Apollolake Intel(R) Firmware Support Package (FSP) Integration Guide

Tue Jun 23 2020 15:35:43

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.

Contents

Chapter 1

INTRODUCTION

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 ApolloLake platforms with Broxton-P processor.

1.2 Intended Audience

This document is targeted to 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 like EDKII or Coreboot, system integrators, as well as end users.

1.3 Related Documents

- *Platform Initialization (PI) Specification v1.4* http://www.uefi.org/specifications
- *UEFI Specification v2.5* http://www.uefi.org/specifications
- *Intel® Firmware Support Package: External Architecture Specification (EAS) v2.0* http://www.↩ intel.com/content/dam/www/public/us/en/documents/technical-specifications/fsp-archit pdf
- *Boot Setting File Specification (BSF) v1.0* https://firmware.intel.com/sites/default/files/↩ BSF_1_0.pdf
- Binary Configuration Tool for Intel® FSP http://www.intel.com/fsp

1.4 Acronyms and Terminology

Acronym	Definition
BCT	Binary Configuration Tool
BSF	Boot Setting File
BSP	Boot Strap Processor
BWG	BIOS Writer's Guide
CAR	Cache As Ram

CRB	Customer Reference Board
eMMC	embedded Multi-Media Controller
FIT	Firmware Interface Table
FSP	Firmware Support Package
FSP API	Firmware Support Package Interface
FW	Firmware
IBB	Initial Boot Block
IBBL	Initial Boot Block Loader
OBB	Oem BIOS Block
PCH	Platform Controller Hub
PMC	Power Management Controller
SBSP	System BSP
SMI	System Management Interrupt
SMM	System Management Mode
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
TSEG	Memory Reserved at the Top of Memory to be used as SMRAM
UPD	Updatable Product Data

Chapter 2

FSP OVERVIEW

FSP 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 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 external tools that will be provided by Intel.
- **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 File
- 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)**
  - Apollo_Lake_FSP_Integration_Guide.pdf
  - Apollo_Lake_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)
- Fsp.bsf (BSF file for configuring the data using BCT tool)
- Fsp.fd (FSP Binary)

Chapter 3

FSP INTEGRATION

3 FSP Integration

This Revision of the FSP is based on FSP EAS v2.0

3.1 Boot Flow

Please refer to FSP EAS 2.0 section 7 for more details on the FSP2.0 boot flow.

3.2 FSP Component Extraction

Apollo Lake FSP image can be split into 3 different components (FSP-T, FSP-M and FSP-S) and each component can be located at different base addresses according to its execution location.

In Apollo Lake boot flow there are 3 different execution stages:

- execution in SRAM
- execution in temporary memory (cache as ram)
- execution in system memory

The 3 extracted FSP components can be exactly mapped into different execution stages on Apollo Lake boot flow.

- FSP-T will be executing in SRAM
- FSP-M will be executing in temporary memory. After the memory is initialized the generic code like PEI dispatcher and other FSP data will be migrated into permanent memory
- FSP-S will be executing in memory

By default the FSP-T component default base address is set to 0xFFFF8000, FSP-M component default base address is set to 0xFEF71000, and the FSP-S component default base address is set to 0x0200000. If the FSP component needs to be loaded at different address, please use the BCT tool to rebase it first before the integration. Specially, to rebase the FSP-S component, it can be done easily by changing the FSP_INFO_HEADER.Image↩ Base** to the desired location and no other steps are required. Please note for FSP-T and FSP-M components, the normal rebasing process has to be done properly.

FSP Binary will be released as a single FD. You can use the SplitFspBin.py to split the FD in to the different FSP components. SplitFspBin.py is available at https://github.com/tianocore/edk2/tree/master/↩ IntelFsp2Pkg/Tools"

3.3 FSP Information Header

The FSP has an **FSP_INFO_HEADER** structure embedded in each FSP component. It provides critical information that is required by the boot loader to successfully interface with the FSP. The structure of the FSP Information Header is documented in the FSP EAS v2.0.

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.

The FSP API parameters documented in this integration guide are applicable for the Image ID and Revision specified as below.

The current FSP ImageId string in the FSP information header is ∗∗$APLFSP$∗∗ and the ImageRevision field is 0x01050100(1.5.2.0).

3.5 FSP APIs

This release of the Apollo Lake FSP supports all APIs required by the FSP EAS v2.0. The FSP information header contains the address offset for these APIs. Register usage and calling convention are described in the FSP EAS v2.0. Any usage not described by the specification is described in the individual sections below.

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

3.5.1 TempRamInit API

Please refer Chapter 8.5 in the FSP EAS v2.0 for complete details including the prototype, parameters and return value details for this API.

If Boot Loader initializes the Temporary RAM (CAR), it can skip calling this API.

FsptUpdPtr is pointer to **FSPT_UPD** structure which is described in header file FsptUpd.h

TempRamInit∗∗ does basic early initialization primarily setting up temporary RAM using cache. It returns a temporary memory data region that can be used by the boot loader with ECX pointing to beginning of temporary memory and EDX pointing to end of temporary memory. The temporary memory data region returned by this FSP release is from 0xFEF00000 to 0xFEFFFC00

On Apollo Lake SOC the microcode will be loaded automatically by the processor before it starts reset vector execution. As a result it is not required to pass in a microcode region in this API, and parameter.

Both **FSPT_UPD.MicrocodeRegionBase** and **FSPT_UPD.MicrocodeRegionLength** can be set to 0. However, if a valid region is passed and a newer microcode update revision is in this region, it will be loaded by the FSP.

On Apollo Lake SoC the top 32KB SRAM region will be used to load and execution IBBL, including boot loader IBBL and FSP-T component. Since the top 128KB SRAM will also be used as a ring buffer to load IBB as, this region is recommended to be set to uncacheable before the completion of the system memory initialization. It is recommended to set parameter **FSPT_UPD.CodeRegionBase** to 0xFFFE0000 and **FSPT_UPD.CodeRegion↩ Length** to 0 to disable the code region caching in FSP. However, it does not exclude any special usage model that enables part of the top 128K SRAM as cacheable at the beginning of the ring buffer protocol, and then disables the caching at the later stage of the ring buffer IBB loading process.

3.5.2 FspMemoryInit API

Please refer to Chapter 8.6 in the FSP external Architecture Specification version 2.0 for the prototype, parameters and return value details for this API.

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

Boot Loader must pass valid CAR region for FSP stack use through **FSPM_UPD.FspmArchUpd.StackBase** and **FSPM_UPD.FspmArchUpd.StackSize** UPDs.

The minimum FSP stack size required for this revision of FSP is 168KB, stack base is 0xFEF22000 by default.

Note

Certain platforms might need some GPIOs to be initialized prior to the memory initialization. In this case the boot loader needs to configure the required GPIO pins properly before calling into **FspMemoryInit**. For example to read SPD data, the SMBUS pins have to be configured properly.

3.5.3 TempRamExit API

Please refer to Chapter 8.7 in the FSP EAS v2.0 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.

This revision of FSP doesn't have any fields/structure to pass as parameter for this API. Pass Null for *TempRam↩ ExitParamPtr*.

At the end of *TempRamExit* the original code and data caching are disabled. FSP will reconfigure all MTRRs as described in the table below for performance optimization.

Memory range	Cache Attribute
0x00000000 – 0x0009FFFF	Write back
0x000C0000 – Top of Low Memory	Write back
0xFF800000 – 0xFFFFFFFF (Flash region)	Write protect
0x1000000000 – Top of High Memory	Write back

If the boot loader wish to reconfigure the MTRRs differently, it can be overridden immediately after this API call.

3.5.4 FspSiliconInit API

Please refer to Chapter 8.8 in the FSP external Architecture Specification version 2.0 for the prototype, parameters and return value details for this API.

The *FspsUpdPtr* is pointer to **FSPS_UPD** structure which is described in header file FspsUpd.h.

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

3.5.5 NotifyPhase API

Please refer Chapter 8.9 in the FSP EAS 2.0 for the prototype, parameters and return value details for this API.

3.5.5.1 PostPciBusEnumeration Notification

This phase *EnumInitPhaseAfterPciEnumeration* is to be called after PCI bus enumeration but before execution of third party code such as option ROMs. Currently, no special operation is done in this phase, but in the future updates, programming may be added in this phase.

3.5.5.2 ReadyToBoot Notification

This phase *EnumInitPhaseReadyToBoot* is to be called before giving control to OS Loader. It includes some final initialization steps recommended by the BWG, including power management settings, security related registers locking down, switching devices into ACPI mode if required, etc.

3.5.5.3 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.

3.6 Memory Map

3.6.1 System Memory Map

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

3.7 Porting recommendation

Here listed some notes or recommendation when porting with FSP.

3.7.1 FSP_STATUS_RESET_REQUIRED

As per FSP External Architecture Specification version 2.0, Any reset required in the FSP flow will be reported as return status FSP_STATUS_RESET_REQUIREDx by the API.It is the bootloader responsibility to reset the system according to the reset type requested. Note:
-If Bootloader ignores the reset request and calls the next FSP API instead of triggering the reset, FSP will trigger the required reset.

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 - not used in the current version of FSP
0x40000003	Shutdown Reset - not used in the current version of FSP
0x40000004	not used
0x40000005	Global Reset - Puts the system to Global reset through Heci or Full Reset through PCH

Chapter 4

FSP OUTPUT

4 FSP Output

The FSP builds a series of data structures called the 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 and to Chapter 9 in the FSP EAS v2.0 for details about FSP Architectural HOBs. Below section describe the HOBs not covered in the above two specifications.

4.1 SMRAM Resource Descriptor HOB

The FSP will report the system SMRAM T-SEG range through a generic resource HOB. This HOB follows the **EFI_HOB_RESOURCE_DESCRIPTOR** format with the owner GUID defined as below:

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

4.1 FSP_VARIABLE_NV_DATA_HOB

The FSP_VARIABLE_NV_DATA_HOB provides a mechanism for FSP to request the bootloader to save the platform configuration data into non-volatile storage so that it can be reused in special cases, such as S3 resume or fast boot.

```
#define FSP_VARIABLE_NV_DATA_HOB_GUID  \
{ 0xa034147d, 0x690c, 0x4154, { 0x8d, 0xe6, 0xc0, 0x44, 0x64, 0x1d, 0xe9, 0x42 } }
```

The bootloader needs to parse the HOB list to see if such a GUID HOB exists after returning from the FspMemory↩ Init() API. If it exists, the bootloader should extract the data portion from the HOB structure and then save it into a platform-specific NVS device, such as flash, EEPROM, etc. On the following boot flow the bootloader should load the data block back from the NVS device to temporary memory and populate the buffer pointer into FSPM_UP↩ D.VariableNvsBufferPtr field before calling into the FspMemoryInit() API. If the NVS device is memory mapped, the bootloader can initialize the buffer pointer directly to the buffer.

This HOB must be parsed after FspMemoryInit() API.

This HOB is produced only when new NVS data is generated. For example, if this HOB is not produced in S3 or fast boot, Bootloader should continue to pass the existing NVS data to FSP during next boot.

Chapter 5

FSP POSTCODE

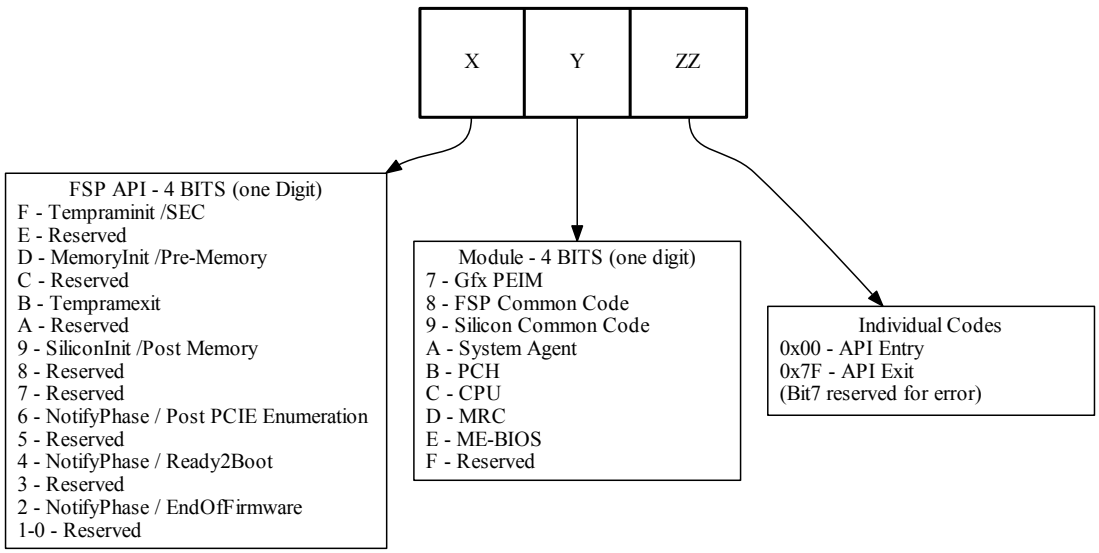
5 FSP StatusCode

The FSP outputs 16 bit postcode to indicate which API and in which module the execution is happening.

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

5.1 Status Code Info

Below diagram represents the 16 bit PostCode usage in FSP.



5.1.1 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)
0xF07F	FSP	TempRamInit API Exit

5.1.2 FspMemoryInit API Status Codes (0xDxxx)

PostCode	Module	Description
0xD800	FSP	FspMemoryInit API Entry
0xD87F	FSP	FSpMemoryInit API Exit
0xDA00	SA	SaInitPreMemEntry
0xDA01	SA	DeviceConfigurePreMem
0xDA02	SA	OverrideDev0Did
0xDA04	SA	OverrideDev2Did
0xDA06	SA	Programming SA Bars
0xDA08	SA	Install SA HOBs
0xDA0A	SA	Reporting SA PCIe code version
0xDA0C	SA	SaSvInit
0xDA10	SA	Initializing DMI
0xDA1F	SA	Initializing Max PayLoad Size
0xDA20	SA	Initializing SwitchableGraphics
0xDA30	SA	Initializing SA PCIe
0xDA3F	SA	FlowControlCreditProgramming↩ UltUlx
0xDA40	SA	Initializing DMI Tc/Vc mapping
0xDA42	SA	CheckOffboardPcieVga
0xDA44	SA	CheckAndInitializePegVga
0xDA50	SA	GraphicsPreMemInit
0xDA7F	SA	Pre-Mem SaInit Exit
0xDB00	PCH	Pre-Mem ScInit Entry
0xDB02	PCH	Pre-Mem Early configuration
0xDB10	PCH	Pre-Mem PCIe Power Sequence configuration
0xDC00	CPU	Pre-Mem Entry
0xDC7F	CPU	Pre-Mem Exit

5.1.3 TempRamExit API Status Codes (0xBxxx)

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

5.1.3 FspSiliconInit API Status Codes (0x9xxx)

PostCode	Module	Description
0x9800	FSP	FspSiliconInit API Entry
0x987F	FSP	FspSiliconInit API Exit
0x9A00	SA	Post-Mem SaInit Entry
0x9A01	SA	DeviceConfigure
0x9A02	SA	InstallSaHob

0x9A03	SA	PeiDisplayInit
0x9A04	SA	PeiGraphicsNotifyCallback Entry
0x9A05	SA	CallPpiAndFillFrameBuffer
0x9A06	SA	GraphicsPpiInit
0x9A07	SA	GraphicsPpiGetMode
0x9A08	SA	FillFrameBufferAndShowLogo
0x9A09	SA	PeiGraphicsNotifyCallback Exit
0x9A0A	SA	ProgramEcBase
0x9A0B	SA	SaAunitInit
0x9A0C	SA	HybridGraphicsInit
0x9A10	SA	SaOcInit
0x9A14	SA	IpuInit
0x9A16	SA	Initializing SA GMM device
0x9A18	SA	SaProgramSvidSid
0x9A1A	SA	SaProgramLlcWays
0x9A20	SA	Initializing PciExpressInitPostMem
0x9A30	SA	Initializing Vtd
0x9A32	SA	Initializing Pavp
0x9A34	SA	PeiInstallSmmAccessPpi
0x9A36	SA	EdramWa
0x9A4F	SA	Post-Mem SaInit Exit
0x9A50	SA	SaSecurityLock Entry
0x9A5F	SA	SaSecurityLock Exit
0x9A60	SA	SaSResetComplete Entry
0x9A61	SA	RESET_CPL
0x9A62	SA	SaSvInit2
0x9A63	SA	GraphicsPmInit
0x9A64	SA	SaPeiPolicyDump
0x9A6F	SA	SaSResetComplete Exit
0x9A70	SA	SaS3ResumeAtEndOfPei Callback Entry
0x9A7F	SA	SaS3ResumeAtEndOfPei Callback Exit
0x9B7F	PCH	Post-Mem ScInit Entry
0x9B01	PCH	Post-Mem Program HSIO ModPHY settings
0x9B02	PCH	Post-Mem SMBus configuration
0x9B03	PCH	Post-Mem LPC configuration
0x9B04	PCH	Post-Mem SATA initialization
0x9B05	PCH	Post-Mem PCIe initialization
0x9B06	PCH	Post-Mem xHCI initialization
0x9B07	PCH	Post-Mem xDCI initialization
0x9B08	PCH	Post-Mem HD Audio initialization
0x9B09	PCH	Post-Mem GMM configuration
0x9B0A	PCH	Post-Mem LPSS initialization
0x9B0B	PCH	Post-Mem SCS initialization
0x9B0C	PCH	Post-Mem ISH initialization
0x9B0D	PCH	Post-Mem ITSS configuration

0x9B40	PCH	Post-Mem OnEndOfPEI Entry
0x9B4F	PCH	Post-Mem OnEndOfPEI Exit
0x9B7F	PCH	Post-Mem ScInit Exit
0x9C00	CPU	Post-Mem Entry
0x9C7F	CPU	Post-Mem Exit

5.1.4 NotifyPhase API Status Codes (0x6xxx)

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

Chapter 6

Class Documentation

6.1 FSP_INFO_EXTENDED_HEADER Struct Reference

FSP Information Extended Header as described in FSP v2.0 Spec section 5.1.2.

`#include <FspHeaderFile.h>`

Public Attributes

- UINT32 Signature

  Byte 0x00: Signature ('FSPE') for the FSP Extended Information Header.
- UINT32 Length

  Byte 0x04: Length of the table in bytes, including all additional FSP producer defined data.
- UINT8 Revision

  Byte 0x08: FSP producer defined revision of the table.
- UINT8 Reserved

  Byte 0x09: Reserved for future use.
- CHAR8 FspProducerId [6]

  Byte 0x0A: FSP producer identification string.
- UINT32 FspProducerRevision

  Byte 0x10: FSP producer implementation revision number.
- UINT32 FspProducerDataSize

  Byte 0x14: Size of the FSP producer defined data (n) in bytes.

6.1.1 Detailed Description

FSP Information Extended Header as described in FSP v2.0 Spec section 5.1.2.

Definition at line 129 of file FspHeaderFile.h.

6.1.2 Member Data Documentation

6.1.2.1 UINT32 FSP_INFO_EXTENDED_HEADER::FspProducerRevision

Byte 0x10: FSP producer implementation revision number.

Larger numbers are assumed to be newer revisions.

Definition at line 153 of file FspHeaderFile.h.

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

- FspHeaderFile.h

6.2 FSP_INFO_HEADER Struct Reference

FSP Information Header as described in FSP v2.0 Spec section 5.1.1.

`#include <FspHeaderFile.h>`

Public Attributes

- UINT32 Signature

  Byte 0x00: Signature ('FSPH') for the FSP Information Header.
- UINT32 HeaderLength

  Byte 0x04: Length of the FSP Information Header.
- UINT8 Reserved1 [2]

  Byte 0x08: Reserved.
- UINT8 SpecVersion

  Byte 0x0A: Indicates compliance with a revision of this specification in the BCD format.
- UINT8 HeaderRevision

  Byte 0x0B: Revision of the FSP Information Header.
- UINT32 ImageRevision

  Byte 0x0C: Revision of the FSP binary.
- CHAR8 ImageId [8]

  Byte 0x10: Signature string that will help match the FSP Binary to a supported HW configuration.
- UINT32 ImageSize

  Byte 0x18: Size of the entire FSP binary.
- UINT32 ImageBase

  Byte 0x1C: FSP binary preferred base address.
- UINT16 ImageAttribute

  Byte 0x20: Attribute for the FSP binary.
- UINT16 ComponentAttribute

  Byte 0x22: Attributes of the FSP Component.
- UINT32 CfgRegionOffset

  Byte 0x24: Offset of the FSP configuration region.
- UINT32 CfgRegionSize

  Byte 0x28: Size of the FSP configuration region.
- UINT32 Reserved2

  Byte 0x2C: Reserved2.
- UINT32 TempRamInitEntryOffset

  Byte 0x30: The offset for the API to setup a temporary stack till the memory is initialized.
- UINT32 Reserved3

  Byte 0x34: Reserved3.
- UINT32 NotifyPhaseEntryOffset

  Byte 0x38: The offset for the API to inform the FSP about the different stages in the boot process.
- UINT32 FspMemoryInitEntryOffset

  Byte 0x3C: The offset for the API to initialize the memory.
- UINT32 TempRamExitEntryOffset

  Byte 0x40: The offset for the API to tear down temporary RAM.
- UINT32 FspSiliconInitEntryOffset

  Byte 0x44: The offset for the API to initialize the CPU and chipset.

6.2.1 Detailed Description

FSP Information Header as described in FSP v2.0 Spec section 5.1.1.

Definition at line 38 of file FspHeaderFile.h.

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

- FspHeaderFile.h

6.3 FSP_M_CONFIG Struct Reference

Fsp M Configuration.

`#include <FspmUpd.h>`

Public Attributes

- UINT32 SerialDebugPortAddress

  Offset 0x0040 - Debug Serial Port Base address Debug serial port base address.
- UINT8 SerialDebugPortType

  Offset 0x0044 - Debug Serial Port Type 16550 compatible debug serial port resource type.
- UINT8 SerialDebugPortDevice

  Offset 0x0045 - Serial Port Debug Device Select active serial port device for debug.
- UINT8 SerialDebugPortStrideSize

  Offset 0x0046 - Debug Serial Port Stride Size Debug serial port register map stride size in bytes.
- UINT8 MrcFastBoot

  Offset 0x0047 - Memory Fast Boot Enable/Disable MRC fast boot support.
- UINT8 Igd

  Offset 0x0048 - Integrated Graphics Device Enable : Enable Integrated Graphics Device (IGD) when selected as the Primary Video Adaptor.
- UINT8 IgdDvmt50PreAlloc

  Offset 0x0049 - DVMT Pre-Allocated Select DVMT 5.0 Pre-Allocated (Fixed) Graphics Memory size used by the Internal Graphics Device.
- UINT8 IgdApertureSize

  Offset 0x004A - Aperture Size Select the Aperture Size used by the Internal Graphics Device.
- UINT8 GttSize

  Offset 0x004B - GTT Size Select the GTT Size used by the Internal Graphics Device.
- UINT8 PrimaryVideoAdaptor

  Offset 0x004C - Primary Display Select which of IGD/PCI Graphics device should be Primary Display.
- UINT8 Package

  Offset 0x004D - Package NOTE: Specifies CA Mapping for all technologies.
- UINT8 Profile

  Offset 0x004E - Profile Profile list.
- UINT8 MemoryDown

  Offset 0x004F - MemoryDown Memory Down.
- UINT8 DDR3LPageSize

  Offset 0x0050 - DDR3LPageSize NOTE: Only for memory down (soldered down memory with no SPD).
- UINT8 DDR3LASR

  Offset 0x0051 - DDR3LASR NOTE: Only for memory down.
- UINT8 ScramblerSupport

  Offset 0x0052 - ScramblerSupport Scrambler Support - Enable or disable the memory scrambler.

- UINT8 InterleavedMode

  Offset 0x0053 - InterleavedMode This field is ignored if one of the PnP channel configurations is used.
- UINT16 ChannelHashMask

  Offset 0x0054 - ChannelHashMask ChannelHashMask and SliceHashMask allow for the channel hashing algorithm to be modified.
- UINT16 SliceHashMask

  Offset 0x0056 - SliceHashMask ChannelHashMask and SliceHashMask allow for the channel hashing algorithm to be modified.
- UINT8 ChannelsSlicesEnable

  Offset 0x0058 - ChannelsSlicesEnable ChannelSlicesEnable field is not used at all on BXTP.
- UINT8 MinRefRate2xEnable

  Offset 0x0059 - MinRefRate2xEnable Provided as a means to defend against Row-Hammer attacks.
- UINT8 DualRankSupportEnable

  Offset 0x005A - DualRankSupportEnable Dual Rank Support Enable.
- UINT8 RmtMode

  Offset 0x005B - RmtMode Rank Margin Tool Mode.
- UINT16 MemorySizeLimit

  Offset 0x005C - MemorySizeLimit Memory Size Limit: This value is used to restrict the total amount of memory and the calculations based on it.
- UINT16 LowMemoryMaxValue

  Offset 0x005E - LowMemoryMaxValue Low Memory Max Value: This value is used to restrict the amount of memory below 4GB and the calculations based on it.
- UINT16 HighMemoryMaxValue

  Offset 0x0060 - HighMemoryMaxValue High Memory Max Value: This value is used to restrict the amount of memory above 4GB and the calculations based on it.
- UINT8 DisableFastBoot

  Offset 0x0062 - DisableFastBoot 00:Disabled; Use saved training data (if valid) after first boot(Default), 01:Enabled; Full re-train of memory on every boot.
- UINT8 DIMM0SPDAddress

  Offset 0x0063 - DIMM0SPDAddress DIMM0 SPD Address (NOTE: Only for DDR3L only.
- UINT8 DIMM1SPDAddress

  Offset 0x0064 - DIMM1SPDAddress DIMM1 SPD Address (NOTE: Only for DDR3L only.
- UINT8 Ch0_RankEnable

  Offset 0x0065 - Ch0_RankEnable NOTE: Only for memory down.
- UINT8 Ch0_DeviceWidth

  Offset 0x0066 - Ch0_DeviceWidth NOTE: Only for memory down.
- UINT8 Ch0_DramDensity

  Offset 0x0067 - Ch0_DramDensity NOTE: Only for memory down.
- UINT8 Ch0_Option

  Offset 0x0068 - Ch0_Option BIT[0] Rank Select Interleaving Enable.
- UINT8 Ch0_OdtConfig

  Offset 0x0069 - Ch0_OdtConfig [0] RX ODT - DDR3L & LPDDR3 only: Change the READ ODT strength , for SOC termination during a READ transaction, ON DQ BITs.
- UINT8 Ch0_TristateClk1

  Offset 0x006A - Ch0_TristateClk1 Not used.
- UINT8 Ch0_Mode2N

  Offset 0x006B - Ch0_Mode2N DDR3L Only: Configures the DDR3L command timing mode.
- UINT8 Ch0_OdtLevels

  Offset 0x006C - Ch0_OdtLevels Parameter used to determine if ODT will be held high or low: 0 - ODT Connected to SoC, 1 - ODT held high.
- UINT8 Ch1_RankEnable

  Offset 0x006D - Ch1_RankEnable NOTE: Only for memory down.

