IO_SPI_MODE
 - SerialIoDevices.h, [498](#)
- SERIAL_IO_UART_ATTRIBUTES, [339](#)
- SERIAL_IO_UART_CONFIG, [339](#)
- SERIAL_IO_UART_MODE
 - SerialIoDevices.h, [499](#)
- SERIAL_IO_UART_PG
 - SerialIoDevices.h, [499](#)
- SerialDebugLevel
 - MEMORY_CONFIG_NO_CRC, [224](#)
- SerialIoConfig.h, [494](#)
- SerialIoDevices.h, [496](#)
 - SERIAL_IO_I2C_MODE, [497](#)
 - SERIAL_IO_SPI_MODE, [498](#)
 - SERIAL_IO_UART_MODE, [499](#)
 - SERIAL_IO_UART_PG, [499](#)
 - SerialIoUartPgAuto, [500](#)
 - SerialIoUartPgDisabled, [500](#)
 - SerialIoUartPgEnabled, [500](#)
- SerialIoUartPgAuto
 - SerialIoDevices.h, [500](#)
- SerialIoUartPgDisabled
 - SerialIoDevices.h, [500](#)
- SerialIoUartPgEnabled

- SerialIoDevices.h, [500](#)
- SetSecuredRegisterLock
 - CPU_PCIE_CONFIG, [103](#)
- SevenCoreRatioLimit
 - CPU_POWER_MGMT_BASIC_CONFIG, [120](#)
- SI_CONFIG, [340](#)
 - CsmFlag, [342](#)
 - CustomizedSsid, [342](#)
 - CustomizedSvid, [342](#)
 - NumberOfSsidTableEntry, [342](#)
 - SkipBiosDoneWhenFwUpdate, [343](#)
 - SkipSsidProgramming, [343](#)
 - SsidTablePtr, [343](#)
- SI_POLICY_REVISION
 - SiPolicyStruct.h, [502](#)
- SI_PREMEM_CONFIG, [344](#)
 - PlatformDebugConsent, [345](#)
 - SkipOverrideBootModeWhenFwUpdate, [345](#)
- SI_PREMEM_POLICY_REVISION
 - SiPolicyStruct.h, [503](#)
- SiConfig.h, [500](#)
- SIGNATURE_16
 - Base.h, [401](#)
- SIGNATURE_32
 - Base.h, [401](#)
- SIGNATURE_64
 - Base.h, [402](#)
- SiPolicyStruct.h, [501](#)
 - SI_POLICY_REVISION, [502](#)
 - SI_PREMEM_POLICY_REVISION, [503](#)
- SiPreMemConfig.h, [503](#)
- SixCoreRatioLimit
 - CPU_POWER_MGMT_BASIC_CONFIG, [121](#)
- SkipBiosDoneWhenFwUpdate
 - SI_CONFIG, [343](#)
- SkipMbpHob
 - ME_PEI_PREMEM_CONFIG, [223](#)
- SkipOverrideBootModeWhenFwUpdate
 - SI_PREMEM_CONFIG, [345](#)
- SkipSsidProgramming
 - SI_CONFIG, [343](#)
- SlotImplemented
 - CPU_PCIE_RP_PREMEM_CONFIG, [109](#)
- SlpLanLowDc
 - PCH_PM_CONFIG, [288](#)
- SlpStrchSusUp
 - PCH_PM_CONFIG, [288](#)
- SMBIOS_STRUCTURE, [346](#)
- SmbiosType4MaxSpeedOverride
 - CPU_CONFIG, [88](#)
- SmbusConfig.h, [504](#)
- SnoopLatency
 - PCIE_DEVICE_OVERRIDE, [300](#)
- SocDieSscEnable
 - CPU_CONFIG_LIB_PREMEM_CONFIG, [95](#)
- SocDieSscValue
 - CPU_CONFIG_LIB_PREMEM_CONFIG, [96](#)
- SPD_BOOT_MODE
 - MemoryConfig.h, [459](#)
- SPD_DATA_BUFFER, [346](#)
- SPD_OFFSET_TABLE, [346](#)
- SpdAddressTable
 - SA_MISC_PEI_PREMEM_CONFIG, [330](#)
- SpdBootModeMax
 - MemoryConfig.h, [459](#)
- SpdCold
 - MemoryConfig.h, [459](#)
- SpdFast
 - MemoryConfig.h, [459](#)
- SPDProfileMax
 - MemoryConfig.h, [459](#)
- SpdS3
 - MemoryConfig.h, [459](#)
- SpdWarm
 - MemoryConfig.h, [459](#)
- SpdWriteDisable
 - PCH_SMBUS_PREMEM_CONFIG, [293](#)
- SpeedLimit
 - SATA_CONFIG, [334](#)
- SPI_PIN_MUX, [347](#)