- UINT8 Ch1_DeviceWidth

  Offset 0x006E - Ch1_DeviceWidth NOTE: Only for memory down.
- UINT8 Ch1_DramDensity

  Offset 0x006F - Ch1_DramDensity NOTE: Only for memory down.
- UINT8 Ch1_Option

  Offset 0x0070 - Ch1_Option BIT[0] Rank Select Interleaving Enable.
- UINT8 Ch1_OdtConfig

  Offset 0x0071 - Ch1_OdtConfig [0] RX ODT - DDR3L & LPDDR3 only: Change the READ ODT strength , for SOC termination during a READ transaction, ON DQ BITs.
- UINT8 Ch1_TristateClk1

  Offset 0x0072 - Ch1_TristateClk1 Not used.
- UINT8 Ch1_Mode2N

  Offset 0x0073 - Ch1_Mode2N DDR3L Only: Configures the DDR3L command timing mode.
- UINT8 Ch1_OdtLevels

  Offset 0x0074 - Ch1_OdtLevels DDR3L Only: Parameter used to determine if ODT will be held high or low: 0 - ODT_AB_HIGH_LOW (default), 1 - ODT_AB_HIGH_HIGH.
- UINT8 Ch2_RankEnable

  Offset 0x0075 - Ch2_RankEnable NOTE: Only for memory down.
- UINT8 Ch2_DeviceWidth

  Offset 0x0076 - Ch2_DeviceWidth NOTE: Only for memory down.
- UINT8 Ch2_DramDensity

  Offset 0x0077 - Ch2_DramDensity NOTE: Only for memory down.
- UINT8 Ch2_Option

  Offset 0x0078 - Ch2_Option BIT[0] Rank Select Interleaving Enable.
- UINT8 Ch2_OdtConfig

  Offset 0x0079 - Ch2_OdtConfig [0] RX ODT - DDR3L & LPDDR3 only: Change the READ ODT strength , for SOC termination during a READ transaction, ON DQ BITs.
- UINT8 Ch2_TristateClk1

  Offset 0x007A - Ch2_TristateClk1 Not used.
- UINT8 Ch2_Mode2N

  Offset 0x007B - Ch2_Mode2N DDR3L Only: Configures the DDR3L command timing mode.
- UINT8 Ch2_OdtLevels

  Offset 0x007C - Ch2_OdtLevels DDR3L Only: Parameter used to determine if ODT will be held high or low: 0 - ODT_AB_HIGH_LOW (default), 1 - ODT_AB_HIGH_HIGH.
- UINT8 Ch3_RankEnable

  Offset 0x007D - Ch3_RankEnable NOTE: Only for memory down.
- UINT8 Ch3_DeviceWidth

  Offset 0x007E - Ch3_DeviceWidth NOTE: Only for memory down.
- UINT8 Ch3_DramDensity

  Offset 0x007F - Ch3_DramDensity NOTE: Only for memory down.
- UINT8 Ch3_Option

  Offset 0x0080 - Ch3_Option BIT[0] Rank Select Interleaving Enable.
- UINT8 Ch3_OdtConfig

  Offset 0x0081 - Ch3_OdtConfig [0] RX ODT - DDR3L & LPDDR3 only: Change the READ ODT strength , for SOC termination during a READ transaction, ON DQ BITs.
- UINT8 Ch3_TristateClk1

  Offset 0x0082 - Ch3_TristateClk1 Not used.
- UINT8 Ch3_Mode2N

  Offset 0x0083 - Ch3_Mode2N DDR3L Only: Configures the DDR3L command timing mode.
- UINT8 Ch3_OdtLevels

  Offset 0x0084 - Ch3_OdtLevels DDR3L Only: Parameter used to determine if ODT will be held high or low: 0 - ODT_AB_HIGH_LOW (default), 1 - ODT_AB_HIGH_HIGH.

- UINT8 RmtCheckRun

  Offset 0x0085 - RmtCheckRun Parameter used to determine whether to run the margin check.
- UINT16 RmtMarginCheckScaleHighThreshold

  Offset 0x0086 - RmtMarginCheckScaleHighThreshold Percentage used to determine the margin tolerances over the failing margin.
- UINT8 Ch0_Bit_swizzling [32]

  Offset 0x0088 - Ch0_Bit_swizzling Channel 0 PHY to DUnit DQ mapping (only used if not 1-1 mapping)Range: 0-32.
- UINT8 Ch1_Bit_swizzling [32]

  Offset 0x00A8 - Ch1_Bit_swizzling Channel 1 PHY to DUnit DQ mapping (only used if not 1-1 mapping)Range: 0-32.
- UINT8 Ch2_Bit_swizzling [32]

  Offset 0x00C8 - Ch2_Bit_swizzling Channel 2 PHY to DUnit DQ mapping (only used if not 1-1 mapping)Range: 0-32.
- UINT8 Ch3_Bit_swizzling [32]

  Offset 0x00E8 - Ch3_Bit_swizzling Channel 3 PHY to DUnit DQ mapping (only used if not 1-1 mapping)Range: 0-32.
- UINT32 MsgLevelMask

  Offset 0x0108 - MsgLevelMask 32 bits used to mask out debug messages.
- UINT8 UnusedUpdSpace0 [4]

  Offset 0x010C.
- UINT8 PreMemGpioTablePinNum [4]

  Offset 0x0110 - PreMem GPIO Pin Number for each table Number of Pins in each PreMem GPIO Table.
- UINT32 PreMemGpioTablePtr

  Offset 0x0114 - PreMem GPIO Table Pointer Pointer to Array of pointers to PreMem GPIO Table.
- UINT8 PreMemGpioTableEntryNum

  Offset 0x0118 - PreMem GPIO Table Entry Number.
- UINT8 EnhancePort8xhDecoding

  Offset 0x0119 - Enhance the port 8xh decoding Enable/Disable Enhance the port 8xh decoding.
- UINT8 SpdWriteEnable

  Offset 0x011A - SPD Data Write Enable/Disable SPD data write on the SMBUS.
- UINT8 MrcDataSaving

  Offset 0x011B - MRC Training Data Saving Enable/Disable MRC training data saving in FSP.
- UINT32 OemLoadingBase

  Offset 0x011C - OEM File Loading Address Determine the memory base address to load a specified file from CSE file system after memory is available.
- UINT8 OemFileName [16]

  Offset 0x0120 - OEM File Name to Load Specify a file name to load from CSE file system after memory is available.
- VOID * MrcBootDataPtr

  Offset 0x0130.
- UINT8 eMMCTraceLen

  Offset 0x0134 - eMMC Trace Length Select eMMC trace length to load OEM file from when loading OEM file name is specified.
- UINT8 SkipCseRbp

  Offset 0x0135 - Skip CSE RBP to support zero sized IBB Enable/Disable skip CSE RBP for bootloader which loads IBB without assistance of CSE.
- UINT8 NpkEn

  Offset 0x0136 - Npk Enable Enable/Disable Npk.
- UINT8 FwTraceEn

  Offset 0x0137 - FW Trace Enable Enable/Disable FW Trace.
- UINT8 FwTraceDestination

  Offset 0x0138 - FW Trace Destination FW Trace Destination.
- UINT8 RecoverDump

  Offset 0x0139 - NPK Recovery Dump Enable/Disable NPK Recovery Dump.
- UINT8 Msc0Wrap

Offset 0x013A - Memory Region 0 Buffer WrapAround Memory Region 0 Buffer WrapAround.

- UINT8 Msc1Wrap

  Offset 0x013B - Memory Region 1 Buffer WrapAround Memory Region 1 Buffer WrapAround.

- UINT32 Msc0Size

  Offset 0x013C - Memory Region 0 Buffer Size Memory Region 0 Buffer Size.

- UINT32 Msc1Size

  Offset 0x0140 - Memory Region 1 Buffer Size Memory Region 1 Buffer Size, 0-0MB(Default), 1-1MB, 2-8MB, 3-64MB, 4-128MB, 5-256MB, 6-512MB, 7-1GB.

- UINT8 PtiMode

  Offset 0x0144 - PTI Mode PTI Mode.

- UINT8 PtiTraining

  Offset 0x0145 - PTI Training PTI Training.

- UINT8 PtiSpeed

  Offset 0x0146 - PTI Speed PTI Speed.

- UINT8 PunitMlvl

  Offset 0x0147 - Punit Message Level Punit Message Output Verbosity Level.

- UINT8 PmcMlvl

  Offset 0x0148 - PMC Message Level PMC Message Output Verbosity Level.

- UINT8 SwTraceEn

  Offset 0x0149 - SW Trace Enable Enable/Disable SW Trace.

- UINT8 PeriodicRetrainingDisable

  Offset 0x014A - Periodic Retraining Disable Periodic Retraining Disable - This option allows customers to disable LPDDR4 Periodic Retraining for debug purposes.

- UINT8 EnableResetSystem

  Offset 0x014B - Enable Reset System Enable FSP to trigger reset instead of returning reset request.

- UINT8 EnableS3Heci2

  Offset 0x014C - Enable HECI2 in S3 resume path Enable HECI2 in S3 resume path.

- UINT8 UnusedUpdSpace1 [3]

  Offset 0x014D.

- VOID * VariableNvsBufferPtr

  Offset 0x0150.

- UINT64 StartTimerTickerOfPfetAssert

  Offset 0x0154 - PCIE SLOT Power Enable Assert Time - PFET.

- UINT8 RtEn

  Offset 0x015C - Real Time Enabling Real-Time Feature Configuration Bits settings.

- UINT8 SkipPciePowerSequence

  Offset 0x015D - Skip Pcie Power Sequence UPD To Skip PciePowerSequence in FSP if set this UPD is set to 1.

- UINT8 RefreshWm

  Offset 0x015E - Refresh Watermark Set the value for Refresh Watermark, bit [7:4] - REFWMPNC, bit [3:0] - REF↩ WMHI.

- UINT8 ReservedFspmUpd

  Offset 0x015F.

6.3.1 Detailed Description

Fsp M Configuration.

Definition at line 79 of file FspmUpd.h.

6.3.2 Member Data Documentation

6.3.2.1 UINT8 FSP_M_CONFIG::Ch0_Bit_swizzling[32]

Offset 0x0088 - Ch0_Bit_swizzling Channel 0 PHY to DUnit DQ mapping (only used if not 1-1 mapping)Range: 0-32.

Frequently asked questions: Q: The DQS (strobes) need to go with the corresponding byte lanes on the DDR module. Are the DQS being swapped around as well? Ans: Yes, DQ strobes need to follow the DQ byte lane they correspond too. So for example if you have DQ[7:0] swapped with DQ[15:8], DQS0 pair also need to be swapped with DQS1 pair. Also, the spreadsheet used for Amenia is essentially a swizzle value lookup that specifies what DRAM DQ bit a particular SoC DQ bit is connected to. Some confusion can arrise from the fact that the indexes to the array do not necessarily map 1:1 to an SoC DQ pin. For example, the CH0 array at index 0 maps to SoC DQB8. The value of 9 at index 0 tells us that SoC DQB8 is connected to DRAM DQA9. Q: The PDG indicates a 2 physical channels need to be stuffed and operated together. Are the CHx_A and CHx_B physical channels operated in tandem or completely separate? If separate, why requirement of pairing them? Ans: We have 2 PHY instances on the SoC each supporting up to 2 x32 LP4 channels. If you have 4 channels both PHYs are active, but if you have 2 channels in order to power gate one PHY, those two channel populated must be on one PHY instance. So yes all channels are independent of each other, but there are some restrictions on how they need to be populated. Q: How is it that an LPDDR4 device is identified as having a x16 width when all 32-bits are used at the same time with a single chip select? That's effectively a x32 device. Ans:LPDDR4 DRAM devices are x16. Each die has 2 x16 devices on them. To make a x32 channel the CS of the two devices in the same die are connected together to make a single rank of one x32 channel (SDP). The second die in the DDP package makes the second rank.

Definition at line 651 of file FspmUpd.h.

6.3.2.2 UINT8 FSP_M_CONFIG::Ch0_DeviceWidth

Offset 0x0066 - Ch0_DeviceWidth NOTE: Only for memory down.

Must specify the DRAM device width per DRAM channel (not to be confused with the SoC Memory Channel width which is always x32 for LPDDR3 and x64 for DDR3L). LPDDR4 devices typically have two channels per die and a x16 device width: 00 - x8; 01 - x16; 10 - x32; 11 - x64 0b0000:x8, 0b0001:x16, 0b0010:x32, 0b0011:x64

Definition at line 307 of file FspmUpd.h.

6.3.2.3 UINT8 FSP_M_CONFIG::Ch0_DramDensity

Offset 0x0067 - Ch0_DramDensity NOTE: Only for memory down.

For LPDDR3 and LPDDR4: Must specify the DRAM device density per rank (per Chip Select). The simplest way of identifying the density per rank is to divide the total SoC memory channel density by the number of ranks. For DDR3L: Must specify the DRAM device density per DRAM device. For example, an 8GB 2Rx8 configuration will utilize sixteen 4Gb density DRAMS. In this configuration, a 4Gb density setting would be selected in the MRC: 000 - 4Gb; 001 - 6Gb; 010 - 8Gb; 011 - 12Gb; 100 - 16Gb; 101 - 2Gb; 110-111 - Reserved 0b0000:4Gb, 0b0001:6Gb, 0b0010:8Gb, 0b0011:12Gb, 0b0100:16Gb

Definition at line 319 of file FspmUpd.h.

6.3.2.4 UINT8 FSP_M_CONFIG::Ch0_Mode2N

Offset 0x006B - Ch0_Mode2N DDR3L Only: Configures the DDR3L command timing mode.

2N Mode is a stretched command mode that provides more setup and hold time for DRAM commands on the DRAM command bus. This is useful for platforms with unusual CMD bus routing or marginal signal integrity: 0 - Auto (1N or 2N mode is automatically selected during Command and Control training), 1 - Force 2N Mode 0x0:Auto, 0x1:Force 2N CMD Timing Mode

Definition at line 365 of file FspmUpd.h.

6.3.2.5 UINT8 FSP_M_CONFIG::Ch0_OdtConfig

Offset 0x0069 - Ch0_OdtConfig [0] RX ODT - DDR3L & LPDDR3 only: Change the READ ODT strength , for SOC termination during a READ transaction, ON DQ BITs.

STRONG ==> 60 OHMS roughly, WEAK ==> 120 OHMS or so roughly. Purpose: Save power on these technologies which burn power directly proportional to ODT strength, because ODT looks like a PU and PD (e.g. a resistor divider, which always burns power when ODT is ON). 0 - WEAK_ODT_CONFIG, 1 - STRONG_ODT_CONFIG. LPDDR4: X - Don't Care. [1] CA ODT - LPDDR4 Only: The customer needs to choose this based on their actual board strapping (how they tie the DRAM's ODT PINs). Effect: LPDDR4 MR11 will be set based on this setting. CAODT_A_B_HIGH_LOW ==> MR11 = 0x34, which is CA ODT = 80 ohms. CAODT_A_B_HIGH_HIGH ==> M↩R11 = 0x24, which is CA ODT = 120 ohms (results in 60 ohm final effective impedance on CA/CLK/CS signals). Purpose: To improve signal integrity and provide a much more optimized CA VREF value during training. Not to save power. 0 - ODT_AB_HIGH_LOW (default), 1 - ODT_AB_HIGH_HIGH. DDR3L & LPDDR3: X - Don't Care. [4] TX ODT. DDR3L only: 0 = RZQ/4 (60 Ohms) = MRC_SMIP_DDR3L_TX_ODT_RTT_WR_60_OHMS, 1 = RZQ/2 (120 Ohms) = MRC_SMIP_DDR3L_TX_ODT_RTT_WR_120_OHMS. LPDDR3 & LPDDR4: X = Don't Care

Definition at line 350 of file FspmUpd.h.

6.3.2.6 UINT8 FSP_M_CONFIG::Ch0_Option

Offset 0x0068 - Ch0_Option BIT[0] Rank Select Interleaving Enable.

See Address Mapping section for full description: 0 - Rank Select Interleaving disabled; 1 - Rank Select Interleaving enabled. BIT[1] Bank Address Hashing Enable. See Address Mapping section for full description: 0 - Bank Address Hashing disabled; 1 - Bank Address Hashing enabled. BIT[2] CH1 CLK Disable. Disables the CH1 CLK PHY Signal when set to 1. This is used on board designs where the CH1 CLK is not routed and left floating or stubbed out: 0 - CH1 CLK is enabled; 1 - CH1 CLK is disabled. BIT[3] Reserved; BIT[5:4] This register specifies the address mapping to be used: 00 - 1KB (A); 01 - 2KB (B)

Definition at line 331 of file FspmUpd.h.

6.3.2.7 UINT8 FSP_M_CONFIG::Ch0_RankEnable

Offset 0x0065 - Ch0_RankEnable NOTE: Only for memory down.

This is a bit mask which specifies what ranks are enabled. NOTE: Only for memory down (soldered down memory with no SPD): BIT[0] Enable Rank 0: Must be set to 1 to enable use of this rank; BIT1[1] Enable Rank 1: Must be set to 1 to enable use of this rank.

Definition at line 298 of file FspmUpd.h.

6.3.2.8 UINT8 FSP_M_CONFIG::Ch1_DeviceWidth

Offset 0x006E - Ch1_DeviceWidth NOTE: Only for memory down.

Must specify the DRAM device width per DRAM channel (not to be confused with the SoC Memory Channel width which is always x32 for LPDDR3 and x64 for DDR3L). LPDDR4 devices typically have two channels per die and a x16 device width: 00 - x8; 01 - x16; 10 - x32; 11 - x64 0b0000:x8, 0b0001:x16, 0b0010:x32, 0b0011:x64

Definition at line 388 of file FspmUpd.h.

6.3.2.9 UINT8 FSP_M_CONFIG::Ch1_DramDensity

Offset 0x006F - Ch1_DramDensity NOTE: Only for memory down.

For LPDDR3 and LPDDR4: Must specify the DRAM device density per rank (per Chip Select). The simplest way of identifying the density per rank is to divide the total SoC memory channel density by the number of ranks. For DDR3L: Must specify the DRAM device density per DRAM device. For example, an 8GB 2Rx8 configuration will utilize sixteen 4Gb density DRAMS. In this configuration, a 4Gb density setting would be selected in the MRC: 000

- 4Gb; 001 - 6Gb; 010 - 8Gb; 011 - 12Gb; 100 - 16Gb; 101 - 2Gb; 110-111 - Reserved 0b0000:4Gb, 0b0001:6Gb, 0b0010:8Gb, 0b0011:12Gb, 0b0100:16Gb

Definition at line 400 of file FspmUpd.h.

6.3.2.10 UINT8 FSP_M_CONFIG::Ch1_Mode2N

Offset 0x0073 - Ch1_Mode2N DDR3L Only: Configures the DDR3L command timing mode.

2N Mode is a stretched command mode that provides more setup and hold time for DRAM commands on the DRAM command bus. This is useful for platforms with unusual CMD bus routing or marginal signal integrity: 0 - Auto (1N or 2N mode is automatically selected during Command and Control training), 1 - Force 2N Mode 0x0:Auto, 0x1:Force 2N CMD Timing Mode

Definition at line 446 of file FspmUpd.h.

6.3.2.11 UINT8 FSP_M_CONFIG::Ch1_OdtConfig

Offset 0x0071 - Ch1_OdtConfig [0] RX ODT - DDR3L & LPDDR3 only: Change the READ ODT strength , for SOC termination during a READ transaction, ON DQ BITs.

STRONG ==> 60 OHMS roughly, WEAK ==> 120 OHMS or so roughly. Purpose: Save power on these technologies which burn power directly proportional to ODT strength, because ODT looks like a PU and PD (e.g. a resistor divider, which always burns power when ODT is ON). 0 - WEAK_ODT_CONFIG, 1 - STRONG_ODT_CONFIG. LPDDR4: X - Don't Care. [1] CA ODT - LPDDR4 Only: The customer needs to choose this based on their actual board strapping (how they tie the DRAM's ODT PINs). Effect: LPDDR4 MR11 will be set based on this setting. CAODT_A_B_HIGH_LOW ==> MR11 = 0x34, which is CA ODT = 80 ohms. CAODT_A_B_HIGH_HIGH ==> M↩ R11 = 0x24, which is CA ODT = 120 ohms (results in 60 ohm final effective impedance on CA/CLK/CS signals). Purpose: To improve signal integrity and provide a much more optimized CA VREF value during training. Not to save power. 0 - ODT_AB_HIGH_LOW (default), 1 - ODT_AB_HIGH_HIGH. DDR3L & LPDDR3: X - Don't Care. [4] TX ODT. DDR3L only: 0 = RZQ/4 (60 Ohms) = MRC_SMIP_DDR3L_TX_ODT_RTT_WR_60_OHMS, 1 = RZQ/2 (120 Ohms) = MRC_SMIP_DDR3L_TX_ODT_RTT_WR_120_OHMS. LPDDR3 & LPDDR4: X = Don't Care

Definition at line 431 of file FspmUpd.h.

6.3.2.12 UINT8 FSP_M_CONFIG::Ch1_Option

Offset 0x0070 - Ch1_Option BIT[0] Rank Select Interleaving Enable.

See Address Mapping section for full description: 0 - Rank Select Interleaving disabled; 1 - Rank Select Interleaving enabled. BIT[1] Bank Address Hashing Enable. See Address Mapping section for full description: 0 - Bank Address Hashing disabled; 1 - Bank Address Hashing enabled. BIT[2] CH1 CLK Disable. Disables the CH1 CLK PHY Signal when set to 1. This is used on board designs where the CH1 CLK is not routed and left floating or stubbed out: 0 - CH1 CLK is enabled; 1 - CH1 CLK is disabled. BIT[3] Reserved; BIT[5:4] This register specifies the address mapping to be used: 00 - 1KB (A); 01 - 2KB (B)

Definition at line 412 of file FspmUpd.h.

6.3.2.13 UINT8 FSP_M_CONFIG::Ch1_RankEnable

Offset 0x006D - Ch1_RankEnable NOTE: Only for memory down.

This is a bit mask which specifies what ranks are enabled. NOTE: Only for memory down (soldered down memory with no SPD): BIT[0] Enable Rank 0: Must be set to 1 to enable use of this rank; BIT1[1] Enable Rank 1: Must be set to 1 to enable use of this rank.

Definition at line 379 of file FspmUpd.h.

6.3.2.14 UINT8 FSP_M_CONFIG::Ch2_DeviceWidth

Offset 0x0076 - Ch2_DeviceWidth NOTE: Only for memory down.

Must specify the DRAM device width per DRAM channel (not to be confused with the SoC Memory Channel width which is always x32 for LPDDR3 and x64 for DDR3L). LPDDR4 devices typically have two channels per die and a x16 device width: 00 - x8; 01 - x16; 10 - x32; 11 - x64 0b0000:x8, 0b0001:x16, 0b0010:x32, 0b0011:x64

Definition at line 469 of file FspmUpd.h.

6.3.2.15 UINT8 FSP_M_CONFIG::Ch2_DramDensity

Offset 0x0077 - Ch2_DramDensity NOTE: Only for memory down.

For LPDDR3 and LPDDR4: Must specify the DRAM device density per rank (per Chip Select). The simplest way of identifying the density per rank is to divide the total SoC memory channel density by the number of ranks. For DDR3L: Must specify the DRAM device density per DRAM device. For example, an 8GB 2Rx8 configuration will utilize sixteen 4Gb density DRAMS. In this configuration, a 4Gb density setting would be selected in the MRC: 000 - 4Gb; 001 - 6Gb; 010 - 8Gb; 011 - 12Gb; 100 - 16Gb; 101 - 2Gb; 110-111 - Reserved 0b0000:4Gb, 0b0001:6Gb, 0b0010:8Gb, 0b0011:12Gb, 0b0100:16Gb

Definition at line 481 of file FspmUpd.h.

6.3.2.16 UINT8 FSP_M_CONFIG::Ch2_Mode2N

Offset 0x007B - Ch2_Mode2N DDR3L Only: Configures the DDR3L command timing mode.

2N Mode is a stretched command mode that provides more setup and hold time for DRAM commands on the DRAM command bus. This is useful for platforms with unusual CMD bus routing or marginal signal integrity: 0 - Auto (1N or 2N mode is automatically selected during Command and Control training), 1 - Force 2N Mode 0x0:Auto, 0x1:Force 2N CMD Timing Mode

Definition at line 527 of file FspmUpd.h.

6.3.2.17 UINT8 FSP_M_CONFIG::Ch2_OdtConfig

Offset 0x0079 - Ch2_OdtConfig [0] RX ODT - DDR3L & LPDDR3 only: Change the READ ODT strength , for SOC termination during a READ transaction, ON DQ BITs.

STRONG ==> 60 OHMS roughly, WEAK ==> 120 OHMS or so roughly. Purpose: Save power on these technologies which burn power directly proportional to ODT strength, because ODT looks like a PU and PD (e.g. a resistor divider, which always burns power when ODT is ON). 0 - WEAK_ODT_CONFIG, 1 - STRONG_ODT_CONFIG. LPDDR4: X - Don't Care. [1] CA ODT - LPDDR4 Only: The customer needs to choose this based on their actual board strapping (how they tie the DRAM's ODT PINs). Effect: LPDDR4 MR11 will be set based on this setting. CAODT_A_B_HIGH_LOW ==> MR11 = 0x34, which is CA ODT = 80 ohms. CAODT_A_B_HIGH_HIGH ==> M↩ R11 = 0x24, which is CA ODT = 120 ohms (results in 60 ohm final effective impedance on CA/CLK/CS signals). Purpose: To improve signal integrity and provide a much more optimized CA VREF value during training. Not to save power. 0 - ODT_AB_HIGH_LOW (default), 1 - ODT_AB_HIGH_HIGH. DDR3L & LPDDR3: X - Don't Care. [4] TX ODT. DDR3L only: 0 = RZQ/4 (60 Ohms) = MRC_SMIP_DDR3L_TX_ODT_RTT_WR_60_OHMS, 1 = RZQ/2 (120 Ohms) = MRC_SMIP_DDR3L_TX_ODT_RTT_WR_120_OHMS. LPDDR3 & LPDDR4: X = Don't Care

Definition at line 512 of file FspmUpd.h.

6.3.2.18 UINT8 FSP_M_CONFIG::Ch2_Option

Offset 0x0078 - Ch2_Option BIT[0] Rank Select Interleaving Enable.

See Address Mapping section for full description: 0 - Rank Select Interleaving disabled; 1 - Rank Select Interleaving enabled. BIT[1] Bank Address Hashing Enable. See Address Mapping section for full description: 0 - Bank Address Hashing disabled; 1 - Bank Address Hashing enabled. BIT[2] CH1 CLK Disable. Disables the CH1 CLK PHY Signal

when set to 1. This is used on board designs where the CH1 CLK is not routed and left floating or stubbed out: 0 - CH1 CLK is enabled; 1 - CH1 CLK is disabled. BIT[3] Reserved; BIT[5:4] This register specifies the address mapping to be used: 00 - 1KB (A); 01 - 2KB (B)

Definition at line 493 of file FspmUpd.h.

6.3.2.19 UINT8 FSP_M_CONFIG::Ch2_RankEnable

Offset 0x0075 - Ch2_RankEnable NOTE: Only for memory down.

This is a bit mask which specifies what ranks are enabled. NOTE: Only for memory down (soldered down memory with no SPD): BIT[0] Enable Rank 0: Must be set to 1 to enable use of this rank; BIT1[1] Enable Rank 1: Must be set to 1 to enable use of this rank.

Definition at line 460 of file FspmUpd.h.

6.3.2.20 UINT8 FSP_M_CONFIG::Ch3_DeviceWidth

Offset 0x007E - Ch3_DeviceWidth NOTE: Only for memory down.

Must specify the DRAM device width per DRAM channel (not to be confused with the SoC Memory Channel width which is always x32 for LPDDR3 and x64 for DDR3L). LPDDR4 devices typically have two channels per die and a x16 device width: 00 - x8; 01 - x16; 10 - x32; 11 - x64 0b0000:x8, 0b0001:x16, 0b0010:x32, 0b0011:x64

Definition at line 550 of file FspmUpd.h.

6.3.2.21 UINT8 FSP_M_CONFIG::Ch3_DramDensity

Offset 0x007F - Ch3_DramDensity NOTE: Only for memory down.

For LPDDR3 and LPDDR4: Must specify the DRAM device density per rank (per Chip Select). The simplest way of identifying the density per rank is to divide the total SoC memory channel density by the number of ranks. For DDR3L: Must specify the DRAM device density per DRAM device. For example, an 8GB 2Rx8 configuration will utilize sixteen 4Gb density DRAMS. In this configuration, a 4Gb density setting would be selected in the MRC: 000 - 4Gb; 001 - 6Gb; 010 - 8Gb; 011 - 12Gb; 100 - 16Gb; 101 - 2Gb; 110-111 - Reserved 0b0000:4Gb, 0b0001:6Gb, 0b0010:8Gb, 0b0011:12Gb, 0b0100:16Gb

Definition at line 562 of file FspmUpd.h.

6.3.2.22 UINT8 FSP_M_CONFIG::Ch3_Mode2N

Offset 0x0083 - Ch3_Mode2N DDR3L Only: Configures the DDR3L command timing mode.

2N Mode is a stretched command mode that provides more setup and hold time for DRAM commands on the DRAM command bus. This is useful for platforms with unusual CMD bus routing or marginal signal integrity: 0 - Auto (1N or 2N mode is automatically selected during Command and Control training), 1 - Force 2N Mode 0x0:Auto, 0x1:Force 2N CMD Timing Mode

Definition at line 608 of file FspmUpd.h.

6.3.2.23 UINT8 FSP_M_CONFIG::Ch3_OdtConfig

Offset 0x0081 - Ch3_OdtConfig [0] RX ODT - DDR3L & LPDDR3 only: Change the READ ODT strength , for SOC termination during a READ transaction, ON DQ BITs.

STRONG ==> 60 OHMS roughly, WEAK ==> 120 OHMS or so roughly. Purpose: Save power on these technologies which burn power directly proportional to ODT strength, because ODT looks like a PU and PD (e.g. a resistor divider, which always burns power when ODT is ON). 0 - WEAK_ODT_CONFIG, 1 - STRONG_ODT_CONFIG. LPDDR4: X - Don't Care. [1] CA ODT - LPDDR4 Only: The customer needs to choose this based on their actual board strapping (how they tie the DRAM's ODT PINs). Effect: LPDDR4 MR11 will be set based on this setting.

CAODT_A_B_HIGH_LOW ==> MR11 = 0x34, which is CA ODT = 80 ohms. CAODT_A_B_HIGH_HIGH ==> M↩ R11 = 0x24, which is CA ODT = 120 ohms (results in 60 ohm final effective impedance on CA/CLK/CS signals). Purpose: To improve signal integrity and provide a much more optimized CA VREF value during training. Not to save power. 0 - ODT_AB_HIGH_LOW (default), 1 - ODT_AB_HIGH_HIGH. DDR3L & LPDDR3: X - Don't Care. [4] TX ODT. DDR3L only: 0 = RZQ/4 (60 Ohms) = MRC_SMIP_DDR3L_TX_ODT_RTT_WR_60_OHMS, 1 = RZQ/2 (120 Ohms) = MRC_SMIP_DDR3L_TX_ODT_RTT_WR_120_OHMS. LPDDR3 & LPDDR4: X = Don't Care

Definition at line 593 of file FspmUpd.h.

6.3.2.24 UINT8 FSP_M_CONFIG::Ch3_Option

Offset 0x0080 - Ch3_Option BIT[0] Rank Select Interleaving Enable.

See Address Mapping section for full description: 0 - Rank Select Interleaving disabled; 1 - Rank Select Interleaving enabled. BIT[1] Bank Address Hashing Enable. See Address Mapping section for full description: 0 - Bank Address Hashing disabled; 1 - Bank Address Hashing enabled. BIT[2] CH1 CLK Disable. Disables the CH1 CLK PHY Signal when set to 1. This is used on board designs where the CH1 CLK is not routed and left floating or stubbed out: 0 - CH1 CLK is enabled; 1 - CH1 CLK is disabled. BIT[3] Reserved; BIT[5:4] This register specifies the address mapping to be used: 00 - 1KB (A); 01 - 2KB (B)

Definition at line 574 of file FspmUpd.h.

6.3.2.25 UINT8 FSP_M_CONFIG::Ch3_RankEnable

Offset 0x007D - Ch3_RankEnable NOTE: Only for memory down.

This is a bit mask which specifies what ranks are enabled. NOTE: Only for memory down (soldered down memory with no SPD): BIT[0] Enable Rank 0: Must be set to 1 to enable use of this rank; BIT1[1] Enable Rank 1: Must be set to 1 to enable use of this rank.

Definition at line 541 of file FspmUpd.h.

6.3.2.26 UINT16 FSP_M_CONFIG::ChannelHashMask

Offset 0x0054 - ChannelHashMask ChannelHashMask and SliceHashMask allow for the channel hashing algorithm to be modified.

These inputs are not used for configurations where an optimized ChannelHashMask has been provided by the PnP validation teams. 0x00(Default).

Definition at line 218 of file FspmUpd.h.

6.3.2.27 UINT8 FSP_M_CONFIG::ChannelsSlicesEnable

Offset 0x0058 - ChannelsSlicesEnable ChannelSlicesEnable field is not used at all on BXTP.

The Channel Slice Configuration is calculated internally based on the enabled channel configuration. 0x00↩ :Disable(Default), 0x01:Enable. $EN_DIS

Definition at line 233 of file FspmUpd.h.

6.3.2.28 UINT8 FSP_M_CONFIG::DDR3LASR

Offset 0x0051 - DDR3LASR NOTE: Only for memory down.

This is specific to ddr3l and used for refresh adjustment in Self Refresh, does not affect LP4. 0x00:Not Supported(↩ Default), 0x01:Supported. 0x0:Not Supported, 0x1:Supported

Definition at line 193 of file FspmUpd.h.

6.3.2.29 UINT8 FSP_M_CONFIG::DDR3LPageSize

Offset 0x0050 - DDR3LPageSize NOTE: Only for memory down (soldered down memory with no SPD).

0x01:1KB(Default), 0x02:2KB. 0x1:1KB, 0x2:2KB

Definition at line 186 of file FspmUpd.h.

6.3.2.30 UINT8 FSP_M_CONFIG::DIMM0SPDAddress

Offset 0x0063 - DIMM0SPDAddress DIMM0 SPD Address (NOTE: Only for DDR3L only.

Please put 0 for MemoryDown. 0xA0(Default).

Definition at line 285 of file FspmUpd.h.

6.3.2.31 UINT8 FSP_M_CONFIG::DIMM1SPDAddress

Offset 0x0064 - DIMM1SPDAddress DIMM1 SPD Address (NOTE: Only for DDR3L only.

Please put 0 for MemoryDown. 0xA4(Default).

Definition at line 290 of file FspmUpd.h.

6.3.2.32 UINT8 FSP_M_CONFIG::DisableFastBoot

Offset 0x0062 - DisableFastBoot 00:Disabled; Use saved training data (if valid) after first boot(Default), 01:Enabled; Full re-train of memory on every boot.

$EN_DIS

Definition at line 280 of file FspmUpd.h.

6.3.2.33 UINT8 FSP_M_CONFIG::DualRankSupportEnable

Offset 0x005A - DualRankSupportEnable Dual Rank Support Enable.

0x00:Disable, 0x01:Enable(Default). $EN_DIS

Definition at line 246 of file FspmUpd.h.

6.3.2.34 UINT8 FSP_M_CONFIG::eMMCTraceLen

Offset 0x0134 - eMMC Trace Length Select eMMC trace length to load OEM file from when loading OEM file name is specified.

0x0:Long(Default), 0x1:Short. 0x0:Long, 0x1:Short

Definition at line 731 of file FspmUpd.h.

6.3.2.35 UINT8 FSP_M_CONFIG::EnableResetSystem

Offset 0x014B - Enable Reset System Enable FSP to trigger reset instead of returning reset request.

0x00: Return the Return Status from FSP if a reset is required. (default); 0x01: Perform Reset inside FSP instead of returning from the API. 0x0:Disabled, 0x1:Eabled

Definition at line 834 of file FspmUpd.h.

6.3.2.36 UINT8 FSP_M_CONFIG::EnableS3Heci2

Offset 0x014C - Enable HECI2 in S3 resume path Enable HECI2 in S3 resume path.

0x00: Skip HECI2 initialization in S3 resume. ; 0x01: Enable HECI2 in S3 resume path.(Default) 0x0:Disabled, 0x1:Enabled

Definition at line 841 of file FspmUpd.h.

6.3.2.37 UINT8 FSP_M_CONFIG::EnhancePort8xhDecoding

Offset 0x0119 - Enhance the port 8xh decoding Enable/Disable Enhance the port 8xh decoding.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 696 of file FspmUpd.h.

6.3.2.38 UINT8 FSP_M_CONFIG::FwTraceDestination

Offset 0x0138 - FW Trace Destination FW Trace Destination.

1-NPK_TRACE_TO_MEMORY, 2-NPK_TRACE_TO_DCI, 3-NPK_TRACE_TO_BSSB, 4-NPK_TRACE_TO_PT↩I(Default).

Definition at line 756 of file FspmUpd.h.

6.3.2.39 UINT8 FSP_M_CONFIG::FwTraceEn

Offset 0x0137 - FW Trace Enable Enable/Disable FW Trace.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 750 of file FspmUpd.h.

6.3.2.40 UINT8 FSP_M_CONFIG::GttSize

Offset 0x004B - GTT Size Select the GTT Size used by the Internal Graphics Device.

0x1:2 MB, 0x2:4 MB, 0x3:8 MB(Default). 0x1:2 MB, 0x2:4 MB, 0x3:8 MB

Definition at line 141 of file FspmUpd.h.

6.3.2.41 UINT16 FSP_M_CONFIG::HighMemoryMaxValue

Offset 0x0060 - HighMemoryMaxValue High Memory Max Value: This value is used to restrict the amount of memory above 4GB and the calculations based on it.

Value is in MB. Example encodings are: 0x0400:1GB, 0x0800:2GB, 0x1000:4GB, 0x2000:8GB. 0x00(Default).

Definition at line 273 of file FspmUpd.h.

6.3.2.42 UINT8 FSP_M_CONFIG::Igd

Offset 0x0048 - Integrated Graphics Device Enable : Enable Integrated Graphics Device (IGD) when selected as the Primary Video Adaptor.

Disable: Always disable IGD. 0x00:Disable, 0x01:Enable(Default). $EN_DIS

Definition at line 118 of file FspmUpd.h.

6.3.2.43 UINT8 FSP_M_CONFIG::IgdApertureSize

Offset 0x004A - Aperture Size Select the Aperture Size used by the Internal Graphics Device.

0x1:128 MB(Default), 0x2:256 MB, 0x3:512 MB. 0x1:128 MB, 0x2:256 MB, 0x3:512 MB

Definition at line 134 of file FspmUpd.h.

6.3.2.44 UINT8 FSP_M_CONFIG::IgdDvmt50PreAlloc

Offset 0x0049 - DVMT Pre-Allocated Select DVMT 5.0 Pre-Allocated (Fixed) Graphics Memory size used by the Internal Graphics Device.

0x02:64 MB(Default). 0x02:64 MB, 0x03:96 MB, 0x04:128 MB, 0x05:160 MB, 0x06:192 MB, 0x07:224 MB, 0x08↩:256 MB, 0x09:288 MB, 0x0A:320 MB, 0x0B:352 MB, 0x0C:384 MB, 0x0D:416 MB, 0x0E:448 MB, 0x0F:480 MB, 0x10:512 MB

Definition at line 127 of file FspmUpd.h.

6.3.2.45 UINT8 FSP_M_CONFIG::InterleavedMode

Offset 0x0053 - InterleavedMode This field is ignored if one of the PnP channel configurations is used.

If the memory configuration is different, then the field is used directly to populate. 0x00:Disable(Default), 0x02↩:Enable. 0x0:Disable, 0x2:Enable

Definition at line 211 of file FspmUpd.h.

6.3.2.46 UINT16 FSP_M_CONFIG::LowMemoryMaxValue

Offset 0x005E - LowMemoryMaxValue Low Memory Max Value: This value is used to restrict the amount of memory below 4GB and the calculations based on it.

Value is in MB.Example encodings are: 0x400 = 1GB, 0x800 = 2GB, 0x1000 = 4GB, 0x2000 8GB. 0x0000(Default).

Definition at line 266 of file FspmUpd.h.

6.3.2.47 UINT8 FSP_M_CONFIG::MemoryDown

Offset 0x004F - MemoryDown Memory Down.

0x0(Default). 0x0:No, 0x1:Yes, 0x2:1MD+SODIMM (for DDR3L only) ACRD, 0x3:1x32 LPDDR4

Definition at line 180 of file FspmUpd.h.

6.3.2.48 UINT16 FSP_M_CONFIG::MemorySizeLimit

Offset 0x005C - MemorySizeLimit Memory Size Limit: This value is used to restrict the total amount of memory and the calculations based on it.

Value is in MB. Example encodings are: 0x400 = 1GB, 0x800 = 2GB, 0x1000 = 4GB, 0x2000 8GB. 0x0000(Default)

Definition at line 259 of file FspmUpd.h.

6.3.2.49 UINT8 FSP_M_CONFIG::MinRefRate2xEnable

Offset 0x0059 - MinRefRate2xEnable Provided as a means to defend against Row-Hammer attacks.

0x00:Disable(Default), 0x01:Enable. $EN_DIS

Definition at line 240 of file FspmUpd.h.

6.3.2.50 UINT8 FSP_M_CONFIG::MrcDataSaving

Offset 0x011B - MRC Training Data Saving Enable/Disable MRC training data saving in FSP.

0x00:Disable(Default), 0x01:Enable. $EN_DIS

Definition at line 708 of file FspmUpd.h.

6.3.2.51 UINT8 FSP_M_CONFIG::MrcFastBoot

Offset 0x0047 - Memory Fast Boot Enable/Disable MRC fast boot support.

0x00:Disable, 0x01:Enable(Default). $EN_DIS

Definition at line 111 of file FspmUpd.h.

6.3.2.52 UINT32 FSP_M_CONFIG::Msc0Size

Offset 0x013C - Memory Region 0 Buffer Size Memory Region 0 Buffer Size.

0-0MB(Default), 1-1MB, 2-8MB, 3-64MB, 4-128MB, 5-256MB, 6-512MB, 7-1GB.

Definition at line 778 of file FspmUpd.h.

6.3.2.53 UINT8 FSP_M_CONFIG::Msc0Wrap

Offset 0x013A - Memory Region 0 Buffer WrapAround Memory Region 0 Buffer WrapAround.

0-n0-warp, 1-warp(Default).

Definition at line 767 of file FspmUpd.h.

6.3.2.54 UINT8 FSP_M_CONFIG::Msc1Wrap

Offset 0x013B - Memory Region 1 Buffer WrapAround Memory Region 1 Buffer WrapAround.

0-n0-warp, 1-warp(Default).

Definition at line 772 of file FspmUpd.h.

6.3.2.55 UINT32 FSP_M_CONFIG::MsgLevelMask

Offset 0x0108 - MsgLevelMask 32 bits used to mask out debug messages.

Masking out bit 0 mask all other messages.

Definition at line 671 of file FspmUpd.h.

6.3.2.56 UINT8 FSP_M_CONFIG::NpkEn

Offset 0x0136 - Npk Enable Enable/Disable Npk.

0:Disable, 1:Enable, 2:Debugger, 3:Auto(Default). 0:Disable, 1:Enable, 2:Debugger, 3:Auto

Definition at line 744 of file FspmUpd.h.

6.3.2.57 UINT8 FSP_M_CONFIG::OemFileName[16]

Offset 0x0120 - OEM File Name to Load Specify a file name to load from CSE file system after memory is available.

Empty indicates no file needs to be loaded.

Definition at line 720 of file FspmUpd.h.

6.3.2.58 UINT8 FSP_M_CONFIG::Package

Offset 0x004D - Package NOTE: Specifies CA Mapping for all technologies.

Supported CA Mappings: 0 - SODIMM(Default); 1 - BGA; 2 - BGA mirrored (LPDDR3 only); 3 - SODIMM/UDIMM with Rank 1 Mirrored (DDR3L); Refer to the IAFW spec for specific details about each CA mapping. 0x0:SODIMM, 0x1:BGA, 0x2:BGA mirrored (LPDDR3 only), 0x3:SODIMM/UDIMM with Rank 1 Mirrored (DDR3L)

Definition at line 157 of file FspmUpd.h.

6.3.2.59 UINT8 FSP_M_CONFIG::PeriodicRetrainingDisable

Offset 0x014A - Periodic Retraining Disable Periodic Retraining Disable - This option allows customers to disable LPDDR4 Periodic Retraining for debug purposes.

Periodic Retraining should be enabled in production. Periodic retraining allows the platform to operate reliably over a larger voltage and temperature range. This field has no effect for DDR3L and LPDDR3 memory type configurations. 0x00: Enable Periodic Retraining (default); 0x01: Disable Periodic Retraining (debug configuration only) 0x0↩ :Enabled, 0x1:Disabled

Definition at line 826 of file FspmUpd.h.

6.3.2.60 UINT8 FSP_M_CONFIG::PmcMlvl

Offset 0x0148 - PMC Message Level PMC Message Output Verbosity Level.

0, 1(Default), 2-4=2-4.

Definition at line 809 of file FspmUpd.h.

6.3.2.61 UINT8 FSP_M_CONFIG::PreMemGpioTableEntryNum

Offset 0x0118 - PreMem GPIO Table Entry Number.

Currently maximum entry number is 4 Number of Entries in PreMem GPIO Table. 0(Default).

Definition at line 690 of file FspmUpd.h.

6.3.2.62 UINT8 FSP_M_CONFIG::PreMemGpioTablePinNum[4]

Offset 0x0110 - PreMem GPIO Pin Number for each table Number of Pins in each PreMem GPIO Table.

0(Default).

Definition at line 680 of file FspmUpd.h.

6.3.2.63 UINT32 FSP_M_CONFIG::PreMemGpioTablePtr

Offset 0x0114 - PreMem GPIO Table Pointer Pointer to Array of pointers to PreMem GPIO Table.

0x00000000(Default).

Definition at line 685 of file FspmUpd.h.

6.3.2.64 UINT8 FSP_M_CONFIG::PrimaryVideoAdaptor

Offset 0x004C - Primary Display Select which of IGD/PCI Graphics device should be Primary Display.

0x0:AUTO(Default), 0x2:IGD, 0x3:PCI 0x0:AUTO, 0x2:IGD, 0x3:PCI

Definition at line 148 of file FspmUpd.h.

6.3.2.65 UINT8 FSP_M_CONFIG::Profile

Offset 0x004E - Profile Profile list.

0x19(Default). 0x1:WIO2_800_7_8_8, 0x2:WIO2_1066_9_10_10, 0x3:LPDDR3_1066_8_10_10, 0x4:LPDDR3↩
_1333_10_12_12, 0x5:LPDDR3_1600_12_15_15, 0x6:LPDDR3_1866_14_17_17, 0x7:LPDDR3_2133_16_20_20,
0x8:LPDDR4_1066_10_10_10, 0x9:LPDDR4_1600_14_15_15, 0xA:LPDDR4_2133_20_20_20, 0xB:LPDDR4_↩
2400_24_22_22, 0xC:LPDDR4_2666_24_24_24, 0xD:LPDDR4_2933_28_27_27, 0xE:LPDDR4_3200_28_29_29,
0xF:DDR3_1066_6_6_6, 0x10:DDR3_1066_7_7_7, 0x11:DDR3_1066_8_8_8, 0x12:DDR3_1333_7_7_7, 0x13↩
:DDR3_1333_8_8_8, 0x14:DDR3_1333_9_9_9, 0x15:DDR3_1333_10_10_10, 0x16:DDR3_1600_8_8_8, 0x17↩
:DDR3_1600_9_9_9, 0x18:DDR3_1600_10_10_10, 0x19:DDR3_1600_11_11_11, 0x1A:DDR3_1866_10_10_10,
0x1B:DDR3_1866_11_11_11, 0x1C:DDR3_1866_12_12_12, 0x1D:DDR3_1866_13_13_13, 0x1E:DDR3_2133↩
_11_11_11, 0x1F:DDR3_2133_12_12_12, 0x20:DDR3_2133_13_13_13, 0x21:DDR3_2133_14_14_14, 0x22:D↩
DR4_1333_10_10_10, 0x23:DDR4_1600_10_10_10, 0x24:DDR4_1600_11_11_11, 0x25:DDR4_1600_12_12_12,
0x26:DDR4_1866_12_12_12, 0x27:DDR4_1866_13_13_13, 0x28:DDR4_1866_14_14_14, 0x29:DDR4_2133_↩
14_14_14, 0x2A:DDR4_2133_15_15_15, 0x2B:DDR4_2133_16_16_16, 0x2C:DDR4_2400_15_15_15, 0x2D:D↩
DR4_2400_16_16_16, 0x2E:DDR4_2400_17_17_17, 0x2F:DDR4_2400_18_18_18

Definition at line 174 of file FspmUpd.h.

6.3.2.66 UINT8 FSP_M_CONFIG::PtiMode

Offset 0x0144 - PTI Mode PTI Mode.

0-0ff, 1-x4(Default), 2-x8, 3-x12, 4-x16.

Definition at line 789 of file FspmUpd.h.

6.3.2.67 UINT8 FSP_M_CONFIG::PtiSpeed

Offset 0x0146 - PTI Speed PTI Speed.

0-full, 1-half, 2-quarter(Default).

Definition at line 799 of file FspmUpd.h.

6.3.2.68 UINT8 FSP_M_CONFIG::PtiTraining

Offset 0x0145 - PTI Training PTI Training.

0-off(Default), 1-6=1-6.

Definition at line 794 of file FspmUpd.h.

6.3.2.69 UINT8 FSP_M_CONFIG::PunitMlvl

Offset 0x0147 - Punit Message Level Punit Message Output Verbosity Level.

0, 1(Default), 2-4=2-4.

Definition at line 804 of file FspmUpd.h.

6.3.2.70 UINT8 FSP_M_CONFIG::RecoverDump

Offset 0x0139 - NPK Recovery Dump Enable/Disable NPK Recovery Dump.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 762 of file FspmUpd.h.

6.3.2.71 UINT8 FSP_M_CONFIG::RefreshWm

Offset 0x015E - Refresh Watermark Set the value for Refresh Watermark, bit [7:4] - REFWMPNC, bit [3:0] - REF↩ WMHI.

Set to 0x75 by default. Consult with DRAM vendor before modifying.

Definition at line 875 of file FspmUpd.h.

6.3.2.72 UINT8 FSP_M_CONFIG::RmtCheckRun

Offset 0x0085 - RmtCheckRun Parameter used to determine whether to run the margin check.

Bit 0 is used for MINIMUM MARGIN CHECK and bit 1 is used for DEGRADE MARGIN CHECK

Definition at line 620 of file FspmUpd.h.

6.3.2.73 UINT8 FSP_M_CONFIG::RmtMode

Offset 0x005B - RmtMode Rank Margin Tool Mode.

0x00(Default), 0x3(Enabled). 0x0:Disabled, 0x3:Enabled

Definition at line 252 of file FspmUpd.h.

6.3.2.74 UINT8 FSP_M_CONFIG::RtEn

Offset 0x015C - Real Time Enabling Real-Time Feature Configuration Bits settings.

0x0:Disabled (default), 0x1:Enabled $EN_DIS

Definition at line 863 of file FspmUpd.h.

6.3.2.75 UINT8 FSP_M_CONFIG::ScramblerSupport

Offset 0x0052 - ScramblerSupport Scrambler Support - Enable or disable the memory scrambler.

Data scrambling is provided as a means to increase signal integrity/reduce RFI generated by the DRAM interface. This is achieved by randomizing seed that encodes/decodes memory data so repeating a worse case pattern is hard to repeat. 00: Disable Scrambler Support, 01: Enable Scrambler Support $EN_DIS

Definition at line 203 of file FspmUpd.h.

6.3.2.76 UINT32 FSP_M_CONFIG::SerialDebugPortAddress

Offset 0x0040 - Debug Serial Port Base address Debug serial port base address.

This option will be used only when the 'Serial Port Debug Device' option is set to 'External Device'. 0x00000000(Default).

Definition at line 85 of file FspmUpd.h.

6.3.2.77 UINT8 FSP_M_CONFIG::SerialDebugPortDevice

Offset 0x0045 - Serial Port Debug Device Select active serial port device for debug.

For SOC UART devices,'Debug Serial Port Base' options will be ignored. 0x02:SOC UART2(Default). 0:SOC UART0, 1:SOC UART1, 2:SOC UART2, 3:External Device

Definition at line 99 of file FspmUpd.h.

6.3.2.78 UINT8 FSP_M_CONFIG::SerialDebugPortStrideSize

Offset 0x0046 - Debug Serial Port Stride Size Debug serial port register map stride size in bytes.

0x00:1, 0x02:4(Default). 0:1, 2:4

Definition at line 105 of file FspmUpd.h.

6.3.2.79 UINT8 FSP_M_CONFIG::SerialDebugPortType

Offset 0x0044 - Debug Serial Port Type 16550 compatible debug serial port resource type.

NONE means no serial port support. 0x02:MMIO(Default). 0:NONE, 1:I/O, 2:MMIO

Definition at line 92 of file FspmUpd.h.

6.3.2.80 UINT8 FSP_M_CONFIG::SkipCseRbp

Offset 0x0135 - Skip CSE RBP to support zero sized IBB Enable/Disable skip CSE RBP for bootloader which loads IBB without assistance of CSE.