- SsidTablePtr
 - SI_CONFIG, [343](#)
- StateRatioMax16
 - PPM_CUSTOM_RATIO_TABLE, [314](#)
- STATIC_ASSERT
 - Base.h, [403](#)
- SupportedVoltageStates
 - FIVR_EXT_RAIL_CONFIG, [173](#)
- SVID_SID_VALUE, [348](#)
- TccActivationOffset
 - CPU_POWER_MGMT_BASIC_CONFIG, [121](#)
- TccEnable
 - PSF_CONFIG, [315](#)
- TccOffsetClamp
 - CPU_POWER_MGMT_BASIC_CONFIG, [121](#)
- TccOffsetTimeWindowForRatl
 - CPU_POWER_MGMT_BASIC_CONFIG, [122](#)
- TCSS_DEVEN_PEI_PREMEM_CONFIG, [348](#)
- TCSS_IOM_ORI_OVERRIDE, [348](#)
- TCSS_IOM_PEI_CONFIG, [349](#)
- TCSS_MISC_PEI_CONFIG, [350](#)
- TCSS_MISC_PEI_PREMEM_CONFIG, [351](#)
 - PcieMultipleSegmentEnabled, [351](#)
 - Rsvd, [351](#)
- TCSS_PCIE_PEI_POLICY, [352](#)
- TCSS_PCIE_PORT_POLICY, [353](#)
- TCSS_PEI_CONFIG, [354](#)
- TCSS_PEI_PREMEM_CONFIG, [355](#)
- TCSS_USBTC_PEI_PERMEM_CONFIG, [356](#)
- TcssPeiConfig.h, [505](#)
- TcssPeiPreMemConfig.h, [507](#)
- TdcTimeWindow
 - CPU_POWER_MGMT_VR_CONFIG, [136](#)
- TELEMETRY_PEI_CONFIG, [357](#)
- TELEMETRY_PEI_PREMEM_CONFIG, [358](#)
- TelemetryPeiConfig.h, [508](#)

- Thc
 - ThcConfig.h, 510
- THC_CONFIG, 359
- THC_HID_OVER_SPI, 360
- THC_MODE
 - ThcConfig.h, 510
- THC_PORT, 361
- THC_PORT_ASSIGNMENT
 - ThcConfig.h, 510
- ThcAssignmentNone
 - ThcConfig.h, 511
- ThcAssignmentThc0
 - ThcConfig.h, 511
- ThcAssignmentThc1
 - ThcConfig.h, 511
- ThcConfig.h, 509
 - HidOverSpi, 510
 - Thc, 510
 - THC_MODE, 510
 - THC_PORT_ASSIGNMENT, 510
 - ThcAssignmentNone, 511
 - ThcAssignmentThc0, 511
 - ThcAssignmentThc1, 511
- THERMAL_CONFIG, 362
 - DmiHaAWC, 363
 - PchHotLevel, 363
 - TTLevels, 363
- THERMAL_THROTTLE_LEVELS, 364
 - PchCrossThrottling, 365
 - TTLock, 365
 - TTState13Enable, 365
- ThermalConfig.h, 511
- ThermalThrottling
 - SATA_CONFIG, 334
- ThermTimerDelay
 - PCH_PM_CONFIG, 289
- ThreeCoreRatioLimit
 - CPU_POWER_MGMT_BASIC_CONFIG, 122
- TjMaxOffset
 - OVERCLOCKING_PREMEM_CONFIG, 253
- TmeEnable
 - CPU_CONFIG_LIB_PREMEM_CONFIG, 96
- TRACE_HUB_CONFIG, 365
 - AetEnabled, 366
 - EnableMode, 366
 - MemReg0Size, 366
- TraceHubConfig.h, 512
- TRUE
 - Base.h, 403
- TS_GPIO_PIN_SETTING, 367
 - PmsyncEnable, 367
- TsegSize
 - SA_MISC_PEI_PREMEM_CONFIG, 330
- TsGpioPinSetting
 - PCH_MEMORY_THROTTLING, 275
- TSN_CONFIG, 368
- TsnConfig.h, 513
- TTLevels
 - THERMAL_CONFIG, 363
- TTLock
 - THERMAL_THROTTLE_LEVELS, 365
- TTState13Enable
 - THERMAL_THROTTLE_LEVELS, 365
- TurboRatioLimitNumCore
 - CPU_POWER_MGMT_BASIC_CONFIG, 122
- TurboRatioLimitRatio
 - CPU_POWER_MGMT_BASIC_CONFIG, 122
- TvbDownBinsTempThreshold0
 - OVERCLOCKING_PREMEM_CONFIG, 254
- TvbDownBinsTempThreshold1
 - OVERCLOCKING_PREMEM_CONFIG, 254
- TvbRatioClipping
 - OVERCLOCKING_PREMEM_CONFIG, 254
- TvbTempThreshold0
 - OVERCLOCKING_PREMEM_CONFIG, 254