0x00:Disable(Default), 0x01:Enable. $EN_DIS

Definition at line 738 of file FspmUpd.h.

6.3.2.81 UINT8 FSP_M_CONFIG::SkipPciePowerSequence

Offset 0x015D - Skip Pcie Power Sequence UPD To Skip PciePowerSequence in FSP if set this UPD is set to 1.

0x0:Disabled (default), 0x1:Skip 0x0:Disabled, 0x1:Skip

Definition at line 869 of file FspmUpd.h.

6.3.2.82 UINT16 FSP_M_CONFIG::SliceHashMask

Offset 0x0056 - SliceHashMask ChannelHashMask and SliceHashMask allow for the channel hashing algorithm to be modified.

These inputs are not used for configurations where an optimized ChannelHashMask has been provided by the PnP validation teams. 0x00(Default).

Definition at line 225 of file FspmUpd.h.

6.3.2.83 UINT8 FSP_M_CONFIG::SpdWriteEnable

Offset 0x011A - SPD Data Write Enable/Disable SPD data write on the SMBUS.

0x00:Disable(Default), 0x01:Enable. $EN_DIS

Definition at line 702 of file FspmUpd.h.

6.3.2.84 UINT64 FSP_M_CONFIG::StartTimerTickerOfPfetAssert

Offset 0x0154 - PCIE SLOT Power Enable Assert Time - PFET.

ACPI Timer Ticker to measure when PCIE Slot Power is enabled through PFET. FSP will wait for 100ms for the power to be stable, before de-asserting PERST bin. Customer who designed the board PCIE slot Power automatically enabled, can pass value of zero here.

Definition at line 857 of file FspmUpd.h.

6.3.2.85 UINT8 FSP_M_CONFIG::SwTraceEn

Offset 0x0149 - SW Trace Enable Enable/Disable SW Trace.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 815 of file FspmUpd.h.

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

- FspmUpd.h

6.4 FSP_PATCH_TABLE Struct Reference

FSP Patch Table as described in FSP v2.0 Spec section 5.1.5.

`#include <FspHeaderFile.h>`

Public Attributes

- UINT32 Signature

  Byte 0x00: FSP Patch Table Signature "FSPP".
- UINT16 HeaderLength

  Byte 0x04: Size including the PatchData.
- UINT8 HeaderRevision

  Byte 0x06: Revision is set to 0x01.
- UINT8 Reserved

  Byte 0x07: Reserved for future use.
- UINT32 PatchEntryNum

  Byte 0x08: Number of entries to Patch.

6.4.1 Detailed Description

FSP Patch Table as described in FSP v2.0 Spec section 5.1.5.

Definition at line 173 of file FspHeaderFile.h.

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

- FspHeaderFile.h

6.5 FSP_S_CONFIG Struct Reference

Fsp S Configuration.

`#include <FspsUpd.h>`

Public Attributes

- UINT8 ActiveProcessorCores

  Offset 0x0020 - ActiveProcessorCores Number of active cores.
- UINT8 DisableCore1

  Offset 0x0021 - Disable Core1 Disable/Enable Core1.
- UINT8 DisableCore2

  Offset 0x0022 - Disable Core2 Disable/Enable Core2.
- UINT8 DisableCore3

  Offset 0x0023 - Disable Core3 Disable/Enable Core3.
- UINT8 VmxEnable

  Offset 0x0024 - VMX Enable Enable or Disable VMX.
- UINT8 ProcTraceMemSize

  Offset 0x0025 - Memory region allocation for Processor Trace Memory region allocation for Processor Trace, allowed range is from 4K (0x0) to 128MB (0xF); ***0xFF: Disable.***
- UINT8 ProcTraceEnable

  Offset 0x0026 - Enable Processor Trace Enable or Disable Processor Trace feature.
- UINT8 Eist

  Offset 0x0027 - Eist Enable or Disable Intel SpeedStep Technology.
- UINT8 BootPState

  Offset 0x0028 - Boot PState Boot PState with HFM or LFM.
- UINT8 EnableCx

  Offset 0x0029 - CPU power states (C-states) Enable or Disable CPU power states (C-states).
- UINT8 C1e

  Offset 0x002A - Enhanced C-states Enable or Disable Enhanced C-states.
- UINT8 BiProcHot

  Offset 0x002B - Bi-Directional PROCHOT# Enable or Disable Bi-Directional PROCHOT#.
- UINT8 PkgCStateLimit

  Offset 0x002C - Max Pkg Cstate Max Pkg Cstate.
- UINT8 CStateAutoDemotion

  Offset 0x002D - C-State auto-demotion C-State Auto Demotion.
- UINT8 CStateUnDemotion

  Offset 0x002E - C-State un-demotion C-State un-demotion.
- UINT8 MaxCoreCState

  Offset 0x002F - Max Core C-State Max Core C-State.
- UINT8 PkgCStateDemotion

  Offset 0x0030 - Package C-State Demotion Enable or Disable Package Cstate Demotion.
- UINT8 PkgCStateUnDemotion

  Offset 0x0031 - Package C-State Un-demotion Enable or Disable Package Cstate UnDemotion.
- UINT8 TurboMode

  Offset 0x0032 - Turbo Mode Enable or Disable long duration Turbo Mode.
- UINT8 HdaVerbTableEntryNum

  Offset 0x0033 - SC HDA Verb Table Entry Number Number of Entries in Verb Table.
- UINT32 HdaVerbTablePtr

  Offset 0x0034 - SC HDA Verb Table Pointer Pointer to Array of pointers to Verb Table.
- UINT8 P2sbUnhide

  Offset 0x0038 - Enable/Disable P2SB device hidden.
- UINT8 IpuEn

  Offset 0x0039 - IPU Enable/Disable Enable/Disable IPU Device.
- UINT8 IpuAcpiMode

  Offset 0x003A - IMGU ACPI mode selection 0:Auto, 1:IGFX Child device(Default), 2:ACPI device.

- UINT8 ForceWake

  Offset 0x003B - Enable ForceWake Enable/disable ForceWake Models.
- UINT32 GttMmAdr

  Offset 0x003C - GttMmAdr GttMmAdr structure for initialization.
- UINT32 GmAdr

  Offset 0x0040 - GmAdr GmAdr structure for initialization.
- UINT8 PavpLock

  Offset 0x0044 - Enable PavpLock Enable/disable PavpLock.
- UINT8 GraphicsFreqModify

  Offset 0x0045 - Enable GraphicsFreqModify Enable/disable GraphicsFreqModify.
- UINT8 GraphicsFreqReq

  Offset 0x0046 - Enable GraphicsFreqReq Enable/disable GraphicsFreqReq.
- UINT8 GraphicsVideoFreq

  Offset 0x0047 - Enable GraphicsVideoFreq Enable/disable GraphicsVideoFreq.
- UINT8 PmLock

  Offset 0x0048 - Enable PmLock Enable/disable PmLock.
- UINT8 DopClockGating

  Offset 0x0049 - Enable DopClockGating Enable/disable DopClockGating.
- UINT8 UnsolicitedAttackOverride

  Offset 0x004A - Enable UnsolicitedAttackOverride Enable/disable UnsolicitedAttackOverride.
- UINT8 WOPCMSupport

  Offset 0x004B - Enable WOPCMSupport Enable/disable WOPCMSupport.
- UINT8 WOPCMSize

  Offset 0x004C - Enable WOPCMSize Enable/disable WOPCMSize.
- UINT8 PowerGating

  Offset 0x004D - Enable PowerGating Enable/disable PowerGating.
- UINT8 UnitLevelClockGating

  Offset 0x004E - Enable UnitLevelClockGating Enable/disable UnitLevelClockGating.
- UINT8 FastBoot

  Offset 0x004F - Enable FastBoot Enable/disable FastBoot.
- UINT8 DynSR

  Offset 0x0050 - Enable DynSR Enable/disable DynSR.
- UINT8 SaIpuEnable

  Offset 0x0051 - Enable SaIpuEnable Enable/disable SaIpuEnable.
- UINT8 PmSupport

  Offset 0x0052 - GT PM Support Enable/Disable GT power management support.
- UINT8 EnableRenderStandby

  Offset 0x0053 - RC6(Render Standby) Enable/Disable render standby support.
- UINT32 LogoSize

  Offset 0x0054 - BMP Logo Data Size BMP logo data buffer size.
- UINT32 LogoPtr

  Offset 0x0058 - BMP Logo Data Pointer BMP logo data pointer to a BMP format buffer.
- UINT32 GraphicsConfigPtr

  Offset 0x005C - Graphics Configuration Data Pointer Graphics configuration data used for initialization.
- UINT8 PavpEnable

  Offset 0x0060 - PAVP Enable Enable/Disable Protected Audio Visual Path (PAVP).
- UINT8 PavpPr3

  Offset 0x0061 - PAVP PR3 Enable/Disable PAVP PR3 0:Disable, 1:Enable(Default).
- UINT8 CdClock

  Offset 0x0062 - CdClock Frequency selection 0:144MHz, 1:288MHz, 2:384MHz, 3:576MHz, 4:624MHz(Default).
- UINT8 PeiGraphicsPeimInit

Offset 0x0063 - Enable/Disable PeiGraphicsPeimInit Enable/Disable PeiGraphicsPeimInit 0:Disable, 1:Enable(↩ Default).

- UINT8 WriteProtectionEnable [5]

  Offset 0x0064 - Write Protection Support Enable/disable Write Protection.
- UINT8 ReadProtectionEnable [5]

  Offset 0x0069 - Read Protection Support Enable/disable Read Protection.
- UINT16 ProtectedRangeLimit [5]

  Offset 0x006E - Protected Range Limitation The address of the upper limit of protection, 0x0FFFh(Default).
- UINT16 ProtectedRangeBase [5]

  Offset 0x0078 - Protected Range Base The base address of the upper limit of protection.
- UINT8 Gmm

  Offset 0x0082 - Enable SC Gaussian Mixture Models Enable/disable SC Gaussian Mixture Models.
- UINT8 ClkGatingPgcbClkTrunk

  Offset 0x0083 - GMM Clock Gating - PGCB Clock Trunk Enable/disable PGCB Clock Trunk.
- UINT8 ClkGatingSb

  Offset 0x0084 - GMM Clock Gating - Sideband Enable/disable Sideband.
- UINT8 ClkGatingSbClkTrunk

  Offset 0x0085 - GMM Clock Gating - Sideband Enable/disable Sideband.
- UINT8 ClkGatingSbClkPartition

  Offset 0x0086 - GMM Clock Gating - Sideband Clock Partition Enable/disable Sideband Clock Partition.
- UINT8 ClkGatingCore

  Offset 0x0087 - GMM Clock Gating - Core Enable/disable Core.
- UINT8 ClkGatingDma

  Offset 0x0088 - GMM Clock Gating - DMA Enable/disable DMA.
- UINT8 ClkGatingRegAccess

  Offset 0x0089 - GMM Clock Gating - Register Access Enable/disable Register Access.
- UINT8 ClkGatingHost

  Offset 0x008A - GMM Clock Gating - Host Enable/disable Host.
- UINT8 ClkGatingPartition

  Offset 0x008B - GMM Clock Gating - Partition Enable/disable Partition.
- UINT8 ClkGatingTrunk

  Offset 0x008C - Clock Gating - Trunk Enable/disable Trunk.
- UINT8 HdaEnable

  Offset 0x008D - HD Audio Support Enable/disable HDA Audio Feature.
- UINT8 DspEnable

  Offset 0x008E - HD Audio DSP Support Enable/disable HDA Audio DSP Feature.
- UINT8 Pme

  Offset 0x008F - Azalia wake-on-ring Enable/disable Azalia wake-on-ring.
- UINT8 HdAudioIoBufferOwnership

  Offset 0x0090 - HD-Audio I/O Buffer Ownership Set HD-Audio I/O Buffer Ownership.
- UINT8 HdAudioIoBufferVoltage

  Offset 0x0091 - HD-Audio I/O Buffer Voltage HD-Audio I/O Buffer Voltage Mode Selectiton .
- UINT8 HdAudioVcType

  Offset 0x0092 - HD-Audio Virtual Channel Type HD-Audio Virtual Channel Type Selectiton.
- UINT8 HdAudioLinkFrequency

  Offset 0x0093 - HD-Audio Link Frequency HD-Audio Virtual Channel Type Selectiton.
- UINT8 HdAudioIDispLinkFrequency

  Offset 0x0094 - HD-Audio iDisp-Link Frequency HD-Audio iDisp-Link Frequency Selectiton.
- UINT8 HdAudioIDispLinkTmode

  Offset 0x0095 - HD-Audio iDisp-Link T-Mode HD-Audio iDisp-Link T-Mode Selectiton.
- UINT8 DspEndpointDmic

Offset 0x0096 - HD-Audio Disp DMIC HD-Audio Disp DMIC Selectiton.

- UINT8 DspEndpointBluetooth

  Offset 0x0097 - HD-Audio Bluetooth Enable/Disable HD-Audio bluetooth.
- UINT8 DspEndpointI2sSkp

  Offset 0x0098 - HD-Audio I2S SHK Enable/Disable HD-Audio I2S SHK.
- UINT8 DspEndpointI2sHp

  Offset 0x0099 - HD-Audio I2S HP Enable/Disable HD-Audio I2S HP.
- UINT8 AudioCtlPwrGate

  Offset 0x009A - HD-Audio Controller Power Gating Enable/Disable HD-Audio Controller Power Gating.
- UINT8 AudioDspPwrGate

  Offset 0x009B - HD-Audio ADSP Power Gating Enable/Disable HD-Audio ADSP Power Gating.
- UINT8 Mmt

  Offset 0x009C - HD-Audio CSME Memory Transfers Enable/Disable HD-Audio CSME Memory Transfers.
- UINT8 Hmt

  Offset 0x009D - HD-Audio Host Memory Transfers Enable/Disable HD-Audio Host Memory Transfers.
- UINT8 HDAudioPwrGate

  Offset 0x009E - HD-Audio Power Gating Enable/Disable HD-Audio BIOS Configuration Lock Down.
- UINT8 HDAudioClkGate

  Offset 0x009F - HD-Audio Clock Gatingn Enable/Disable HD-Audio Clock Gating.
- UINT32 DspFeatureMask

  Offset 0x00A0 - Bitmask of DSP Feature Set Bitmask of HD-Audio DSP Feature.
- UINT32 DspPpModuleMask

  Offset 0x00A4 - Bitmask of supported DSP Post-Processing Modules Set HD-Audio Bitmask of supported DSP Post-Processing Modules.
- UINT8 BiosCfgLockDown

  Offset 0x00A8 - HD-Audio BIOS Configuration Lock Down Enable/Disable HD-Audio BIOS Configuration Lock Down.
- UINT8 Hpet

  Offset 0x00A9 - Enable High Precision Timer Enable/Disable Hpet.
- UINT8 HpetBdfValid

  Offset 0x00AA - Hpet Valid BDF Value Enable/Disable Hpet Valid BDF Value.
- UINT8 HpetBusNumber

  Offset 0x00AB - Bus Number of Hpet Completer ID of Bus Number of Hpet.
- UINT8 HpetDeviceNumber

  Offset 0x00AC - Device Number of Hpet Completer ID of Device Number of Hpet.
- UINT8 HpetFunctionNumber

  Offset 0x00AD - Function Number of Hpet Completer ID of Function Number of Hpet.
- UINT8 IoApicBdfValid

  Offset 0x00AE - IoApic Valid BDF Value Enable/Disable IoApic Valid BDF Value.
- UINT8 IoApicBusNumber

  Offset 0x00AF - Bus Number of IoApic Completer ID of Bus Number of IoApic.
- UINT8 IoApicDeviceNumber

  Offset 0x00B0 - Device Number of IoApic Completer ID of Device Number of IoApic.
- UINT8 IoApicFunctionNumber

  Offset 0x00B1 - Function Number of IoApic Completer ID of Function Number of IoApic.
- UINT8 IoApicEntry24_119

  Offset 0x00B2 - IOAPIC Entry 24-119 Enable/Disable IOAPIC Entry 24-119.
- UINT8 IoApicId

  Offset 0x00B3 - IO APIC ID This member determines IOAPIC ID.
- UINT8 IoApicRangeSelect

  Offset 0x00B4 - IoApic Range Define address bits 19:12 for the IOxAPIC range.
- UINT8 IshEnable

Offset 0x00B5 - ISH Controller Enable/Disable ISH Controller.

- UINT8 BiosInterface

  Offset 0x00B6 - BIOS Interface Lock Down Enable/Disable BIOS Interface Lock Down bit to prevent writes to the Backup Control Register.

- UINT8 BiosLock

  Offset 0x00B7 - Bios LockDown Enable Enable the BIOS Lock Enable (BLE) feature and set EISS bit.

- UINT8 SpiEiss

  Offset 0x00B8 - SPI EISS Status Enable/Disable InSMM.STS (EISS) in SPI.

- UINT8 BiosLockSwSmiNumber

  Offset 0x00B9 - BiosLock SWSMI Number This member describes the SwSmi value for Bios Lock.

- UINT8 LPSS_S0ixEnable

  Offset 0x00BA - LPSS IOSF PMCTL S0ix Enable Enable/Disable LPSS IOSF Bridge PMCTL Register S0ix Bits.

- UINT8 UnusedUpdSpace0 [1]

  Offset 0x00BB.

- UINT8 I2cClkGateCfg [8]

  Offset 0x00BC - LPSS I2C Clock Gating Configuration Enable/Disable LPSS I2C Clock Gating.

- UINT8 HsuartClkGateCfg [4]

  Offset 0x00C4 - PSS HSUART Clock Gating Configuration Enable/Disable LPSS HSUART Clock Gating.

- UINT8 SpiClkGateCfg [3]

  Offset 0x00C8 - LPSS SPI Clock Gating Configuration Enable/Disable LPSS SPI Clock Gating.

- UINT8 I2c0Enable

  Offset 0x00CB - I2C Device 0 Enable/Disable I2C Device 0.

- UINT8 I2c1Enable

  Offset 0x00CC - I2C Device 1 Enable/Disable I2C Device 1.

- UINT8 I2c2Enable

  Offset 0x00CD - I2C Device 2 Enable/Disable I2C Device 2.

- UINT8 I2c3Enable

  Offset 0x00CE - I2C Device 3 Enable/Disable I2C Device 3.

- UINT8 I2c4Enable

  Offset 0x00CF - I2C Device 4 Enable/Disable I2C Device 4.

- UINT8 I2c5Enable

  Offset 0x00D0 - I2C Device 5 Enable/Disable I2C Device 5.

- UINT8 I2c6Enable

  Offset 0x00D1 - I2C Device 6 Enable/Disable I2C Device 6.

- UINT8 I2c7Enable

  Offset 0x00D2 - I2C Device 7 Enable/Disable I2C Device 7.

- UINT8 Hsuart0Enable

  Offset 0x00D3 - UART Device 0 Enable/Disable UART Device 0.

- UINT8 Hsuart1Enable

  Offset 0x00D4 - UART Device 1 Enable/Disable UART Device 1.

- UINT8 Hsuart2Enable

  Offset 0x00D5 - UART Device 2 Enable/Disable UART Device 2.

- UINT8 Hsuart3Enable

  Offset 0x00D6 - UART Device 3 Enable/Disable UART Device 3.

- UINT8 Spi0Enable

  Offset 0x00D7 - SPI UART Device 0 Enable/Disable SPI Device 0.

- UINT8 Spi1Enable

  Offset 0x00D8 - SPI UART Device 1 Enable/Disable SPI Device 1.

- UINT8 Spi2Enable

  Offset 0x00D9 - SPI UART Device 2 Enable/Disable SPI Device 2.

- UINT8 OsDbgEnable

Offset 0x00DA - OS Debug Feature Enable/Disable OS Debug Feature.

- UINT8 DciEn

  Offset 0x00DB - DCI Feature Enable/Disable DCI Feature.
- UINT32 Uart2KernelDebugBaseAddress

  Offset 0x00DC - UART Debug Base Address UART Debug Base Address.
- UINT8 PcieClockGatingDisabled

  Offset 0x00E0 - Enable PCIE Clock Gating Enable/disable PCIE Clock Gating.
- UINT8 PcieRootPort8xhDecode

  Offset 0x00E1 - Enable PCIE Root Port 8xh Decode Enable/disable PCIE Root Port 8xh Decode.
- UINT8 Pcie8xhDecodePortIndex

  Offset 0x00E2 - PCIE 8xh Decode Port Index PCIE 8xh Decode Port Index.
- UINT8 PcieRootPortPeerMemoryWriteEnable

  Offset 0x00E3 - Enable PCIE Root Port Peer Memory Write Enable/disable PCIE root port peer memory write.
- UINT8 PcieAspmSwSmiNumber

  Offset 0x00E4 - PCIE SWSMI Number This member describes the SwSmi value for override PCIe ASPM table.
- UINT8 UnusedUpdSpace1 [1]

  Offset 0x00E5.
- UINT8 PcieRootPortEn [6]

  Offset 0x00E6 - PCI Express Root Port Control the PCI Express Root Port .
- UINT8 PcieRpHide [6]

  Offset 0x00EC - Hide PCIE Root Port Configuration Space Enable/disable Hide PCIE Root Port Configuration Space.
- UINT8 PcieRpSlotImplemented [6]

  Offset 0x00F2 - PCIE Root Port Slot Implement Enable/disable PCIE Root Port Slot Implement.
- UINT8 PcieRpHotPlug [6]

  Offset 0x00F8 - Hot Plug PCI Express Hot Plug Enable/Disable.
- UINT8 PcieRpPmSci [6]

  Offset 0x00FE - PCIE PM SCI Enable/Disable PCI Express PME SCI.
- UINT8 PcieRpExtSync [6]

  Offset 0x0104 - PCIE Root Port Extended Sync Enable/Disable PCIE Root Port Extended Sync.
- UINT8 PcieRpTransmitterHalfSwing [6]

  Offset 0x010A - Transmitter Half Swing Transmitter Half Swing Enable/Disable.
- UINT8 PcieRpAcsEnabled [6]

  Offset 0x0110 - ACS Enable/Disable Access Control Services Extended Capability.
- UINT8 PcieRpClkReqSupported [6]

  Offset 0x0116 - Clock Request Support Enable/Disable CLKREQ# Support.
- UINT8 PcieRpClkReqNumber [6]

  Offset 0x011C - Configure CLKREQ Number Configure Root Port CLKREQ Number if CLKREQ is supported.
- UINT8 PcieRpClkReqDetect [6]

  Offset 0x0122 - CLKREQ# Detection Enable/Disable CLKREQ# Detection Probe.
- UINT8 AdvancedErrorReporting [6]

  Offset 0x0128 - Advanced Error Reporting Enable/Disable Advanced Error Reporting.
- UINT8 PmeInterrupt [6]

  Offset 0x012E - PME Interrupt Enable/Disable PME Interrupt.
- UINT8 UnsupportedRequestReport [6]

  Offset 0x0134 - URR PCI Express Unsupported Request Reporting Enable/Disable.
- UINT8 FatalErrorReport [6]

  Offset 0x013A - FER PCI Express Device Fatal Error Reporting Enable/Disable.
- UINT8 NoFatalErrorReport [6]

  Offset 0x0140 - NFER PCI Express Device Non-Fatal Error Reporting Enable/Disable.
- UINT8 CorrectableErrorReport [6]

  Offset 0x0146 - CER PCI Express Device Correctable Error Reporting Enable/Disable.

- UINT8 SystemErrorOnFatalError [6]

  Offset 0x014C - SEFE Root PCI Express System Error on Fatal Error Enable/Disable.
- UINT8 SystemErrorOnNonFatalError [6]

  Offset 0x0152 - SENFE Root PCI Express System Error on Non-Fatal Error Enable/Disable.
- UINT8 SystemErrorOnCorrectableError [6]

  Offset 0x0158 - SECE Root PCI Express System Error on Correctable Error Enable/Disable.
- UINT8 PcieRpSpeed [6]

  Offset 0x015E - PCIe Speed Configure PCIe Speed.
- UINT8 PhysicalSlotNumber [6]

  Offset 0x0164 - Physical Slot Number Physical Slot Number for PCIE Root Port.
- UINT8 PcieRpCompletionTimeout [6]

  Offset 0x016A - CTO Enable/Disable PCI Express Completion Timer TO .
- UINT8 PtmEnable [6]

  Offset 0x0170 - PTM Support Enable/Disable PTM Support.
- UINT8 PcieRpAspm [6]

  Offset 0x0176 - ASPM PCI Express Active State Power Management settings.
- UINT8 PcieRpL1Substates [6]

  Offset 0x017C - L1 Substates PCI Express L1 Substates settings.
- UINT8 PcieRpLtrEnable [6]

  Offset 0x0182 - PCH PCIe LTR PCH PCIE Latency Reporting Enable/Disable.
- UINT8 PcieRpLtrConfigLock [6]

  Offset 0x0188 - PCIE LTR Lock PCIE LTR Configuration Lock.
- UINT8 PmeB0S5Dis

  Offset 0x018E - PME_B0_S5 Disable bit PME_B0_S5_DIS bit in the General PM Configuration B (GEN_PMCON_B) register.
- UINT8 PciClockRun

  Offset 0x018F - PCI Clock Run This member describes whether or not the PCI ClockRun feature of SC should be enabled.
- UINT8 Timer8254ClkSetting

  Offset 0x0190 - Enable/Disable Timer 8254 Clock Setting Enable/Disable Timer 8254 Clock.
- UINT8 EnableSata

  Offset 0x0191 - Chipset SATA Enables or Disables the Chipset SATA Controller.
- UINT8 SataMode

  Offset 0x0192 - SATA Mode Selection Determines how SATA controller(s) operate.
- UINT8 SataSalpSupport

  Offset 0x0193 - Aggressive LPM Support Enable PCH to aggressively enter link power state.
- UINT8 SataPwrOptEnable

  Offset 0x0194 - SATA Power Optimization Enable SATA Power Optimizer on SC side.
- UINT8 eSATASpeedLimit

  Offset 0x0195 - eSATA Speed Limit Enable/Disable eSATA Speed Limit.
- UINT8 SpeedLimit

  Offset 0x0196 - SATA Speed Limit SATA Speed Limit.
- UINT8 UnusedUpdSpace2 [1]

  Offset 0x0197.
- UINT8 SataPortsEnable [2]

  Offset 0x0198 - SATA Port Enable or Disable SATA Port.
- UINT8 SataPortsDevSlp [2]

  Offset 0x019A - SATA Port DevSlp Enable/Disable SATA Port DevSlp.
- UINT8 SataPortsHotPlug [2]

  Offset 0x019C - SATA Port HotPlug Enable/Disable SATA Port Hotplug .
- UINT8 SataPortsInterlockSw [2]

Offset 0x019E - Mechanical Presence Switch Controls reporting if this port has an Mechanical Presence Switch.

- UINT8 SataPortsExternal [2]

  Offset 0x01A0 - External SATA Ports Enable/Disable External SATA Ports.
- UINT8 SataPortsSpinUp [2]

  Offset 0x01A2 - Spin Up Device Enable/Disable device spin up at boot on selected Sata Ports.
- UINT8 SataPortsSolidStateDrive [2]

  Offset 0x01A4 - SATA Solid State Identify the SATA port is connected to Solid State Drive or Hard Disk Drive.
- UINT8 SataPortsEnableDitoConfig [2]

  Offset 0x01A6 - DITO Configuration Enable/Disable DITO Configuration.
- UINT8 SataPortsDmVal [2]

  Offset 0x01A8 - DM Value DM Value.
- UINT8 UnusedUpdSpace3 [2]

  Offset 0x01AA.
- UINT16 SataPortsDitoVal [2]

  Offset 0x01AC - DITO Value DEVSLP Idle Timeout Value.
- UINT16 SubSystemVendorId

  Offset 0x01B0 - Subsystem Vendor ID Subsystem Vendor ID.
- UINT16 SubSystemId

  Offset 0x01B2 - Subsystem ID Subsystem ID.
- UINT8 CRIDSettings

  Offset 0x01B4 - CRIDSettings PMC CRID setting.
- UINT8 ResetSelect

  Offset 0x01B5 - ResetSelect ResetSelect.
- UINT8 SdcardEnabled

  Offset 0x01B6 - SD Card Support (D27:F0) Enable/Disable SD Card Support.
- UINT8 eMMCEnabled

  Offset 0x01B7 - SeMMC Support (D28:F0) Enable/Disable eMMC Support.
- UINT8 eMMCHostMaxSpeed

  Offset 0x01B8 - eMMC Max Speed Select the eMMC max Speed allowed.
- UINT8 UfsEnabled

  Offset 0x01B9 - UFS Support (D29:F0) Enable/Disable SDIO Support.
- UINT8 SdioEnabled

  Offset 0x01BA - SDIO Support (D30:F0) Enable/Disable SDIO Support.
- UINT8 GppLock

  Offset 0x01BB - GPP Lock Feature Enable/Disable GPP lock.
- UINT8 SirqEnable

  Offset 0x01BC - Serial IRQ Enable/Disable Serial IRQ.
- UINT8 SirqMode

  Offset 0x01BD - Serial IRQ Mode Serial IRQ Mode Selection.
- UINT8 StartFramePulse

  Offset 0x01BE - Start Frame Pulse Width Start Frame Pulse Width Value.
- UINT8 SmbusEnable

  Offset 0x01BF - Enable SMBus Enable/disable SMBus controller.
- UINT8 ArpEnable

  Offset 0x01C0 - SMBus ARP Support Enable/disable SMBus ARP Support.
- UINT8 UnusedUpdSpace4

  Offset 0x01C1.
- UINT16 NumRsvdSmbusAddresses

  Offset 0x01C2 - SMBus Table Elements The number of elements in the Reserved SMBus Address Table.
- UINT8 RsvdSmbusAddressTable [128]

  Offset 0x01C4 - Reserved SMBus Address Table Array of addresses reserved for non-ARP-capable SMBus devices.

- UINT8 DisableComplianceMode

  Offset 0x0244 - XHCI Disable Compliance Mode Options to disable XHCI Link Compliance Mode.
- UINT8 UsbPerPortCtl

  Offset 0x0245 - USB Per-Port Control Control each of the USB ports enable/disable.
- UINT8 Usb30Mode

  Offset 0x0246 - xHCI Mode Mode of operation of xHCI controller.
- UINT8 UnusedUpdSpace5 [1]

  Offset 0x0247.
- UINT8 PortUsb20Enable [8]

  Offset 0x0248 - Enable USB2 ports Enable/disable per USB2 ports.
- UINT8 PortUs20bOverCurrentPin [8]

  Offset 0x0250 - USB20 Over Current Pin Over Current Pin number of USB 2.0 Port.
- UINT8 UsbOtg

  Offset 0x0258 - XDCI Support Enable/Disable XDCI.
- UINT8 HsicSupportEnable

  Offset 0x0259 - Enable XHCI HSIC Support Enable/Disable USB HSIC1.
- UINT8 PortUsb30Enable [6]

  Offset 0x025A - Enable USB3 ports Enable/disable per USB3 ports.
- UINT8 PortUs30bOverCurrentPin [6]

  Offset 0x0260 - USB20 Over Current Pin Over Current Pin number of USB 3.0 Port.
- UINT8 SsicPortEnable [2]

  Offset 0x0266 - Enable XHCI SSIC Support Enable/disable XHCI SSIC ports.
- UINT16 DlanePwrGating

  Offset 0x0268 - SSIC Dlane PowerGating Enable/Disable SSIC Data lane Power Gating.
- UINT8 VtdEnable

  Offset 0x026A - VT-d Enable/Disable VT-d.
- UINT8 LockDownGlobalSmi

  Offset 0x026B - SMI Lock bit Enable/Disable SMI_LOCK bit to prevent writes to the Global SMI Enable bit.
- UINT16 ResetWaitTimer

  Offset 0x026C - HDAudio Delay Timer The delay timer after Azalia reset.
- UINT8 RtcLock

  Offset 0x026E - RTC Lock Bits Enable/Disable RTC Lock Bits.
- UINT8 SataTestMode

  Offset 0x026F - SATA Test Mode Selection Enable/Disable SATA Test Mode.
- UINT8 SsicRate [2]

  Offset 0x0270 - XHCI SSIC RATE Set XHCI SSIC1 Rate to A Series or B Series.
- UINT16 DynamicPowerGating

  Offset 0x0272 - SMBus Dynamic Power Gating Enable/Disable SMBus dynamic power gating.
- UINT16 PcieRpLtrMaxSnoopLatency [6]

  Offset 0x0274 - Max Snoop Latency Latency Tolerance Reporting Max Snoop Latency.
- UINT8 PcieRpSnoopLatencyOverrideMode [6]

  Offset 0x0280 - Snoop Latency Override Snoop Latency Override for PCH PCIE.
- UINT8 UnusedUpdSpace6 [2]

  Offset 0x0286.
- UINT16 PcieRpSnoopLatencyOverrideValue [6]

  Offset 0x0288 - Snoop Latency Value LTR Snoop Latency value of PCH PCIE.
- UINT8 PcieRpSnoopLatencyOverrideMultiplier [6]

  Offset 0x0294 - Snoop Latency Multiplier LTR Snoop Latency Multiplier of PCH PCIE.
- UINT8 SkipMpInit

  Offset 0x029A - Skip Multi-Processor Initialization When this is skipped, boot loader must initialize processors before SilicionInit API.

- UINT8 DciAutoDetect

  Offset 0x029B - DCI Auto Detect Deprecated: Enable/disable DCI AUTO mode.
- UINT16 PcieRpLtrMaxNonSnoopLatency [6]

  Offset 0x029C - Max Non-Snoop Latency Latency Tolerance Reporting, Max Non-Snoop Latency.
- UINT8 PcieRpNonSnoopLatencyOverrideMode [6]

  Offset 0x02A8 - Non Snoop Latency Override Non Snoop Latency Override for PCH PCIE.
- UINT8 TcoTimerHaltLock

  Offset 0x02AE - Halt and Lock TCO Timer Halt and Lock the TCO Timer (Watchdog).
- UINT8 PwrBtnOverridePeriod

  Offset 0x02AF - Power Button Override Period specifies how long will PMC wait before initiating a global reset.
- UINT16 PcieRpNonSnoopLatencyOverrideValue [6]

  Offset 0x02B0 - Non Snoop Latency Value LTR Non Snoop Latency value of PCH PCIE.
- UINT8 PcieRpNonSnoopLatencyOverrideMultiplier [6]

  Offset 0x02BC - Non Snoop Latency Multiplier LTR Non Snoop Latency Multiplier of PCH PCIE.
- UINT8 PcieRpSlotPowerLimitScale [6]

  Offset 0x02C2 - PCIE Root Port Slot Power Limit Scale Specifies scale used for slot power limit value.
- UINT8 PcieRpSlotPowerLimitValue [6]

  Offset 0x02C8 - PCIE Root Port Slot Power Limit Value Specifies upper limit on power supplie by slot.
- UINT8 DisableNativePowerButton

  Offset 0x02CE - Power Button Native Mode Disable Disable power button native mode, when 1, this will result in the PMC logic constantly seeing the power button as de-asserted.
- UINT8 PowerButterDebounceMode

  Offset 0x02CF - Power Button Debounce Mode Enable interrupt when PWRBTN# is asserted.
- UINT32 SdioTxCmdCntl

  Offset 0x02D0 - SDIO_TX_CMD_DLL_CNTL SDIO_TX_CMD_DLL_CNTL.
- UINT32 SdioTxDataCntl1

  Offset 0x02D4 - SDIO_TX_DATA_DLL_CNTL1 SDIO_TX_DATA_DLL_CNTL1.
- UINT32 SdioTxDataCntl2

  Offset 0x02D8 - SDIO_TX_DATA_DLL_CNTL2 SDIO_TX_DATA_DLL_CNTL2.
- UINT32 SdioRxCmdDataCntl1

  Offset 0x02DC - SDIO_RX_CMD_DATA_DLL_CNTL1 SDIO_RX_CMD_DATA_DLL_CNTL1.
- UINT32 SdioRxCmdDataCntl2

  Offset 0x02E0 - SDIO_RX_CMD_DATA_DLL_CNTL2 SDIO_RX_CMD_DATA_DLL_CNTL2.
- UINT32 SdcardTxCmdCntl

  Offset 0x02E4 - SDCARD_TX_CMD_DLL_CNTL SDCARD_TX_CMD_DLL_CNTL.
- UINT32 SdcardTxDataCntl1

  Offset 0x02E8 - SDCARD_TX_DATA_DLL_CNTL1 SDCARD_TX_DATA_DLL_CNTL1.
- UINT32 SdcardTxDataCntl2

  Offset 0x02EC - SDCARD_TX_DATA_DLL_CNTL2 SDCARD_TX_DATA_DLL_CNTL2.
- UINT32 SdcardRxCmdDataCntl1

  Offset 0x02F0 - SDCARD_RX_CMD_DATA_DLL_CNTL1 SDCARD_RX_CMD_DATA_DLL_CNTL1.
- UINT32 SdcardRxStrobeCntl

  Offset 0x02F4 - SDCARD_RX_STROBE_DLL_CNTL SDCARD_RX_STROBE_DLL_CNTL.
- UINT32 SdcardRxCmdDataCntl2

  Offset 0x02F8 - SDCARD_RX_CMD_DATA_DLL_CNTL2 SDCARD_RX_CMD_DATA_DLL_CNTL2.
- UINT32 EmmcTxCmdCntl

  Offset 0x02FC - EMMC_TX_CMD_DLL_CNTL EMMC_TX_CMD_DLL_CNTL.
- UINT32 EmmcTxDataCntl1

  Offset 0x0300 - EMMC_TX_DATA_DLL_CNTL1 EMMC_TX_DATA_DLL_CNTL1.
- UINT32 EmmcTxDataCntl2

  Offset 0x0304 - EMMC_TX_DATA_DLL_CNTL2 EMMC_TX_DATA_DLL_CNTL2.

- UINT32 EmmcRxCmdDataCntl1

  Offset 0x0308 - EMMC_RX_CMD_DATA_DLL_CNTL1 EMMC_RX_CMD_DATA_DLL_CNTL1.
- UINT32 EmmcRxStrobeCntl

  Offset 0x030C - EMMC_RX_STROBE_DLL_CNTL EMMC_RX_STROBE_DLL_CNTL.
- UINT32 EmmcRxCmdDataCntl2

  Offset 0x0310 - EMMC_RX_CMD_DATA_DLL_CNTL2 EMMC_RX_CMD_DATA_DLL_CNTL2.
- UINT32 EmmcMasterSwCntl

  Offset 0x0314 - EMMC_MASTER_DLL_CNTL EMMC_MASTER_DLL_CNTL.
- UINT8 PcieRpSelectableDeemphasis [6]

  Offset 0x0318 - PCIe Selectable De-emphasis When the Link is operating at 5.0 GT/s speed, this bit selects the level of de-emphasis for an Upstream component.
- UINT8 MonitorMwaitEnable

  Offset 0x031E - Monitor Mwait Enable Enable/Disable Monitor Mwait.
- UINT8 HdAudioDspUaaCompliance

  Offset 0x031F - 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 IPC [4]

  Offset 0x0320 - IRQ Interrupt Polarity Control Set IRQ Interrupt Polarity Control to ITSS.IPC[0]~IPC[3].
- UINT8 SataPortsDisableDynamicPg [2]

  Offset 0x0330 - Disable ModPHY dynamic power gate Disable ModPHY dynamic power gate for the specific SATA port.
- UINT8 InitS3Cpu

  Offset 0x0332 - Init CPU during S3 resume 0: Do not initialize CPU during S3 resume.
- UINT8 SkipPunitInit

  Offset 0x0333 - Skip P-unit Initialization When this is skipped, boot loader must initialize P-unit before SilicionInit API.
- UINT8 UnusedUpdSpace7 [4]

  Offset 0x0334.
- UINT8 PortUsb20PerPortTxPeHalf [8]

  Offset 0x0338 - PerPort Half Bit Pre-emphasis PerPort Half Bit Pre-emphasis.
- UINT8 PortUsb20PerPortPeTxiSet [8]

  Offset 0x0340 - PerPort HS Pre-emphasis Bias PerPort HS Pre-emphasis Bias.
- UINT8 PortUsb20PerPortTxiSet [8]

  Offset 0x0348 - PerPort HS Transmitter Bias PerPort HS Transmitter Bias.
- UINT8 PortUsb20HsSkewSel [8]

  Offset 0x0350 - Select the skew direction for HS transition Select the skew direction for HS transition.
- UINT8 PortUsb20IUsbTxEmphasisEn [8]

  Offset 0x0358 - Per Port HS Transmitter Emphasis Per Port HS Transmitter Emphasis.
- UINT8 PortUsb20PerPortRXISet [8]

  Offset 0x0360 - PerPort HS Receiver Bias PerPort HS Receiver Bias.
- UINT8 PortUsb20HsNpreDrvSel [8]

  Offset 0x0368 - Delay/skew's strength control for HS driver Delay/skew's strength control for HS driver.
- UINT8 OsSelection

  Offset 0x0370 - OS Selection Windows or Android or Linux OS selection to be used by HDA, USB Common, PWM and PEI Graphics modules.
- UINT8 DptfEnabled

  Offset 0x0371 - DPTF Intel® Dynamic Platform and Thermal Framework.
- UINT8 PWMEnabled

  Offset 0x0372 - PWM Enabled PWM Device Enabling.
- UINT8 P2sbSecEn

  Offset 0x0373 - P2SB Security Option Enable/Disable Enable/Disable P2SB Security Option.
- UINT8 ReservedFspsUpd [12]

  Offset 0x0374.

6.5.1 Detailed Description

Fsp S Configuration.

Definition at line 43 of file FspsUpd.h.

6.5.2 Member Data Documentation

6.5.2.1 UINT8 FSP_S_CONFIG::ActiveProcessorCores

Offset 0x0020 - ActiveProcessorCores Number of active cores.

0:Disable(Default), 1:Enable.

Definition at line 48 of file FspsUpd.h.

6.5.2.2 UINT8 FSP_S_CONFIG::AdvancedErrorReporting[6]

Offset 0x0128 - Advanced Error Reporting Enable/Disable Advanced Error Reporting.

0: Disable(Default), 1: Enable.

Definition at line 879 of file FspsUpd.h.

6.5.2.3 UINT8 FSP_S_CONFIG::ArpEnable

Offset 0x01C0 - SMBus ARP Support Enable/disable SMBus ARP Support.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 1162 of file FspsUpd.h.

6.5.2.4 UINT8 FSP_S_CONFIG::AudioCtlPwrGate

Offset 0x009A - HD-Audio Controller Power Gating Enable/Disable HD-Audio Controller Power Gating.

This option is deprecated. $EN_DIS

Definition at line 507 of file FspsUpd.h.

6.5.2.5 UINT8 FSP_S_CONFIG::AudioDspPwrGate

Offset 0x009B - HD-Audio ADSP Power Gating Enable/Disable HD-Audio ADSP Power Gating.

This option is deprecated. $EN_DIS

Definition at line 513 of file FspsUpd.h.

6.5.2.6 UINT8 FSP_S_CONFIG::BiosCfgLockDown

Offset 0x00A8 - HD-Audio BIOS Configuration Lock Down Enable/Disable HD-Audio BIOS Configuration Lock Down.

0:Disable(Default), 1:Enable. This option is deprecated $EN_DIS

Definition at line 560 of file FspsUpd.h.

6.5.2.7 UINT8 FSP_S_CONFIG::BiosInterface

Offset 0x00B6 - BIOS Interface Lock Down Enable/Disable BIOS Interface Lock Down bit to prevent writes to the Backup Control Register.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 637 of file FspsUpd.h.

6.5.2.8 UINT8 FSP_S_CONFIG::BiosLock

Offset 0x00B7 - Bios LockDown Enable Enable the BIOS Lock Enable (BLE) feature and set EISS bit.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 643 of file FspsUpd.h.

6.5.2.9 UINT8 FSP_S_CONFIG::BiosLockSwSmiNumber

Offset 0x00B9 - BiosLock SWSMI Number This member describes the SwSmi value for Bios Lock.

0xA9(Default).

Definition at line 654 of file FspsUpd.h.

6.5.2.10 UINT8 FSP_S_CONFIG::BiProcHot

Offset 0x002B - Bi-Directional PROCHOT# Enable or Disable Bi-Directional PROCHOT#.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 113 of file FspsUpd.h.

6.5.2.11 UINT8 FSP_S_CONFIG::BootPState

Offset 0x0028 - Boot PState Boot PState with HFM or LFM.

0:HFM(Default), 1:LFM.

Definition at line 95 of file FspsUpd.h.

6.5.2.12 UINT8 FSP_S_CONFIG::C1e

Offset 0x002A - Enhanced C-states Enable or Disable Enhanced C-states.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 107 of file FspsUpd.h.

6.5.2.13 UINT8 FSP_S_CONFIG::CdClock

Offset 0x0062 - CdClock Frequency selection 0:144MHz, 1:288MHz, 2:384MHz, 3:576MHz, 4:624MHz(Default).

0: 144 MHz, 1: 288 MHz, 2: 384 MHz, 3: 576 MHz, 4: 624 MHz

Definition at line 328 of file FspsUpd.h.

6.5.2.14 UINT8 FSP_S_CONFIG::ClkGatingCore

Offset 0x0087 - GMM Clock Gating - Core Enable/disable Core.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 390 of file FspsUpd.h.

6.5.2.15 UINT8 FSP_S_CONFIG::ClkGatingDma

Offset 0x0088 - GMM Clock Gating - DMA Enable/disable DMA.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 396 of file FspsUpd.h.

6.5.2.16 UINT8 FSP_S_CONFIG::ClkGatingHost

Offset 0x008A - GMM Clock Gating - Host Enable/disable Host.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 408 of file FspsUpd.h.

6.5.2.17 UINT8 FSP_S_CONFIG::ClkGatingPartition

Offset 0x008B - GMM Clock Gating - Partition Enable/disable Partition.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 414 of file FspsUpd.h.

6.5.2.18 UINT8 FSP_S_CONFIG::ClkGatingPgcbClkTrunk

Offset 0x0083 - GMM Clock Gating - PGCB Clock Trunk Enable/disable PGCB Clock Trunk.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 366 of file FspsUpd.h.

6.5.2.19 UINT8 FSP_S_CONFIG::ClkGatingRegAccess

Offset 0x0089 - GMM Clock Gating - Register Access Enable/disable Register Access.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 402 of file FspsUpd.h.

6.5.2.20 UINT8 FSP_S_CONFIG::ClkGatingSb

Offset 0x0084 - GMM Clock Gating - Sideband Enable/disable Sideband.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 372 of file FspsUpd.h.

6.5.2.21 UINT8 FSP_S_CONFIG::ClkGatingSbClkPartition

Offset 0x0086 - GMM Clock Gating - Sideband Clock Partition Enable/disable Sideband Clock Partition.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 384 of file FspsUpd.h.

6.5.2.22 UINT8 FSP_S_CONFIG::ClkGatingSbClkTrunk

Offset 0x0085 - GMM Clock Gating - Sideband Enable/disable Sideband.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 378 of file FspsUpd.h.

6.5.2.23 UINT8 FSP_S_CONFIG::ClkGatingTrunk

Offset 0x008C - Clock Gating - Trunk Enable/disable Trunk.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 420 of file FspsUpd.h.

6.5.2.24 UINT8 FSP_S_CONFIG::CorrectableErrorReport[6]

Offset 0x0146 - CER PCI Express Device Correctable Error Reporting Enable/Disable.

0:Disable(Default), 1:Enable.

Definition at line 904 of file FspsUpd.h.

6.5.2.25 UINT8 FSP_S_CONFIG::CRIDSettings

Offset 0x01B4 - CRIDSettings PMC CRID setting.

0:Disable(Default), 1:CRID_1, 2:CRID_2, 3:CRID_3.

Definition at line 1091 of file FspsUpd.h.

6.5.2.26 UINT8 FSP_S_CONFIG::CStateAutoDemotion

Offset 0x002D - C-State auto-demotion C-State Auto Demotion.

0:Disable(Default) C1 and C3 Auto-demotion, 1:Enable C3/C6/C7 Auto-demotion to C1, 2:Enable C6/C7 Auto-demotion to C3, 3:Enable C6/C7 Auto-demotion to C1 and C3.

Definition at line 126 of file FspsUpd.h.

6.5.2.27 UINT8 FSP_S_CONFIG::CStateUnDemotion

Offset 0x002E - C-State un-demotion C-State un-demotion.

0:Disable(Default) C1 and C3 Un-demotion, 1:Enable C1 Un-demotion, 2:Enable C3 Un-demotion, 3:Enable C1 and C3 Un-demotion.

Definition at line 132 of file FspsUpd.h.

6.5.2.28 UINT8 FSP_S_CONFIG::DciAutoDetect

Offset 0x029B - DCI Auto Detect Deprecated: Enable/disable DCI AUTO mode.

Enabled(Default). $EN_DIS

Definition at line 1327 of file FspsUpd.h.

6.5.2.29 UINT8 FSP_S_CONFIG::DciEn

Offset 0x00DB - DCI Feature Enable/Disable DCI Feature.

0:Disable(Default), 1: Enable. $EN_DIS

Definition at line 781 of file FspsUpd.h.

6.5.2.30 UINT8 FSP_S_CONFIG::DisableComplianceMode

Offset 0x0244 - XHCI Disable Compliance Mode Options to disable XHCI Link Compliance Mode.

Default is FALSE to not disable Compliance Mode. Set TRUE to disable Compliance Mode. 0:FALSE(Default), 1:True. $EN_DIS

Definition at line 1183 of file FspsUpd.h.

6.5.2.31 UINT8 FSP_S_CONFIG::DisableCore1

Offset 0x0021 - Disable Core1 Disable/Enable Core1.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 54 of file FspsUpd.h.

6.5.2.32 UINT8 FSP_S_CONFIG::DisableCore2

Offset 0x0022 - Disable Core2 Disable/Enable Core2.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 60 of file FspsUpd.h.

6.5.2.33 UINT8 FSP_S_CONFIG::DisableCore3

Offset 0x0023 - Disable Core3 Disable/Enable Core3.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 66 of file FspsUpd.h.

6.5.2.34 UINT8 FSP_S_CONFIG::DisableNativePowerButton

Offset 0x02CE - Power Button Native Mode Disable Disable power button native mode, when 1, this will result in the PMC logic constantly seeing the power button as de-asserted.

0 (default)) $EN_DIS

Definition at line 1381 of file FspsUpd.h.

6.5.2.35 UINT16 FSP_S_CONFIG::DlanePwrGating

Offset 0x0268 - SSIC Dlane PowerGating Enable/Disable SSIC Data lane Power Gating.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 1245 of file FspsUpd.h.

6.5.2.36 UINT8 FSP_S_CONFIG::DopClockGating

Offset 0x0049 - Enable DopClockGating Enable/disable DopClockGating.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 235 of file FspsUpd.h.

6.5.2.37 UINT8 FSP_S_CONFIG::DptfEnabled

Offset 0x0371 - DPTF Intel® Dynamic Platform and Thermal Framework.

0x0:Disabled (default), 0x1:Enabled $EN_DIS

Definition at line 1572 of file FspsUpd.h.

6.5.2.38 UINT8 FSP_S_CONFIG::DspEnable

Offset 0x008E - HD Audio DSP Support Enable/disable HDA Audio DSP Feature.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 432 of file FspsUpd.h.

6.5.2.39 UINT8 FSP_S_CONFIG::DspEndpointBluetooth

Offset 0x0097 - HD-Audio Bluetooth Enable/Disable HD-Audio bluetooth.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 489 of file FspsUpd.h.

6.5.2.40 UINT8 FSP_S_CONFIG::DspEndpointDmic

Offset 0x0096 - HD-Audio Disp DMIC HD-Audio Disp DMIC Selectiton.

0:Disable, 1:2ch array(Default), 2:4ch array. 0: Disable, 1: 2ch array, 2: 4ch array

Definition at line 483 of file FspsUpd.h.

6.5.2.41 UINT8 FSP_S_CONFIG::DspEndpointI2sHp

Offset 0x0099 - HD-Audio I2S HP Enable/Disable HD-Audio I2S HP.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 501 of file FspsUpd.h.

6.5.2.42 UINT8 FSP_S_CONFIG::DspEndpointI2sSkp

Offset 0x0098 - HD-Audio I2S SHK Enable/Disable HD-Audio I2S SHK.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 495 of file FspsUpd.h.

6.5.2.43 UINT32 FSP_S_CONFIG::DspFeatureMask

Offset 0x00A0 - Bitmask of DSP Feature Set Bitmask of HD-Audio DSP Feature.

0x00000000(Default). [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

Definition at line 545 of file FspsUpd.h.

6.5.2.44 UINT32 FSP_S_CONFIG::DspPpModuleMask

Offset 0x00A4 - Bitmask of supported DSP Post-Processing Modules Set HD-Audio Bitmask of supported DSP Post-Processing Modules.

0x00000000(Default). [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

Definition at line 553 of file FspsUpd.h.

6.5.2.45 UINT16 FSP_S_CONFIG::DynamicPowerGating

Offset 0x0272 - SMBus Dynamic Power Gating Enable/Disable SMBus dynamic power gating.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 1286 of file FspsUpd.h.

6.5.2.46 UINT8 FSP_S_CONFIG::DynSR

Offset 0x0050 - Enable DynSR Enable/disable DynSR.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 277 of file FspsUpd.h.

6.5.2.47 UINT8 FSP_S_CONFIG::Eist

Offset 0x0027 - Eist Enable or Disable Intel SpeedStep Technology.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 90 of file FspsUpd.h.

6.5.2.48 UINT8 FSP_S_CONFIG::eMMCEnabled

Offset 0x01B7 - SeMMC Support (D28:F0) Enable/Disable eMMC Support.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 1108 of file FspsUpd.h.