- TvbTempThreshold1
 - OVERCLOCKING_PREMEM_CONFIG, 255
- TvbVoltageOptimization
 - OVERCLOCKING_PREMEM_CONFIG, 255
- TwoCoreRatioLimit
 - CPU_POWER_MGMT_BASIC_CONFIG, 123
- Txt
 - CPU_SECURITY_PREMEM_CONFIG, 140
- TxtEnable
 - CPU_CONFIG, 88
- UaolEnable
 - USB_CONFIG, 377
- UART_PIN_MUX, 369
- UefiBaseType.h, 514
 - EFI_IPv4_ADDRESS, 517
 - EFI_IPv6_ADDRESS, 517
 - EFI_PAGES_TO_SIZE, 516
 - EFI_SIZE_TO_PAGES, 517
- UnlockGpioPads
 - PCH_LOCK_DOWN_CONFIG, 274
- UNREACHABLE
 - Base.h, 403
- USB2_PHY_CONFIG, 370
 - Port, 371
- USB2_PHY_PARAMETERS, 371
- USB2_PORT_CONFIG, 372
 - OverCurrentPin, 372
- Usb2PhyConfig.h, 518
- Usb2PhySusPgEnable
 - PCH_PM_CONFIG, 289
- USB3_HSIO_CONFIG, 373
- USB3_PORT_CONFIG, 374
 - OverCurrentPin, 374
- Usb3HsioConfig.h, 519
- USB_CONFIG, 375
 - LtrOverrideEnable, 376
 - OverCurrentEnable, 376
 - PdoProgramming, 376
 - UaolEnable, 377
 - XhciOcLock, 377
- UsbConfig.h, 520

- VA_ARG
 - Base.h, [403](#)
- VA_COPY
 - Base.h, [405](#)
- VA_END
 - Base.h, [405](#)
- VA_LIST
 - Base.h, [406](#)
- VA_START
 - Base.h, [405](#)
- VccInDemotionCapacitanceInUf
 - CPU_POWER_MGMT_VR_CONFIG, [136](#)
- VccInDemotionEnable
 - CPU_POWER_MGMT_VR_CONFIG, [136](#)
- VccInDemotionQuiescentPowerInMw
 - CPU_POWER_MGMT_VR_CONFIG, [137](#)
- VerbTableEntryNum
 - HDAUDIO_CONFIG, [194](#)
- VerbTablePtr
 - HDAUDIO_CONFIG, [194](#)
- Version
 - EFI_HOB_HANDOFF_INFO_TABLE, [155](#)
- VMD_PEI_CONFIG, [378](#)
 - VmdCfgBarSize, [379](#)
- VmdCfgBarSize
 - VMD_PEI_CONFIG, [379](#)
- VmdPeiConfig.h, [521](#)
- VmxEnable
 - CPU_CONFIG_LIB_PREMEM_CONFIG, [96](#)
- Voltage
 - FIVR_EXT_RAIL_CONFIG, [173](#)
- VppPort
 - PCH_PCIE_ROOT_PORT_CONFIG, [280](#)
- VrPowerDeliveryDesign
 - CPU_POWER_MGMT_VR_CONFIG, [137](#)
- VsysAssertionDeglitchMantissa
 - CPU_POWER_MGMT_VR_CONFIG, [137](#)
- VsysFullScale
 - CPU_POWER_MGMT_VR_CONFIG, [137](#)
- VTD_CONFIG, [380](#)
 - EnableDmaBuffer, [381](#)
 - PreBootDmaMask, [381](#)
 - VtdDisable, [381](#)
- VTD_DXE_CONFIG, [382](#)
- VTD_ENABLE_DMA_BUFFER
 - VtdConfig.h, [523](#)
- VtdConfig.h, [522](#)
 - VTD_ENABLE_DMA_BUFFER, [523](#)
- VtdDisable
 - VTD_CONFIG, [381](#)
- WatchDogConfig.h, [524](#)
- WatchDogEnabled
 - AMT_PEI_CONFIG, [79](#)
- WatchDogTimerBios
 - AMT_PEI_CONFIG, [79](#)
- WatchDogTimerOs
 - AMT_PEI_CONFIG, [79](#)
- WRDSUDT
 - MEMORY_CONFIGURATION, [238](#)
- X2ApicSupport
 - CPU_CONFIG, [88](#)
- XDCI_CONFIG, [383](#)
 - Enable, [383](#)
- XhciOcLock
 - USB_CONFIG, [377](#)
- XMP_PROFILE_1
 - MemoryConfig.h, [459](#)
- XMP_PROFILE_2
 - MemoryConfig.h, [459](#)
- XMP_PROFILE_3
 - MemoryConfig.h, [459](#)
- XMP_USER_PROFILE_4
 - MemoryConfig.h, [459](#)
- XMP_USER_PROFILE_5
 - MemoryConfig.h, [459](#)
- ZpOdd
 - PCH_SATA_PORT_CONFIG, [291](#)