6.5.2.49 UINT8 FSP_S_CONFIG::eMMCHostMaxSpeed

Offset 0x01B8 - eMMC Max Speed Select the eMMC max Speed allowed.

0:HS400(Default), 1:HS200, 2:DDR50. 0:HS400, 1: HS200, 2:DDR50

Definition at line 1114 of file FspsUpd.h.

6.5.2.50 UINT32 FSP_S_CONFIG::EmmcMasterSwCntl

Offset 0x0314 - EMMC_MASTER_DLL_CNTL EMMC_MASTER_DLL_CNTL.

0x001(Default).

Definition at line 1477 of file FspsUpd.h.

6.5.2.51 UINT32 FSP_S_CONFIG::EmmcRxCmdDataCntl1

Offset 0x0308 - EMMC_RX_CMD_DATA_DLL_CNTL1 EMMC_RX_CMD_DATA_DLL_CNTL1.

0x000D162F(Default).

Definition at line 1462 of file FspsUpd.h.

6.5.2.52 UINT32 FSP_S_CONFIG::EmmcRxCmdDataCntl2

Offset 0x0310 - EMMC_RX_CMD_DATA_DLL_CNTL2 EMMC_RX_CMD_DATA_DLL_CNTL2.

0x1003b(Default).

Definition at line 1472 of file FspsUpd.h.

6.5.2.53 UINT32 FSP_S_CONFIG::EmmcRxStrobeCntl

Offset 0x030C - EMMC_RX_STROBE_DLL_CNTL EMMC_RX_STROBE_DLL_CNTL.

0x0a0a(Default).

Definition at line 1467 of file FspsUpd.h.

6.5.2.54 UINT32 FSP_S_CONFIG::EmmcTxCmdCntl

Offset 0x02FC - EMMC_TX_CMD_DLL_CNTL EMMC_TX_CMD_DLL_CNTL.

0x505(Default).

Definition at line 1447 of file FspsUpd.h.

6.5.2.55 UINT32 FSP_S_CONFIG::EmmcTxDataCntl1

Offset 0x0300 - EMMC_TX_DATA_DLL_CNTL1 EMMC_TX_DATA_DLL_CNTL1.

0xC11(Default).

Definition at line 1452 of file FspsUpd.h.

6.5.2.56 UINT32 FSP_S_CONFIG::EmmcTxDataCntl2

Offset 0x0304 - EMMC_TX_DATA_DLL_CNTL2 EMMC_TX_DATA_DLL_CNTL2.

0x1C2A2927(Default).

Definition at line 1457 of file FspsUpd.h.

6.5.2.57 UINT8 FSP_S_CONFIG::EnableCx

Offset 0x0029 - CPU power states (C-states) Enable or Disable CPU power states (C-states).

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 101 of file FspsUpd.h.

6.5.2.58 UINT8 FSP_S_CONFIG::EnableRenderStandby

Offset 0x0053 - RC6(Render Standby) Enable/Disable render standby support.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 295 of file FspsUpd.h.

6.5.2.59 UINT8 FSP_S_CONFIG::EnableSata

Offset 0x0191 - Chipset SATA Enables or Disables the Chipset SATA Controller.

The Chipset SATA controller supports the 2 black internal SATA ports (up to 3Gb/s supported per port). 0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 986 of file FspsUpd.h.

6.5.2.60 UINT8 FSP_S_CONFIG::eSATASpeedLimit

Offset 0x0195 - eSATA Speed Limit Enable/Disable eSATA Speed Limit.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 1010 of file FspsUpd.h.

6.5.2.61 UINT8 FSP_S_CONFIG::FastBoot

Offset 0x004F - Enable FastBoot Enable/disable FastBoot.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 271 of file FspsUpd.h.

6.5.2.62 UINT8 FSP_S_CONFIG::FatalErrorReport[6]

Offset 0x013A - FER PCI Express Device Fatal Error Reporting Enable/Disable.

0:Disable(Default), 1:Enable.

Definition at line 894 of file FspsUpd.h.

6.5.2.63 UINT8 FSP_S_CONFIG::ForceWake

Offset 0x003B - Enable ForceWake Enable/disable ForceWake Models.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 189 of file FspsUpd.h.

6.5.2.64 UINT32 FSP_S_CONFIG::GmAdr

Offset 0x0040 - GmAdr GmAdr structure for initialization.

0xA0000000(Default).

Definition at line 199 of file FspsUpd.h.

6.5.2.65 UINT8 FSP_S_CONFIG::Gmm

Offset 0x0082 - Enable SC Gaussian Mixture Models Enable/disable SC Gaussian Mixture Models.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 360 of file FspsUpd.h.

6.5.2.66 UINT8 FSP_S_CONFIG::GppLock

Offset 0x01BB - GPP Lock Feature Enable/Disable GPP lock.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 1132 of file FspsUpd.h.

6.5.2.67 UINT32 FSP_S_CONFIG::GraphicsConfigPtr

Offset 0x005C - Graphics Configuration Data Pointer Graphics configuration data used for initialization.

0x00000000(Default).

Definition at line 310 of file FspsUpd.h.

6.5.2.68 UINT8 FSP_S_CONFIG::GraphicsFreqModify

Offset 0x0045 - Enable GraphicsFreqModify Enable/disable GraphicsFreqModify.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 211 of file FspsUpd.h.

6.5.2.69 UINT8 FSP_S_CONFIG::GraphicsFreqReq

Offset 0x0046 - Enable GraphicsFreqReq Enable/disable GraphicsFreqReq.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 217 of file FspsUpd.h.

6.5.2.70 UINT8 FSP_S_CONFIG::GraphicsVideoFreq

Offset 0x0047 - Enable GraphicsVideoFreq Enable/disable GraphicsVideoFreq.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 223 of file FspsUpd.h.

6.5.2.71 UINT32 FSP_S_CONFIG::GttMmAdr

Offset 0x003C - GttMmAdr GttMmAdr structure for initialization.

0xBF000000(Default).

Definition at line 194 of file FspsUpd.h.

6.5.2.72 UINT8 FSP_S_CONFIG::HdaEnable

Offset 0x008D - HD Audio Support Enable/disable HDA Audio Feature.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 426 of file FspsUpd.h.

6.5.2.73 UINT8 FSP_S_CONFIG::HDAudioClkGate

Offset 0x009F - HD-Audio Clock Gatingn Enable/Disable HD-Audio Clock Gating.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 537 of file FspsUpd.h.

6.5.2.74 UINT8 FSP_S_CONFIG::HdAudioDspUaaCompliance

Offset 0x031F - 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).

$EN_DIS

Definition at line 1497 of file FspsUpd.h.

6.5.2.75 UINT8 FSP_S_CONFIG::HdAudioIDispLinkFrequency

Offset 0x0094 - HD-Audio iDisp-Link Frequency HD-Audio iDisp-Link Frequency Selectiton.

0:6MHz(Default), 1:12MHz, 2:24MHz, 3:48MHz, 4:96MHz, 5:Invalid. 0: 6MHz, 1: 12MHz, 2: 24MHz, 3: 48MHz, 4: 96MHz, 5: Invalid

Definition at line 471 of file FspsUpd.h.

6.5.2.76 UINT8 FSP_S_CONFIG::HdAudioIDispLinkTmode

Offset 0x0095 - HD-Audio iDisp-Link T-Mode HD-Audio iDisp-Link T-Mode Selectiton.

0:2T(Default), 1:1T. 0: 2T, 1: 1T

Definition at line 477 of file FspsUpd.h.

6.5.2.77 UINT8 FSP_S_CONFIG::HdAudioIoBufferOwnership

Offset 0x0090 - HD-Audio I/O Buffer Ownership Set HD-Audio I/O Buffer Ownership.

0:HD-Audio link owns all the I/O buffers(Default) 0:HD-Audio link owns all the I/O buffers, 1:HD-Audio link owns 4 I/O buffers and I2S port owns 4 I/O buffers, 3:I2S port owns all the I/O buffers

Definition at line 445 of file FspsUpd.h.

6.5.2.78 UINT8 FSP_S_CONFIG::HdAudioIoBufferVoltage

Offset 0x0091 - HD-Audio I/O Buffer Voltage HD-Audio I/O Buffer Voltage Mode Selectiton .

0:3.3V(Default), 1:1.8V. 0: 3.3V, 1: 1.8V

Definition at line 451 of file FspsUpd.h.

6.5.2.79 UINT8 FSP_S_CONFIG::HdAudioLinkFrequency

Offset 0x0093 - HD-Audio Link Frequency HD-Audio Virtual Channel Type Selectiton.

0:6MHz(Default), 1:12MHz, 2:24MHz, 3:48MHz, 4:96MHz, 5:Invalid. 0: 6MHz, 1: 12MHz, 2: 24MHz, 3: 48MHz, 4: 96MHz, 5: Invalid

Definition at line 464 of file FspsUpd.h.

6.5.2.80 UINT8 FSP_S_CONFIG::HDAudioPwrGate

Offset 0x009E - HD-Audio Power Gating Enable/Disable HD-Audio BIOS Configuration Lock Down.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 531 of file FspsUpd.h.

6.5.2.81 UINT8 FSP_S_CONFIG::HdAudioVcType

Offset 0x0092 - HD-Audio Virtual Channel Type HD-Audio Virtual Channel Type Selectiton.

0:VC0(Default), 1:VC1. 0: VC0, 1: VC1

Definition at line 457 of file FspsUpd.h.

6.5.2.82 UINT8 FSP_S_CONFIG::HdaVerbTableEntryNum

Offset 0x0033 - SC HDA Verb Table Entry Number Number of Entries in Verb Table.

0(Default).

Definition at line 160 of file FspsUpd.h.

6.5.2.83 UINT32 FSP_S_CONFIG::HdaVerbTablePtr

Offset 0x0034 - SC HDA Verb Table Pointer Pointer to Array of pointers to Verb Table.

0x00000000(Default).

Definition at line 165 of file FspsUpd.h.

6.5.2.84 UINT8 FSP_S_CONFIG::Hmt

Offset 0x009D - HD-Audio Host Memory Transfers Enable/Disable HD-Audio Host Memory Transfers.

0:VC0(Default), 1:VC2. 0: VC0, 1: VC2

Definition at line 525 of file FspsUpd.h.

6.5.2.85 UINT8 FSP_S_CONFIG::Hpet

Offset 0x00A9 - Enable High Precision Timer Enable/Disable Hpet.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 566 of file FspsUpd.h.

6.5.2.86 UINT8 FSP_S_CONFIG::HpetBdfValid

Offset 0x00AA - Hpet Valid BDF Value Enable/Disable Hpet Valid BDF Value.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 572 of file FspsUpd.h.

6.5.2.87 UINT8 FSP_S_CONFIG::HpetBusNumber

Offset 0x00AB - Bus Number of Hpet Completer ID of Bus Number of Hpet.

Default = 0xFA(Default).

Definition at line 577 of file FspsUpd.h.

6.5.2.88 UINT8 FSP_S_CONFIG::HpetDeviceNumber

Offset 0x00AC - Device Number of Hpet Completer ID of Device Number of Hpet.

0x1F(Default).

Definition at line 582 of file FspsUpd.h.

6.5.2.89 UINT8 FSP_S_CONFIG::HpetFunctionNumber

Offset 0x00AD - Function Number of Hpet Completer ID of Function Number of Hpet.

0x00(Default).

Definition at line 587 of file FspsUpd.h.

6.5.2.90 UINT8 FSP_S_CONFIG::HsicSupportEnable

Offset 0x0259 - Enable XHCI HSIC Support Enable/Disable USB HSIC1.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 1222 of file FspsUpd.h.

6.5.2.91 UINT8 FSP_S_CONFIG::Hsuart0Enable

Offset 0x00D3 - UART Device 0 Enable/Disable UART Device 0.

0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode

Definition at line 733 of file FspsUpd.h.

6.5.2.92 UINT8 FSP_S_CONFIG::Hsuart1Enable

Offset 0x00D4 - UART Device 1 Enable/Disable UART Device 1.

0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode

Definition at line 739 of file FspsUpd.h.

6.5.2.93 UINT8 FSP_S_CONFIG::Hsuart2Enable

Offset 0x00D5 - UART Device 2 Enable/Disable UART Device 2.

0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode

Definition at line 745 of file FspsUpd.h.

6.5.2.94 UINT8 FSP_S_CONFIG::Hsuart3Enable

Offset 0x00D6 - UART Device 3 Enable/Disable UART Device 3.

0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode

Definition at line 751 of file FspsUpd.h.

6.5.2.95 UINT8 FSP_S_CONFIG::HsuartClkGateCfg[4]

Offset 0x00C4 - PSS HSUART Clock Gating Configuration Enable/Disable LPSS HSUART Clock Gating.
0:Disable, 1:Enable(Default).
Definition at line 674 of file FspsUpd.h.

6.5.2.96 UINT8 FSP_S_CONFIG::I2c0Enable

Offset 0x00CB - I2C Device 0 Enable/Disable I2C Device 0.
0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode
Definition at line 685 of file FspsUpd.h.

6.5.2.97 UINT8 FSP_S_CONFIG::I2c1Enable

Offset 0x00CC - I2C Device 1 Enable/Disable I2C Device 1.
0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode
Definition at line 691 of file FspsUpd.h.

6.5.2.98 UINT8 FSP_S_CONFIG::I2c2Enable

Offset 0x00CD - I2C Device 2 Enable/Disable I2C Device 2.
0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode
Definition at line 697 of file FspsUpd.h.

6.5.2.99 UINT8 FSP_S_CONFIG::I2c3Enable

Offset 0x00CE - I2C Device 3 Enable/Disable I2C Device 3.
0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode
Definition at line 703 of file FspsUpd.h.

6.5.2.100 UINT8 FSP_S_CONFIG::I2c4Enable

Offset 0x00CF - I2C Device 4 Enable/Disable I2C Device 4.
0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode
Definition at line 709 of file FspsUpd.h.

6.5.2.101 UINT8 FSP_S_CONFIG::I2c5Enable

Offset 0x00D0 - I2C Device 5 Enable/Disable I2C Device 5.
0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode
Definition at line 715 of file FspsUpd.h.

6.5.2.102 UINT8 FSP_S_CONFIG::I2c6Enable

Offset 0x00D1 - I2C Device 6 Enable/Disable I2C Device 6.

0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode

Definition at line 721 of file FspsUpd.h.

6.5.2.103 UINT8 FSP_S_CONFIG::I2c7Enable

Offset 0x00D2 - I2C Device 7 Enable/Disable I2C Device 7.

0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode

Definition at line 727 of file FspsUpd.h.

6.5.2.104 UINT8 FSP_S_CONFIG::I2cClkGateCfg[8]

Offset 0x00BC - LPSS I2C Clock Gating Configuration Enable/Disable LPSS I2C Clock Gating.

0:Disable, 1:Enable(Default).

Definition at line 669 of file FspsUpd.h.

6.5.2.105 UINT8 FSP_S_CONFIG::InitS3Cpu

Offset 0x0332 - Init CPU during S3 resume 0: Do not initialize CPU during S3 resume.

1: Initialize CPU during S3 resume. $EN_DIS

Definition at line 1513 of file FspsUpd.h.

6.5.2.106 UINT8 FSP_S_CONFIG::IoApicBdfValid

Offset 0x00AE - IoApic Valid BDF Value Enable/Disable IoApic Valid BDF Value.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 593 of file FspsUpd.h.

6.5.2.107 UINT8 FSP_S_CONFIG::IoApicBusNumber

Offset 0x00AF - Bus Number of IoApic Completer ID of Bus Number of IoApic.

0xFA(Default).

Definition at line 598 of file FspsUpd.h.

6.5.2.108 UINT8 FSP_S_CONFIG::IoApicDeviceNumber

Offset 0x00B0 - Device Number of IoApic Completer ID of Device Number of IoApic.

0x0F(Default).

Definition at line 603 of file FspsUpd.h.

6.5.2.109 UINT8 FSP_S_CONFIG::IoApicEntry24_119

Offset 0x00B2 - IOAPIC Entry 24-119 Enable/Disable IOAPIC Entry 24-119.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 614 of file FspsUpd.h.

6.5.2.110 UINT8 FSP_S_CONFIG::IoApicFunctionNumber

Offset 0x00B1 - Function Number of IoApic Completer ID of Function Number of IoApic.

0x00(Default).

Definition at line 608 of file FspsUpd.h.

6.5.2.111 UINT8 FSP_S_CONFIG::IoApicId

Offset 0x00B3 - IO APIC ID This member determines IOAPIC ID.

0x01(Default).

Definition at line 619 of file FspsUpd.h.

6.5.2.112 UINT8 FSP_S_CONFIG::IoApicRangeSelect

Offset 0x00B4 - IoApic Range Define address bits 19:12 for the IOxAPIC range.

0x00(Default).

Definition at line 624 of file FspsUpd.h.

6.5.2.113 UINT32 FSP_S_CONFIG::IPC[4]

Offset 0x0320 - IRQ Interrupt Polarity Control Set IRQ Interrupt Polarity Control to ITSS.IPC[0]∼IPC[3].

0:Active High, 1:Active Low

Definition at line 1502 of file FspsUpd.h.

6.5.2.114 UINT8 FSP_S_CONFIG::IpuAcpiMode

Offset 0x003A - IMGU ACPI mode selection 0:Auto, 1:IGFX Child device(Default), 2:ACPI device.

0:Disable, 1:IGFX Child device, 2:ACPI device

Definition at line 183 of file FspsUpd.h.

6.5.2.115 UINT8 FSP_S_CONFIG::IpuEn

Offset 0x0039 - IPU Enable/Disable Enable/Disable IPU Device.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 177 of file FspsUpd.h.

6.5.2.116 UINT8 FSP_S_CONFIG::IshEnable

Offset 0x00B5 - ISH Controller Enable/Disable ISH Controller.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 630 of file FspsUpd.h.

6.5.2.117 UINT8 FSP_S_CONFIG::LockDownGlobalSmi

Offset 0x026B - SMI Lock bit Enable/Disable SMI_LOCK bit to prevent writes to the Global SMI Enable bit.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 1258 of file FspsUpd.h.

6.5.2.118 UINT32 FSP_S_CONFIG::LogoPtr

Offset 0x0058 - BMP Logo Data Pointer BMP logo data pointer to a BMP format buffer.

0x00000000(Default).

Definition at line 305 of file FspsUpd.h.

6.5.2.119 UINT32 FSP_S_CONFIG::LogoSize

Offset 0x0054 - BMP Logo Data Size BMP logo data buffer size.

0x00000000(Default).

Definition at line 300 of file FspsUpd.h.

6.5.2.120 UINT8 FSP_S_CONFIG::LPSS_S0ixEnable

Offset 0x00BA - LPSS IOSF PMCTL S0ix Enable Enable/Disable LPSS IOSF Bridge PMCTL Register S0ix Bits.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 660 of file FspsUpd.h.

6.5.2.121 UINT8 FSP_S_CONFIG::MaxCoreCState

Offset 0x002F - Max Core C-State Max Core C-State.

0:Unlimited, 1:C1, 2:C3, 3:C6, 4:C7, 5:C8, 6:C9, 7:C10, 8:CCx(Default).

Definition at line 137 of file FspsUpd.h.

6.5.2.122 UINT8 FSP_S_CONFIG::Mmt

Offset 0x009C - HD-Audio CSME Memory Transfers Enable/Disable HD-Audio CSME Memory Transfers.

0:VC0(Default), 1:VC2. 0: VC0, 1: VC2

Definition at line 519 of file FspsUpd.h.

6.5.2.123 UINT8 FSP_S_CONFIG::MonitorMwaitEnable

Offset 0x031E - Monitor Mwait Enable Enable/Disable Monitor Mwait.

For Windows∗ OS, this should be Enabled. For Linux based OS, this should be Disabled. 0:Disable, 1:Enable(↩ Default). $EN_DIS

Definition at line 1490 of file FspsUpd.h.

6.5.2.124 UINT8 FSP_S_CONFIG::NoFatalErrorReport[6]

Offset 0x0140 - NFER PCI Express Device Non-Fatal Error Reporting Enable/Disable.

0:Disable(Default), 1:Enable.

Definition at line 899 of file FspsUpd.h.

6.5.2.125 UINT16 FSP_S_CONFIG::NumRsvdSmbusAddresses

Offset 0x01C2 - SMBus Table Elements The number of elements in the Reserved SMBus Address Table.

0x0080(Default).

Definition at line 1171 of file FspsUpd.h.

6.5.2.126 UINT8 FSP_S_CONFIG::OsDbgEnable

Offset 0x00DA - OS Debug Feature Enable/Disable OS Debug Feature.

0:Disable(Default), 1: Enable. $EN_DIS

Definition at line 775 of file FspsUpd.h.

6.5.2.127 UINT8 FSP_S_CONFIG::OsSelection

Offset 0x0370 - OS Selection Windows or Android or Linux OS selection to be used by HDA, USB Common, PWM and PEI Graphics modules.

Windows (default), Android, Linux 0x0:Windows, 0x1:Android, 0x3:Linux

Definition at line 1566 of file FspsUpd.h.

6.5.2.128 UINT8 FSP_S_CONFIG::P2sbSecEn

Offset 0x0373 - P2SB Security Option Enable/Disable Enable/Disable P2SB Security Option.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 1585 of file FspsUpd.h.

6.5.2.129 UINT8 FSP_S_CONFIG::P2sbUnhide

Offset 0x0038 - Enable/Disable P2SB device hidden.

Enable/Disable P2SB device hidden. 0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 171 of file FspsUpd.h.

6.5.2.130 UINT8 FSP_S_CONFIG::PavpEnable

Offset 0x0060 - PAVP Enable Enable/Disable Protected Audio Visual Path (PAVP).

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 316 of file FspsUpd.h.

6.5.2.131 UINT8 FSP_S_CONFIG::PavpLock

Offset 0x0044 - Enable PavpLock Enable/disable PavpLock.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 205 of file FspsUpd.h.

6.5.2.132 UINT8 FSP_S_CONFIG::PavpPr3

Offset 0x0061 - PAVP PR3 Enable/Disable PAVP PR3 0:Disable, 1:Enable(Default).

$EN_DIS

Definition at line 322 of file FspsUpd.h.

6.5.2.133 UINT8 FSP_S_CONFIG::PciClockRun

Offset 0x018F - PCI Clock Run This member describes whether or not the PCI ClockRun feature of SC should be enabled.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 973 of file FspsUpd.h.

6.5.2.134 UINT8 FSP_S_CONFIG::Pcie8xhDecodePortIndex

Offset 0x00E2 - PCIE 8xh Decode Port Index PCIE 8xh Decode Port Index.

0x00(Default).

Definition at line 803 of file FspsUpd.h.

6.5.2.135 UINT8 FSP_S_CONFIG::PcieAspmSwSmiNumber

Offset 0x00E4 - PCIE SWSMI Number This member describes the SwSmi value for override PCIe ASPM table.

0xAA(Default).

Definition at line 814 of file FspsUpd.h.

6.5.2.136 UINT8 FSP_S_CONFIG::PcieClockGatingDisabled

Offset 0x00E0 - Enable PCIE Clock Gating Enable/disable PCIE Clock Gating.

0:Enable, 1:Disable(Default). 0:Enable, 1:Disable

Definition at line 792 of file FspsUpd.h.

6.5.2.137 UINT8 FSP_S_CONFIG::PcieRootPort8xhDecode

Offset 0x00E1 - Enable PCIE Root Port 8xh Decode Enable/disable PCIE Root Port 8xh Decode.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 798 of file FspsUpd.h.

6.5.2.138 UINT8 FSP_S_CONFIG::PcieRootPortEn[6]

Offset 0x00E6 - PCI Express Root Port Control the PCI Express Root Port .

0:Disable, 1:Enable(Default).

Definition at line 823 of file FspsUpd.h.

6.5.2.139 UINT8 FSP_S_CONFIG::PcieRootPortPeerMemoryWriteEnable

Offset 0x00E3 - Enable PCIE Root Port Peer Memory Write Enable/disable PCIE root port peer memory write.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 809 of file FspsUpd.h.

6.5.2.140 UINT8 FSP_S_CONFIG::PcieRpAcsEnabled[6]

Offset 0x0110 - ACS Enable/Disable Access Control Services Extended Capability.

0:Disable, 1:Enable(Default).

Definition at line 858 of file FspsUpd.h.

6.5.2.141 UINT8 FSP_S_CONFIG::PcieRpAspm[6]

Offset 0x0176 - ASPM PCI Express Active State Power Management settings.

0:Disable, 1:L0s, 2:L1, 3:L0sL1, 4:Auto(Default).

Definition at line 945 of file FspsUpd.h.

6.5.2.142 UINT8 FSP_S_CONFIG::PcieRpClkReqDetect[6]

Offset 0x0122 - CLKREQ# Detection Enable/Disable CLKREQ# Detection Probe.

0: Disable(Default), 1: Enable.

Definition at line 874 of file FspsUpd.h.

6.5.2.143 UINT8 FSP_S_CONFIG::PcieRpClkReqNumber[6]

Offset 0x011C - Configure CLKREQ Number Configure Root Port CLKREQ Number if CLKREQ is supported.

Default=0x04, 0x05, 0x00, 0x01, 0x02, 0x03.

Definition at line 869 of file FspsUpd.h.

6.5.2.144 UINT8 FSP_S_CONFIG::PcieRpClkReqSupported[6]

Offset 0x0116 - Clock Request Support Enable/Disable CLKREQ# Support.

0:Disable, 1:Enable(Default).

Definition at line 863 of file FspsUpd.h.

6.5.2.145 UINT8 FSP_S_CONFIG::PcieRpCompletionTimeout[6]

Offset 0x016A - CTO Enable/Disable PCI Express Completion Timer TO .

0:Disable(Default), 1:Enable.

Definition at line 934 of file FspsUpd.h.

6.5.2.146 UINT8 FSP_S_CONFIG::PcieRpExtSync[6]

Offset 0x0104 - PCIE Root Port Extended Sync Enable/Disable PCIE Root Port Extended Sync.

0:Disable, 1:Enable(Default).

Definition at line 848 of file FspsUpd.h.

6.5.2.147 UINT8 FSP_S_CONFIG::PcieRpHide[6]

Offset 0x00EC - Hide PCIE Root Port Configuration Space Enable/disable Hide PCIE Root Port Configuration Space.

0:Disable(Default), 1:Enable.

Definition at line 828 of file FspsUpd.h.

6.5.2.148 UINT8 FSP_S_CONFIG::PcieRpHotPlug[6]

Offset 0x00F8 - Hot Plug PCI Express Hot Plug Enable/Disable.

0:Disable, 1:Enable(Default).

Definition at line 838 of file FspsUpd.h.

6.5.2.149 UINT8 FSP_S_CONFIG::PcieRpL1Substates[6]

Offset 0x017C - L1 Substates PCI Express L1 Substates settings.

0:Disable, 1:L1.1, 2:L1.2, 3:L1.1 & L1.2(Default).

Definition at line 950 of file FspsUpd.h.

6.5.2.150 UINT8 FSP_S_CONFIG::PcieRpLtrConfigLock[6]

Offset 0x0188 - PCIE LTR Lock PCIE LTR Configuration Lock.

0:Disable(Default), 1:Enable.

Definition at line 960 of file FspsUpd.h.

6.5.2.151 UINT8 FSP_S_CONFIG::PcieRpLtrEnable[6]

Offset 0x0182 - PCH PCIe LTR PCH PCIE Latency Reporting Enable/Disable.

0:Disable, 1:Enable(Default).

Definition at line 955 of file FspsUpd.h.

6.5.2.152 UINT16 FSP_S_CONFIG::PcieRpLtrMaxNonSnoopLatency[6]

Offset 0x029C - Max Non-Snoop Latency Latency Tolerance Reporting, Max Non-Snoop Latency.

0x0000(Default).

Definition at line 1332 of file FspsUpd.h.

6.5.2.153 UINT16 FSP_S_CONFIG::PcieRpLtrMaxSnoopLatency[6]

Offset 0x0274 - Max Snoop Latency Latency Tolerance Reporting Max Snoop Latency.

0x0000(Default).

Definition at line 1291 of file FspsUpd.h.

6.5.2.154 UINT8 FSP_S_CONFIG::PcieRpNonSnoopLatencyOverrideMode[6]

Offset 0x02A8 - Non Snoop Latency Override Non Snoop Latency Override for PCH PCIE.

Disabled:Disable override.
Manual:Manually enter override values.
Auto: Maintain default BIOS flow. 0:Disable, 1:Enable, 2:Auto(Default).

Definition at line 1340 of file FspsUpd.h.

6.5.2.155 UINT8 FSP_S_CONFIG::PcieRpNonSnoopLatencyOverrideMultiplier[6]

Offset 0x02BC - Non Snoop Latency Multiplier LTR Non Snoop Latency Multiplier of PCH PCIE.

0:1ns, 1:32ns, 2:1024ns(Default), 3:32768ns, 4:1048576ns, 5:33554432ns.

Definition at line 1364 of file FspsUpd.h.

6.5.2.156 UINT16 FSP_S_CONFIG::PcieRpNonSnoopLatencyOverrideValue[6]

Offset 0x02B0 - Non Snoop Latency Value LTR Non Snoop Latency value of PCH PCIE.

0:Minimum, 0x03FF:Maximum, 0x003C(Default).

Definition at line 1358 of file FspsUpd.h.

6.5.2.157 UINT8 FSP_S_CONFIG::PcieRpPmSci[6]

Offset 0x00FE - PCIE PM SCI Enable/Disable PCI Express PME SCI.

0:Disable(Default), 1:Enable.

Definition at line 843 of file FspsUpd.h.

6.5.2.158 UINT8 FSP_S_CONFIG::PcieRpSelectableDeemphasis[6]

Offset 0x0318 - PCIe Selectable De-emphasis When the Link is operating at 5.0 GT/s speed, this bit selects the level of de-emphasis for an Upstream component.

1b:-3.5 dB 0b:-6 dB. 0:Disable, 1:Enable(Default).

Definition at line 1483 of file FspsUpd.h.

6.5.2.159 UINT8 FSP_S_CONFIG::PcieRpSlotImplemented[6]

Offset 0x00F2 - PCIE Root Port Slot Implement Enable/disable PCIE Root Port Slot Implement.

0:Disable, 1:Enable(Default).

Definition at line 833 of file FspsUpd.h.

6.5.2.160 UINT8 FSP_S_CONFIG::PcieRpSlotPowerLimitScale[6]

Offset 0x02C2 - PCIE Root Port Slot Power Limit Scale Specifies scale used for slot power limit value.

0x00(Default).

Definition at line 1369 of file FspsUpd.h.

6.5.2.161 UINT8 FSP_S_CONFIG::PcieRpSlotPowerLimitValue[6]

Offset 0x02C8 - PCIE Root Port Slot Power Limit Value Specifies upper limit on power supplie by slot.

0x00(Default).

Definition at line 1374 of file FspsUpd.h.

6.5.2.162 UINT8 FSP_S_CONFIG::PcieRpSnoopLatencyOverrideMode[6]

Offset 0x0280 - Snoop Latency Override Snoop Latency Override for PCH PCIE.

Disabled:Disable override.
Manual:Manually enter override values.
Auto:Maintain default BIOS flow. 0:Disable, 1:Enable, 2:Auto(Default).

Definition at line 1299 of file FspsUpd.h.

6.5.2.163 UINT8 FSP_S_CONFIG::PcieRpSnoopLatencyOverrideMultiplier[6]

Offset 0x0294 - Snoop Latency Multiplier LTR Snoop Latency Multiplier of PCH PCIE.

0:1ns, 1:32ns, 2:1024ns(Default), 3:32768ns, 4:1048576ns, 5:33554432ns.

Definition at line 1314 of file FspsUpd.h.

6.5.2.164 UINT16 FSP_S_CONFIG::PcieRpSnoopLatencyOverrideValue[6]

Offset 0x0288 - Snoop Latency Value LTR Snoop Latency value of PCH PCIE.

0:Minimum, 0x03FF:Maximum, 0x003C(Default).

Definition at line 1308 of file FspsUpd.h.

6.5.2.165 UINT8 FSP_S_CONFIG::PcieRpSpeed[6]

Offset 0x015E - PCIe Speed Configure PCIe Speed.

0:Auto(Default), 1:Gen1, 2:Gen2, 3:Gen3.

Definition at line 924 of file FspsUpd.h.

6.5.2.166 UINT8 FSP_S_CONFIG::PcieRpTransmitterHalfSwing[6]

Offset 0x010A - Transmitter Half Swing Transmitter Half Swing Enable/Disable.

0:Disable, 1:Enable(Default).

Definition at line 853 of file FspsUpd.h.

6.5.2.167 UINT8 FSP_S_CONFIG::PeiGraphicsPeimInit

Offset 0x0063 - Enable/Disable PeiGraphicsPeimInit Enable/Disable PeiGraphicsPeimInit 0:Disable, 1:Enable(↩ Default).

$EN_DIS

Definition at line 334 of file FspsUpd.h.

6.5.2.168 UINT8 FSP_S_CONFIG::PhysicalSlotNumber[6]

Offset 0x0164 - Physical Slot Number Physical Slot Number for PCIE Root Port.

Default=0x00, 0x01, 0x02, 0x03, 0x04, 0x05.

Definition at line 929 of file FspsUpd.h.

6.5.2.169 UINT8 FSP_S_CONFIG::PkgCStateDemotion

Offset 0x0030 - Package C-State Demotion Enable or Disable Package Cstate Demotion.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 143 of file FspsUpd.h.

6.5.2.170 UINT8 FSP_S_CONFIG::PkgCStateLimit

Offset 0x002C - Max Pkg Cstate Max Pkg Cstate.

0:PkgC0C1, 1:PkgC2, 2:PkgC3(Default), 3:PkgC6, 4:PkgC7, 5:PkgC7s, 6:PkgC8, 7:PkgC9, 8:PkgC10, 9:PkgC↩ Max, 254:PkgCpuDefault, 255:PkgAuto.

Definition at line 119 of file FspsUpd.h.

6.5.2.171 UINT8 FSP_S_CONFIG::PkgCStateUnDemotion

Offset 0x0031 - Package C-State Un-demotion Enable or Disable Package Cstate UnDemotion.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 149 of file FspsUpd.h.

6.5.2.172 UINT8 FSP_S_CONFIG::Pme

Offset 0x008F - Azalia wake-on-ring Enable/disable Azalia wake-on-ring.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 438 of file FspsUpd.h.

6.5.2.173 UINT8 FSP_S_CONFIG::PmeB0S5Dis

Offset 0x018E - PME_B0_S5 Disable bit PME_B0_S5_DIS bit in the General PM Configuration B (GEN_PMCO↩ N_B) register.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 966 of file FspsUpd.h.

6.5.2.174 UINT8 FSP_S_CONFIG::PmeInterrupt[6]

Offset 0x012E - PME Interrupt Enable/Disable PME Interrupt.

0: Disable(Default), 1: Enable.

Definition at line 884 of file FspsUpd.h.

6.5.2.175 UINT8 FSP_S_CONFIG::PmLock

Offset 0x0048 - Enable PmLock Enable/disable PmLock.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 229 of file FspsUpd.h.

6.5.2.176 UINT8 FSP_S_CONFIG::PmSupport

Offset 0x0052 - GT PM Support Enable/Disable GT power management support.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 289 of file FspsUpd.h.

6.5.2.177 UINT8 FSP_S_CONFIG::PortUs20bOverCurrentPin[8]

Offset 0x0250 - USB20 Over Current Pin Over Current Pin number of USB 2.0 Port.

0x00(Default).

Definition at line 1210 of file FspsUpd.h.

6.5.2.178 UINT8 FSP_S_CONFIG::PortUs30bOverCurrentPin[6]

Offset 0x0260 - USB20 Over Current Pin Over Current Pin number of USB 3.0 Port.

0x01(Default).

Definition at line 1233 of file FspsUpd.h.

6.5.2.179 UINT8 FSP_S_CONFIG::PortUsb20Enable[8]

Offset 0x0248 - Enable USB2 ports Enable/disable per USB2 ports.

One byte for each port, byte0 for port0, byte1 for port1, and so on. 0x01(Default).

Definition at line 1205 of file FspsUpd.h.

6.5.2.180 UINT8 FSP_S_CONFIG::PortUsb20HsNpreDrvSel[8]

Offset 0x0368 - Delay/skew's strength control for HS driver Delay/skew's strength control for HS driver.

Value of register USB2_PER_PORT_2_PPX [1:0]

Definition at line 1559 of file FspsUpd.h.

6.5.2.181 UINT8 FSP_S_CONFIG::PortUsb20HsSkewSel[8]

Offset 0x0350 - Select the skew direction for HS transition Select the skew direction for HS transition.

Value of register USB2_PER_PORT_2_PPX [25]

Definition at line 1544 of file FspsUpd.h.

6.5.2.182 UINT8 FSP_S_CONFIG::PortUsb20IUsbTxEmphasisEn[8]

Offset 0x0358 - Per Port HS Transmitter Emphasis Per Port HS Transmitter Emphasis.

Value of register USB2_PER_PORT_2_PPX [24:23]

Definition at line 1549 of file FspsUpd.h.

6.5.2.183 UINT8 FSP_S_CONFIG::PortUsb20PerPortPeTxiSet[8]

Offset 0x0340 - PerPort HS Pre-emphasis Bias PerPort HS Pre-emphasis Bias.

Value of register USB2_PER_PORT_PPX [13:11]

Definition at line 1534 of file FspsUpd.h.

6.5.2.184 UINT8 FSP_S_CONFIG::PortUsb20PerPortRXISet[8]

Offset 0x0360 - PerPort HS Receiver Bias PerPort HS Receiver Bias.

Value of register USB2_PER_PORT_PPX [19:17]

Definition at line 1554 of file FspsUpd.h.

6.5.2.185 UINT8 FSP_S_CONFIG::PortUsb20PerPortTxiSet[8]

Offset 0x0348 - PerPort HS Transmitter Bias PerPort HS Transmitter Bias.

Value of register USB2_PER_PORT_PPX [10:8]

Definition at line 1539 of file FspsUpd.h.

6.5.2.186 UINT8 FSP_S_CONFIG::PortUsb20PerPortTxPeHalf[8]

Offset 0x0338 - PerPort Half Bit Pre-emphasis PerPort Half Bit Pre-emphasis.

Value of register USB2_PER_PORT_PPX [14]

Definition at line 1529 of file FspsUpd.h.

6.5.2.187 UINT8 FSP_S_CONFIG::PortUsb30Enable[6]

Offset 0x025A - Enable USB3 ports Enable/disable per USB3 ports.

One byte for each port, byte0 for port0, byte1 for port1, and so on. 0x01(Default).

Definition at line 1228 of file FspsUpd.h.

6.5.2.188 UINT8 FSP_S_CONFIG::PowerButterDebounceMode

Offset 0x02CF - Power Button Debounce Mode Enable interrupt when PWRBTN# is asserted.

0:Disabled, 1:Enabled(default) $EN_DIS

Definition at line 1387 of file FspsUpd.h.

6.5.2.189 UINT8 FSP_S_CONFIG::PowerGating

Offset 0x004D - Enable PowerGating Enable/disable PowerGating.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 259 of file FspsUpd.h.

6.5.2.190 UINT8 FSP_S_CONFIG::ProcTraceEnable

Offset 0x0026 - Enable Processor Trace Enable or Disable Processor Trace feature.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 84 of file FspsUpd.h.

6.5.2.191 UINT8 FSP_S_CONFIG::ProcTraceMemSize

Offset 0x0025 - Memory region allocation for Processor Trace Memory region allocation for Processor Trace, allowed range is from 4K (0x0) to 128MB (0xF); **0xFF: Disable.**

0xFF:Disable(Default)

Definition at line 78 of file FspsUpd.h.

6.5.2.192 UINT16 FSP_S_CONFIG::ProtectedRangeBase[5]

Offset 0x0078 - Protected Range Base The base address of the upper limit of protection.

0x0000(Default).

Definition at line 354 of file FspsUpd.h.

6.5.2.193 UINT8 FSP_S_CONFIG::PtmEnable[6]

Offset 0x0170 - PTM Support Enable/Disable PTM Support.

0:Disable(Default), 1:Enable.

Definition at line 939 of file FspsUpd.h.

6.5.2.194 UINT8 FSP_S_CONFIG::PWMEnabled

Offset 0x0372 - PWM Enabled PWM Device Enabling.

Windows needs this to be disabled, while Android needs this to be enabled. 0x0:Disabled (default), 0x1:Enabled $EN_DIS

Definition at line 1579 of file FspsUpd.h.

6.5.2.195 UINT8 FSP_S_CONFIG::PwrBtnOverridePeriod

Offset 0x02AF - Power Button Override Period specifies how long will PMC wait before initiating a global reset.

000b-4s(default), 001b-6s, 010b-8s, 011b-10s, 100b-12s, 101b-14s.) 0x0:4s, 0x1:6s, 0x2:8s, 0x3:10s, 0x4:12s, 0x5:14s

Definition at line 1353 of file FspsUpd.h.

6.5.2.196 UINT8 FSP_S_CONFIG::ReadProtectionEnable[5]

Offset 0x0069 - Read Protection Support Enable/disable Read Protection.

0:Disable, 1:Enable(Default).

Definition at line 344 of file FspsUpd.h.

6.5.2.197 UINT8 FSP_S_CONFIG::ResetSelect

Offset 0x01B5 - ResetSelect ResetSelect.

0x6:warm reset(Default), 0xE:cold reset.

Definition at line 1096 of file FspsUpd.h.

6.5.2.198 UINT16 FSP_S_CONFIG::ResetWaitTimer

Offset 0x026C - HDAudio Delay Timer The delay timer after Azalia reset.

0x012C(Default).

Definition at line 1263 of file FspsUpd.h.

6.5.2.199 UINT8 FSP_S_CONFIG::RsvdSmbusAddressTable[128]

Offset 0x01C4 - Reserved SMBus Address Table Array of addresses reserved for non-ARP-capable SMBus devices.

0x00(Default).

Definition at line 1176 of file FspsUpd.h.

6.5.2.200 UINT8 FSP_S_CONFIG::RtcLock

Offset 0x026E - RTC Lock Bits Enable/Disable RTC Lock Bits.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 1269 of file FspsUpd.h.

6.5.2.201 UINT8 FSP_S_CONFIG::SaIpuEnable

Offset 0x0051 - Enable SaIpuEnable Enable/disable SaIpuEnable.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 283 of file FspsUpd.h.

6.5.2.202 UINT8 FSP_S_CONFIG::SataMode

Offset 0x0192 - SATA Mode Selection Determines how SATA controller(s) operate.

0:AHCI(Default), 1:RAID. 0:AHCI, 1:RAID

Definition at line 992 of file FspsUpd.h.

6.5.2.203 UINT8 FSP_S_CONFIG::SataPortsDevSlp[2]

Offset 0x019A - SATA Port DevSlp Enable/Disable SATA Port DevSlp.

Board rework for LP needed before enable. 0:Disable(Default), 1:Enable.

Definition at line 1030 of file FspsUpd.h.

6.5.2.204 UINT16 FSP_S_CONFIG::SataPortsDitoVal[2]

Offset 0x01AC - DITO Value DEVSLP Idle Timeout Value.

0:Minimum, 0x03FF:Maximum, 0x0271(Default).

Definition at line 1076 of file FspsUpd.h.

6.5.2.205 UINT8 FSP_S_CONFIG::SataPortsDmVal[2]

Offset 0x01A8 - DM Value DM Value.

0:Minimum, 0x0F:Maximum(Default).

Definition at line 1067 of file FspsUpd.h.

6.5.2.206 UINT8 FSP_S_CONFIG::SataPortsEnable[2]

Offset 0x0198 - SATA Port Enable or Disable SATA Port.

0:Disable, 1:Enable(Default).

Definition at line 1025 of file FspsUpd.h.

6.5.2.207 UINT8 FSP_S_CONFIG::SataPortsEnableDitoConfig[2]

Offset 0x01A6 - DITO Configuration Enable/Disable DITO Configuration.

0:Disable(Default), 1:Enable.

Definition at line 1062 of file FspsUpd.h.

6.5.2.208 UINT8 FSP_S_CONFIG::SataPortsExternal[2]

Offset 0x01A0 - External SATA Ports Enable/Disable External SATA Ports.

0:Disable(Default), 1:Enable.

Definition at line 1046 of file FspsUpd.h.

6.5.2.209 UINT8 FSP_S_CONFIG::SataPortsHotPlug[2]

Offset 0x019C - SATA Port HotPlug Enable/Disable SATA Port Hotplug .

0:Disable(Default), 1:Enable.

Definition at line 1035 of file FspsUpd.h.

6.5.2.210 UINT8 FSP_S_CONFIG::SataPortsInterlockSw[2]

Offset 0x019E - Mechanical Presence Switch Controls reporting if this port has an Mechanical Presence Switch.

Note:Requires hardware support. 0:Disable, 1:Enable(Default).

Definition at line 1041 of file FspsUpd.h.

6.5.2.211 UINT8 FSP_S_CONFIG::SataPortsSolidStateDrive[2]

Offset 0x01A4 - SATA Solid State Identify the SATA port is connected to Solid State Drive or Hard Disk Drive.

0:Hard Disk Drive(Default), 1:Solid State Drive.

Definition at line 1057 of file FspsUpd.h.

6.5.2.212 UINT8 FSP_S_CONFIG::SataPortsSpinUp[2]

Offset 0x01A2 - Spin Up Device Enable/Disable device spin up at boot on selected Sata Ports.

0:Disable(Default), 1:Enable.

Definition at line 1051 of file FspsUpd.h.

6.5.2.213 UINT8 FSP_S_CONFIG::SataPwrOptEnable

Offset 0x0194 - SATA Power Optimization Enable SATA Power Optimizer on SC side.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 1004 of file FspsUpd.h.

6.5.2.214 UINT8 FSP_S_CONFIG::SataSalpSupport

Offset 0x0193 - Aggressive LPM Support Enable PCH to aggressively enter link power state.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 998 of file FspsUpd.h.

6.5.2.215 UINT8 FSP_S_CONFIG::SataTestMode

Offset 0x026F - SATA Test Mode Selection Enable/Disable SATA Test Mode.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 1275 of file FspsUpd.h.

6.5.2.216 UINT8 FSP_S_CONFIG::SdcardEnabled

Offset 0x01B6 - SD Card Support (D27:F0) Enable/Disable SD Card Support.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 1102 of file FspsUpd.h.

6.5.2.217 UINT32 FSP_S_CONFIG::SdcardRxCmdDataCntl1

Offset 0x02F0 - SDCARD_RX_CMD_DATA_DLL_CNTL1 SDCARD_RX_CMD_DATA_DLL_CNTL1.

0x73A3637(Default).

Definition at line 1432 of file FspsUpd.h.

6.5.2.218 UINT32 FSP_S_CONFIG::SdcardRxCmdDataCntl2

Offset 0x02F8 - SDCARD_RX_CMD_DATA_DLL_CNTL2 SDCARD_RX_CMD_DATA_DLL_CNTL2.

0x10000(Default).

Definition at line 1442 of file FspsUpd.h.

6.5.2.219 UINT32 FSP_S_CONFIG::SdcardRxStrobeCntl

Offset 0x02F4 - SDCARD_RX_STROBE_DLL_CNTL SDCARD_RX_STROBE_DLL_CNTL.

0x0(Default).

Definition at line 1437 of file FspsUpd.h.

6.5.2.220 UINT32 FSP_S_CONFIG::SdcardTxCmdCntl

Offset 0x02E4 - SDCARD_TX_CMD_DLL_CNTL SDCARD_TX_CMD_DLL_CNTL.

0x505(Default).

Definition at line 1417 of file FspsUpd.h.

6.5.2.221 UINT32 FSP_S_CONFIG::SdcardTxDataCntl1

Offset 0x02E8 - SDCARD_TX_DATA_DLL_CNTL1 SDCARD_TX_DATA_DLL_CNTL1.

0xA13(Default).

Definition at line 1422 of file FspsUpd.h.

6.5.2.222 UINT32 FSP_S_CONFIG::SdcardTxDataCntl2

Offset 0x02EC - SDCARD_TX_DATA_DLL_CNTL2 SDCARD_TX_DATA_DLL_CNTL2.

0x24242828(Default).

Definition at line 1427 of file FspsUpd.h.

6.5.2.223 UINT8 FSP_S_CONFIG::SdioEnabled

Offset 0x01BA - SDIO Support (D30:F0) Enable/Disable SDIO Support.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 1126 of file FspsUpd.h.

6.5.2.224 UINT32 FSP_S_CONFIG::SdioRxCmdDataCntl1

Offset 0x02DC - SDIO_RX_CMD_DATA_DLL_CNTL1 SDIO_RX_CMD_DATA_DLL_CNTL1.

0x16161616(Default).

Definition at line 1407 of file FspsUpd.h.

6.5.2.225 UINT32 FSP_S_CONFIG::SdioRxCmdDataCntl2

Offset 0x02E0 - SDIO_RX_CMD_DATA_DLL_CNTL2 SDIO_RX_CMD_DATA_DLL_CNTL2.

0x10000(Default).

Definition at line 1412 of file FspsUpd.h.

6.5.2.226 UINT32 FSP_S_CONFIG::SdioTxCmdCntl

Offset 0x02D0 - SDIO_TX_CMD_DLL_CNTL SDIO_TX_CMD_DLL_CNTL.

0x505(Default).

Definition at line 1392 of file FspsUpd.h.

6.5.2.227 UINT32 FSP_S_CONFIG::SdioTxDataCntl1

Offset 0x02D4 - SDIO_TX_DATA_DLL_CNTL1 SDIO_TX_DATA_DLL_CNTL1.

0xE(Default).

Definition at line 1397 of file FspsUpd.h.

6.5.2.228 UINT32 FSP_S_CONFIG::SdioTxDataCntl2

Offset 0x02D8 - SDIO_TX_DATA_DLL_CNTL2 SDIO_TX_DATA_DLL_CNTL2.

0x22272828(Default).

Definition at line 1402 of file FspsUpd.h.

6.5.2.229 UINT8 FSP_S_CONFIG::SirqEnable

Offset 0x01BC - Serial IRQ Enable/Disable Serial IRQ.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 1138 of file FspsUpd.h.

6.5.2.230 UINT8 FSP_S_CONFIG::SirqMode

Offset 0x01BD - Serial IRQ Mode Serial IRQ Mode Selection.

0:Quiet mode(Default), 1:Continuous mode. $EN_DIS

Definition at line 1144 of file FspsUpd.h.

6.5.2.231 UINT8 FSP_S_CONFIG::SkipMpInit

Offset 0x029A - Skip Multi-Processor Initialization When this is skipped, boot loader must initialize processors before SilicionInit API.

0: Initialize(Default), **1: Skip $EN_DIS**

Definition at line 1321 of file FspsUpd.h.

6.5.2.232 UINT8 FSP_S_CONFIG::SkipPunitInit

Offset 0x0333 - Skip P-unit Initialization When this is skipped, boot loader must initialize P-unit before SilicionInit API.

0: Initialize(Default), 1: Skip $EN_DIS

Definition at line 1520 of file FspsUpd.h.

6.5.2.233 UINT8 FSP_S_CONFIG::SmbusEnable

Offset 0x01BF - Enable SMBus Enable/disable SMBus controller.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 1156 of file FspsUpd.h.

6.5.2.234 UINT8 FSP_S_CONFIG::SpeedLimit

Offset 0x0196 - SATA Speed Limit SATA Speed Limit.

0h:ScSataSpeed(Default), 1h:1.5Gb/s(Gen 1), 2h:3Gb/s(Gen 2), 3h:6Gb/s(Gen 3). 0:Default, 1: 1.5 Gb/s (Gen 1), 2: 3 Gb/s(Gen 2), 3: 6 Gb/s (Gen 1)

Definition at line 1016 of file FspsUpd.h.

6.5.2.235 UINT8 FSP_S_CONFIG::Spi0Enable

Offset 0x00D7 - SPI UART Device 0 Enable/Disable SPI Device 0.

0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode

Definition at line 757 of file FspsUpd.h.

6.5.2.236 UINT8 FSP_S_CONFIG::Spi1Enable

Offset 0x00D8 - SPI UART Device 1 Enable/Disable SPI Device 1.

0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode

Definition at line 763 of file FspsUpd.h.

6.5.2.237 UINT8 FSP_S_CONFIG::Spi2Enable

Offset 0x00D9 - SPI UART Device 2 Enable/Disable SPI Device 2.

0:Disabled, 1:PCI Mode(Default), 2:ACPI Mode. 0: Disabled, 1: PCI Mode, 2: ACPI Mode

Definition at line 769 of file FspsUpd.h.

6.5.2.238 UINT8 FSP_S_CONFIG::SpiClkGateCfg[3]

Offset 0x00C8 - LPSS SPI Clock Gating Configuration Enable/Disable LPSS SPI Clock Gating.

0:Disable, 1:Enable(Default).

Definition at line 679 of file FspsUpd.h.

6.5.2.239 UINT8 FSP_S_CONFIG::SpiEiss

Offset 0x00B8 - SPI EISS Status Enable/Disable InSMM.STS (EISS) in SPI.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 649 of file FspsUpd.h.

6.5.2.240 UINT8 FSP_S_CONFIG::SsicPortEnable[2]

Offset 0x0266 - Enable XHCI SSIC Support Enable/disable XHCI SSIC ports.

One byte for each port, byte0 for port0, byte1 for port1. 0x00(Default).

Definition at line 1239 of file FspsUpd.h.

6.5.2.241 UINT8 FSP_S_CONFIG::SsicRate[2]

Offset 0x0270 - XHCI SSIC RATE Set XHCI SSIC1 Rate to A Series or B Series.

1:A Series(Default), 2:B Series.

Definition at line 1280 of file FspsUpd.h.

6.5.2.242 UINT8 FSP_S_CONFIG::StartFramePulse

Offset 0x01BE - Start Frame Pulse Width Start Frame Pulse Width Value.

0:ScSfpw4Clk(Default), 1: ScSfpw6Clk, 2:ScSfpw8Clk. 0:ScSfpw4Clk, 1:ScSfpw6Clk, 2:ScSfpw8Clk

Definition at line 1150 of file FspsUpd.h.

6.5.2.243 UINT16 FSP_S_CONFIG::SubSystemId

Offset 0x01B2 - Subsystem ID Subsystem ID.

0x7270(Default).

Definition at line 1086 of file FspsUpd.h.

6.5.2.244 UINT16 FSP_S_CONFIG::SubSystemVendorId

Offset 0x01B0 - Subsystem Vendor ID Subsystem Vendor ID.

0x8086(Default).

Definition at line 1081 of file FspsUpd.h.

6.5.2.245 UINT8 FSP_S_CONFIG::SystemErrorOnCorrectableError[6]

Offset 0x0158 - SECE Root PCI Express System Error on Correctable Error Enable/Disable.

0:Disable(Default), 1:Enable.

Definition at line 919 of file FspsUpd.h.

6.5.2.246 UINT8 FSP_S_CONFIG::SystemErrorOnFatalError[6]

Offset 0x014C - SEFE Root PCI Express System Error on Fatal Error Enable/Disable.

0:Disable(Default), 1:Enable.

Definition at line 909 of file FspsUpd.h.

6.5.2.247 UINT8 FSP_S_CONFIG::SystemErrorOnNonFatalError[6]

Offset 0x0152 - SENFE Root PCI Express System Error on Non-Fatal Error Enable/Disable.

0:Disable(Default), 1:Enable.

Definition at line 914 of file FspsUpd.h.

6.5.2.248 UINT8 FSP_S_CONFIG::TcoTimerHaltLock

Offset 0x02AE - Halt and Lock TCO Timer Halt and Lock the TCO Timer (Watchdog).

0:No, 1:Yes (default)

Definition at line 1346 of file FspsUpd.h.

6.5.2.249 UINT8 FSP_S_CONFIG::Timer8254ClkSetting

Offset 0x0190 - Enable/Disable Timer 8254 Clock Setting Enable/Disable Timer 8254 Clock.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 979 of file FspsUpd.h.

6.5.2.250 UINT8 FSP_S_CONFIG::TurboMode

Offset 0x0032 - Turbo Mode Enable or Disable long duration Turbo Mode.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 155 of file FspsUpd.h.

6.5.2.251 UINT32 FSP_S_CONFIG::Uart2KernelDebugBaseAddress

Offset 0x00DC - UART Debug Base Address UART Debug Base Address.

0x00000000(Default).

Definition at line 786 of file FspsUpd.h.

6.5.2.252 UINT8 FSP_S_CONFIG::UfsEnabled

Offset 0x01B9 - UFS Support (D29:F0) Enable/Disable SDIO Support.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 1120 of file FspsUpd.h.

6.5.2.253 UINT8 FSP_S_CONFIG::UnitLevelClockGating

Offset 0x004E - Enable UnitLevelClockGating Enable/disable UnitLevelClockGating.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 265 of file FspsUpd.h.

6.5.2.254 UINT8 FSP_S_CONFIG::UnsolicitedAttackOverride

Offset 0x004A - Enable UnsolicitedAttackOverride Enable/disable UnsolicitedAttackOverride.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 241 of file FspsUpd.h.

6.5.2.255 UINT8 FSP_S_CONFIG::UnsupportedRequestReport[6]

Offset 0x0134 - URR PCI Express Unsupported Request Reporting Enable/Disable.

0:Disable(Default), 1:Enable.

Definition at line 889 of file FspsUpd.h.

6.5.2.256 UINT8 FSP_S_CONFIG::Usb30Mode

Offset 0x0246 - xHCI Mode Mode of operation of xHCI controller.

0:Disable, 1:Enable, 2:Auto(Default) 0:Disable, 1:Enable, 2:Auto

Definition at line 1195 of file FspsUpd.h.

6.5.2.257 UINT8 FSP_S_CONFIG::UsbOtg

Offset 0x0258 - XDCI Support Enable/Disable XDCI.

0:Disable, 1:PCI_Mode(Default), 2:ACPI_mode. 0:Disable, 1:PCI_Mode, 2:ACPI_mode

Definition at line 1216 of file FspsUpd.h.

6.5.2.258 UINT8 FSP_S_CONFIG::UsbPerPortCtl

Offset 0x0245 - USB Per-Port Control Control each of the USB ports enable/disable.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 1189 of file FspsUpd.h.

6.5.2.259 UINT8 FSP_S_CONFIG::VmxEnable

Offset 0x0024 - VMX Enable Enable or Disable VMX.

0:Disable, 1:Enable(Default). $EN_DIS

Definition at line 72 of file FspsUpd.h.

6.5.2.260 UINT8 FSP_S_CONFIG::VtdEnable

Offset 0x026A - VT-d Enable/Disable VT-d.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 1251 of file FspsUpd.h.

6.5.2.261 UINT8 FSP_S_CONFIG::WOPCMSize

Offset 0x004C - Enable WOPCMSize Enable/disable WOPCMSize.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 253 of file FspsUpd.h.

6.5.2.262 UINT8 FSP_S_CONFIG::WOPCMSupport

Offset 0x004B - Enable WOPCMSupport Enable/disable WOPCMSupport.

0:Disable(Default), 1:Enable. $EN_DIS

Definition at line 247 of file FspsUpd.h.

6.5.2.263 UINT8 FSP_S_CONFIG::WriteProtectionEnable[5]

Offset 0x0064 - Write Protection Support Enable/disable Write Protection.

0:Disable, 1:Enable(Default).

Definition at line 339 of file FspsUpd.h.

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

- FspsUpd.h

6.6 FSP_UPD_HEADER Struct Reference

Fsp UPD HEADER Configuration.

`#include <FspApi.h>`

Public Attributes

- UINT64 Signature

  UPD Region Signature.
- UINT8 Revision

  Revision of the Data structure.

6.6.1 Detailed Description

Fsp UPD HEADER Configuration.

FSP_UPD_HEADER Configuration.

Definition at line 23 of file BroxtonFspBinPkg/Include/FspApi.h.

6.6.2 Member Data Documentation

6.6.2.1 UINT8 FSP_UPD_HEADER::Revision

Revision of the Data structure.

For FSP v2.0 value is 1.

Definition at line 35 of file BroxtonFspBinPkg/Include/FspApi.h.

6.6.2.2 UINT64 FSP_UPD_HEADER::Signature

UPD Region Signature.

This signature will be "XXXXXX_T" for FSP-T "XXXXXX_M" for FSP-M "XXXXXX_S" for FSP-S Where XXXXXX is an unique signature

Definition at line 31 of file BroxtonFspBinPkg/Include/FspApi.h.

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

- BroxtonFspBinPkg/Include/FspApi.h

6.7 FSPM_ARCH_UPD Struct Reference

FSPM_ARCH_UPD Configuration.

`#include <FspApi.h>`

Public Attributes

- UINT8 Revision

  Revision of the structure.
- VOID * NvsBufferPtr

  Pointer to the non-volatile storage (NVS) data buffer.
- VOID * StackBase

  Pointer to the temporary stack base address to be consumed inside FspMemoryInit() API.
- UINT32 StackSize

  Temporary stack size to be consumed inside FspMemoryInit() API.
- UINT32 BootLoaderTolumSize

Size of memory to be reserved by FSP below "top of low usable memory" for bootloader usage.

- UINT32 BootMode

  Current boot mode.

6.7.1 Detailed Description

FSPM_ARCH_UPD Configuration.

Definition at line 42 of file BroxtonFspBinPkg/Include/FspApi.h.

6.7.2 Member Data Documentation

6.7.2.1 VOID * FSPM_ARCH_UPD::NvsBufferPtr

Pointer to the non-volatile storage (NVS) data buffer.

If it is NULL it indicates the NVS data is not available.

Definition at line 52 of file BroxtonFspBinPkg/Include/FspApi.h.

6.7.2.2 UINT8 FSPM_ARCH_UPD::Revision

Revision of the structure.

For FSP v2.0 value is 1.

Definition at line 46 of file BroxtonFspBinPkg/Include/FspApi.h.

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

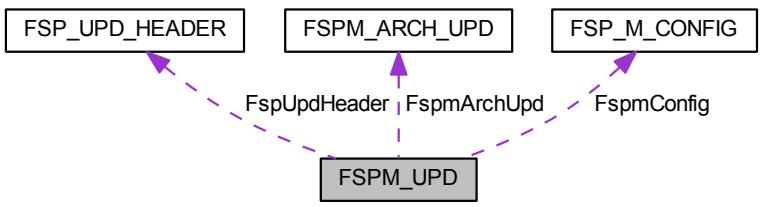
- BroxtonFspBinPkg/Include/FspApi.h

6.8 FSPM_UPD Struct Reference

Fsp M UPD Configuration.

`#include <FspmUpd.h>`

Collaboration diagram for FSPM_UPD:



Public Attributes

- FSP_UPD_HEADER FspUpdHeader

Offset 0x0000.

- FSPM_ARCH_UPD FspmArchUpd

  Offset 0x0020.

- FSP_M_CONFIG FspmConfig

  Offset 0x0040.

- UINT8 UnusedUpdSpace2 [158]

  Offset 0x0160.

- UINT16 UpdTerminator

  Offset 0x01FE.

6.8.1 Detailed Description

Fsp M UPD Configuration.

Definition at line 884 of file FspmUpd.h.

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

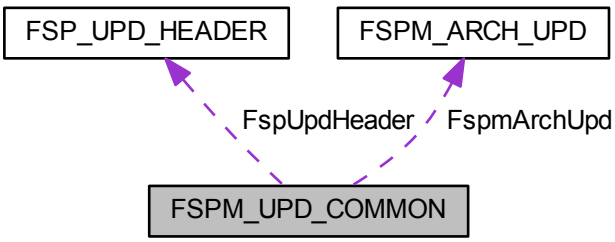
- FspmUpd.h

6.9 FSPM_UPD_COMMON Struct Reference

FSPM_UPD_COMMON Configuration.

`#include <FspApi.h>`

Collaboration diagram for FSPM_UPD_COMMON:



Public Attributes

- FSP_UPD_HEADER FspUpdHeader

  FSP_UPD_HEADER Configuration.

- FSPM_ARCH_UPD FspmArchUpd

  FSPM_ARCH_UPD Configuration.

6.9.1 Detailed Description

FSPM_UPD_COMMON Configuration.

Definition at line 79 of file BroxtonFspBinPkg/Include/FspApi.h.

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

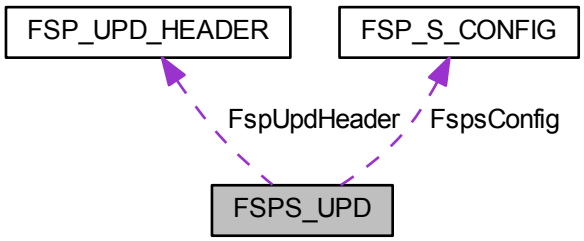
- BroxtonFspBinPkg/Include/FspApi.h

6.10 FSPS_UPD Struct Reference

Fsp S UPD Configuration.

`#include <FspsUpd.h>`

Collaboration diagram for FSPS_UPD:



Public Attributes

- FSP_UPD_HEADER FspUpdHeader

  Offset 0x0000.
- FSP_S_CONFIG FspsConfig

  Offset 0x0020.
- UINT8 UnusedUpdSpace8 [46]

  Offset 0x0380.
- UINT16 UpdTerminator

  Offset 0x03AE.

6.10.1 Detailed Description

Fsp S UPD Configuration.

Definition at line 1594 of file FspsUpd.h.

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

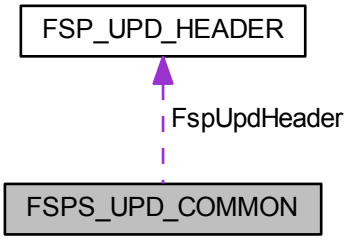
- FspsUpd.h

6.11 FSPS_UPD_COMMON Struct Reference

FSPS_UPD_COMMON Configuration.

`#include <FspApi.h>`

Collaboration diagram for FSPS_UPD_COMMON:



Public Attributes

- FSP_UPD_HEADER FspUpdHeader
  FSP_UPD_HEADER Configuration.

6.11.1 Detailed Description

FSPS_UPD_COMMON Configuration.

Definition at line 84 of file BroxtonFspBinPkg/Include/FspApi.h.

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

- BroxtonFspBinPkg/Include/FspApi.h

6.12 FSPT_COMMON_UPD Struct Reference

Fsp T Common UPD.

`#include <FsptUpd.h>`

Public Attributes

- UINT8 Revision
  Offset 0x0020.
- UINT8 Reserved [3]
  Offset 0x0021.
- UINT32 MicrocodeRegionBase
  Offset 0x0024.
- UINT32 MicrocodeRegionLength
  Offset 0x0028.

- UINT32 CodeRegionBase

  Offset 0x002C.
- UINT32 CodeRegionLength

  Offset 0x0030.
- UINT8 Reserved1 [12]

  Offset 0x0034.

6.12.1 Detailed Description

Fsp T Common UPD.

Definition at line 43 of file FsptUpd.h.

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

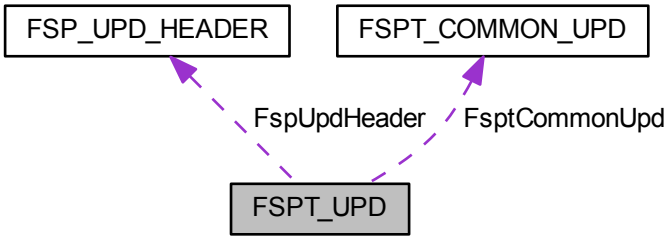
- FsptUpd.h

6.13 FSPT_UPD Struct Reference

Fsp T UPD Configuration.

`#include <FsptUpd.h>`

Collaboration diagram for FSPT_UPD:



Public Attributes

- FSP_UPD_HEADER FspUpdHeader

  Offset 0x0000.
- FSPT_COMMON_UPD FsptCommonUpd

  Offset 0x0020.
- UINT8 ReservedFsptUpd1 [16]

  Offset 0x0040.
- UINT8 UnusedUpdSpace0 [6]

  Offset 0x0050.
- UINT16 UpdTerminator

  Offset 0x0056.

6.13.1 Detailed Description

Fsp T UPD Configuration.

Definition at line 76 of file FsptUpd.h.

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

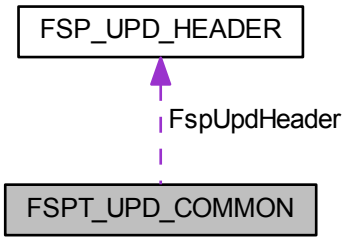
- FsptUpd.h

6.14 FSPT_UPD_COMMON Struct Reference

FSPT_UPD_COMMON Configuration.

`#include <FspApi.h>`

Collaboration diagram for FSPT_UPD_COMMON:



Public Attributes

- FSP_UPD_HEADER FspUpdHeader

  FSP_UPD_HEADER Configuration.

6.14.1 Detailed Description

FSPT_UPD_COMMON Configuration.

Definition at line 75 of file BroxtonFspBinPkg/Include/FspApi.h.

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

- BroxtonFspBinPkg/Include/FspApi.h

6.15 NOTIFY_PHASE_PARAMS Struct Reference

Definition of NOTIFY_PHASE_PARAMS.

`#include <FspApi.h>`

Public Attributes

- FSP_INIT_PHASE Phase

  Notification phase used for NotifyPhase API.

6.15.1 Detailed Description

Definition of NOTIFY_PHASE_PARAMS.

Definition at line 108 of file BroxtonFspBinPkg/Include/FspApi.h.

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

- BroxtonFspBinPkg/Include/FspApi.h

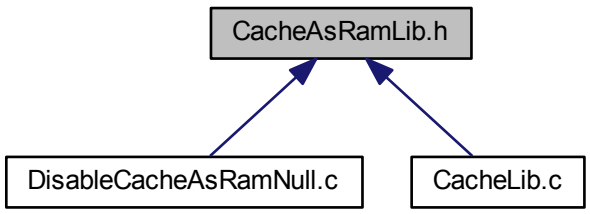
Chapter 7

File Documentation

7.1 CacheAsRamLib.h File Reference

Copyright (c) 2014, Intel Corporation.

This graph shows which files directly or indirectly include this file:



Functions

- VOID DisableCacheAsRam (IN BOOLEAN DisableCar)

  This function disable CAR.

7.1.1 Detailed Description

Copyright (c) 2014, 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.

7.1.2 Function Documentation

7.1.2.1 VOID DisableCacheAsRam (IN BOOLEAN *DisableCar*)

This function disable CAR.

Parameters

in	*DisableCar*	TRUE means use INVD, FALSE means use WBINVD

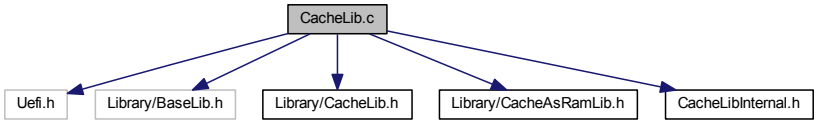
Definition at line 26 of file DisableCacheAsRamNull.c.

7.2 CacheLib.c File Reference

Copyright (c) 2014 - 2015, Intel Corporation.

```
#include <Uefi.h>
#include <Library/BaseLib.h>
#include <Library/CacheLib.h>
#include <Library/CacheAsRamLib.h>
#include "CacheLibInternal.h"
```

Include dependency graph for CacheLib.c:



Functions

- EFI_STATUS SearchForExactMtrr (IN EFI_PHYSICAL_ADDRESS MemoryAddress, IN UINT64 Memory↩ Length, IN UINT64 ValidMtrrAddressMask, OUT UINT32 ∗UsedMsrNum, OUT EFI_MEMORY_CACHE_T↩ YPE ∗MemoryCacheType)

  Search the memory cache type for specific memory from MTRR.
- BOOLEAN IsDefaultType (IN EFI_MEMORY_CACHE_TYPE MemoryCacheType)

  Check if CacheType match current default setting.
- UINT32 CheckMtrrAlignment (IN UINT64 BaseAddress, IN UINT64 Size)

  Return MTRR alignment requirement for base address and size.
- INT8 CheckDirection (IN UINT64 Input)

  Given the input, check if the number of MTRR is lesser.
- VOID EfiDisableCacheMtrr (OUT UINT64 ∗OldMtrr)

  Disable cache and its mtrr.
- VOID EfiRecoverCacheMtrr (IN BOOLEAN EnableMtrr, IN UINT64 OldMtrr)

  Recover cache MTRR.
- VOID EfiProgramMtrr (IN UINTN MtrrNumber, IN EFI_PHYSICAL_ADDRESS MemoryAddress, IN UINT64 MemoryLength, IN EFI_MEMORY_CACHE_TYPE MemoryCacheType, IN UINT64 ValidMtrrAddressMask)

  Programming MTRR according to Memory address, length, and type.
- UINT64 Power2MaxMemory (IN UINT64 MemoryAddress, IN UINT64 MemoryLength)

  Calculate the maximum value which is a power of 2, but less the MemoryLength.
- EFI_STATUS ProgramFixedMtrr (IN EFI_MEMORY_CACHE_TYPE MemoryCacheType, IN UINT64 ∗Base, IN UINT64 ∗Len)

  Programs fixed MTRRs registers.
- BOOLEAN CheckMtrrOverlap (IN EFI_PHYSICAL_ADDRESS Start, IN EFI_PHYSICAL_ADDRESS End)

  Check if there is a valid variable MTRR that overlaps the given range.
- EFI_STATUS SetCacheAttributes (IN EFI_PHYSICAL_ADDRESS MemoryAddress, IN UINT64 Memory↩ Length, IN EFI_MEMORY_CACHE_TYPE MemoryCacheType)

Given the memory range and cache type, programs the MTRRs.

- EFI_STATUS ResetCacheAttributes (VOID)

  Reset all the MTRRs to a known state.

7.2.1 Detailed Description

Copyright (c) 2014 - 2015, 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.

7.2.2 Function Documentation

7.2.2.1 INT8 CheckDirection (IN UINT64 *Input*)

Given the input, check if the number of MTRR is lesser.

if positive or subtractive.

Parameters

in	*Input*	Length of Memory to program MTRR.

Return values

Zero	do positive.
Non-Zero	do subtractive.

Definition at line 102 of file CacheLib.c.

7.2.2.2 UINT32 CheckMtrrAlignment (IN UINT64 *BaseAddress*, IN UINT64 *Size*)

Return MTRR alignment requirement for base address and size.

Parameters

in	*BaseAddress*	Base address.
in	*Size*	Size.

Return values

Zero	Alligned.
Non-Zero	Not alligned.

Definition at line 261 of file CacheLib.c.

7.2.2.3 BOOLEAN CheckMtrrOverlap (IN EFI_PHYSICAL_ADDRESS *Start*, IN EFI_PHYSICAL_ADDRESS *End*)

Check if there is a valid variable MTRR that overlaps the given range.

Parameters

in	*Start*	Base Address of the range to check.
in	*End*	End address of the range to check.

Return values

TRUE	Mtrr overlap.
FALSE	Mtrr not overlap.

Definition at line 354 of file CacheLib.c.

7.2.2.4 VOID EfiDisableCacheMtrr (OUT UINT64 * *OldMtrr*)

Disable cache and its mtrr.

Parameters

out	*OldMtrr*	To return the Old MTRR value

Definition at line 116 of file CacheLib.c.

7.2.2.5 VOID EfiProgramMtrr (IN UINTN *MtrrNumber,* IN EFI_PHYSICAL_ADDRESS *MemoryAddress,* IN UINT64 *MemoryLength,* IN EFI_MEMORY_CACHE_TYPE *MemoryCacheType,* IN UINT64 *ValidMtrrAddressMask*)

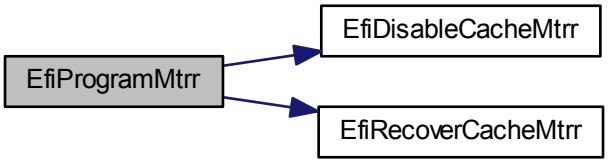
Programming MTRR according to Memory address, length, and type.

Parameters

in	*MtrrNumber*	the variable MTRR index number
in	*MemoryAddress*	the address of target memory
in	*MemoryLength*	the length of target memory
in	*MemoryCacheType*	the cache type of target memory
in	*ValidMtrrAddressMask*	the MTRR address mask

Definition at line 172 of file CacheLib.c.

Here is the call graph for this function:



7.2.2.6 VOID EfiRecoverCacheMtrr (IN BOOLEAN *EnableMtrr,* IN UINT64 *OldMtrr*)

Recover cache MTRR.

Parameters

in	*EnableMtrr*	Whether to enable the MTRR
in	*OldMtrr*	The saved old MTRR value to restore when not to enable the MTRR

Definition at line 139 of file CacheLib.c.

7.2.2.7 BOOLEAN IsDefaultType (IN EFI_MEMORY_CACHE_TYPE *MemoryCacheType*)

Check if CacheType match current default setting.

Parameters

in	*MemoryCache↩ Type*	input cache type to be checked.

Return values

TRUE	MemoryCacheType is default MTRR setting.
FALSE	MemoryCacheType is NOT default MTRR setting.

Parameters

in	*MemoryCache↩ Type*	input cache type to be checked.

Return values

TRUE	MemoryCacheType is default MTRR setting.
TRUE	MemoryCacheType is NOT default MTRR setting.

Definition at line 693 of file CacheLib.c.

7.2.2.8 UINT64 Power2MaxMemory (IN UINT64 *MemoryAddress,* IN UINT64 *MemoryLength*)

Calculate the maximum value which is a power of 2, but less the MemoryLength.

Parameters

in	*MemoryAddress*	Memory address.
in	*MemoryLength*	The number to pass in.

Returns

The maximum value which is align to power of 2 and less the MemoryLength

Definition at line 214 of file CacheLib.c.

Here is the call graph for this function:

7.2.2.9 EFI_STATUS ProgramFixedMtrr (IN EFI_MEMORY_CACHE_TYPE *MemoryCacheType,* IN UINT64 * *Base,* IN UINT64 * *Len*)

Programs fixed MTRRs registers.

Parameters

in	*MemoryCache↩ Type*	The memory type to set.
in	*Base*	The base address of memory range.
in	*Length*	The length of memory range.

Return values

RETURN_SUCCESS	The cache type was updated successfully
RETURN_UNSUPPORT↩ ED	The requested range or cache type was invalid for the fixed MTRRs.

Definition at line 296 of file CacheLib.c.

7.2.2.10 EFI_STATUS ResetCacheAttributes (VOID)

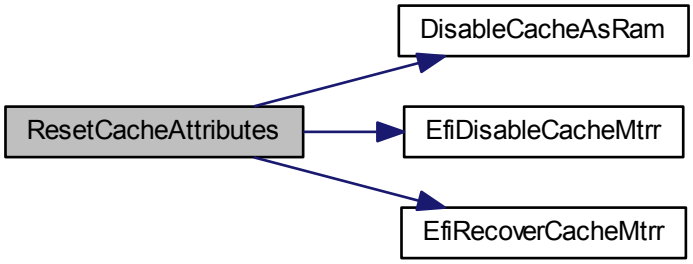
Reset all the MTRRs to a known state.

Return values

EFI_SUCCESS	All MTRRs have been reset successfully.

Definition at line 578 of file CacheLib.c.

Here is the call graph for this function:



7.2.2.11 EFI_STATUS SearchForExactMtrr (IN EFI_PHYSICAL_ADDRESS *MemoryAddress*, IN UINT64 *MemoryLength*, IN UINT64 *ValidMtrrAddressMask*, OUT UINT32 * *UsedMsrNum*, OUT EFI_MEMORY_CACHE_TYPE * *UsedMemoryCacheType*)

Search the memory cache type for specific memory from MTRR.

Parameters

in	*MemoryAddress*	the address of target memory
in	*MemoryLength*	the length of target memory
in	*ValidMtrr↩ AddressMask*	the MTRR address mask

out	*UsedMsrNum*	the used MSR number
out	*UsedMemory↩ CacheType*	the cache type for the target memory

Return values

EFI_SUCCESS	The memory is found in MTRR and cache type is returned
EFI_NOT_FOUND	The memory is not found in MTRR

Definition at line 644 of file CacheLib.c.

7.2.2.12 EFI_STATUS SetCacheAttributes (IN EFI_PHYSICAL_ADDRESS *MemoryAddress,* IN UINT64 *MemoryLength,* IN EFI_MEMORY_CACHE_TYPE *MemoryCacheType*)

Given the memory range and cache type, programs the MTRRs.

Parameters

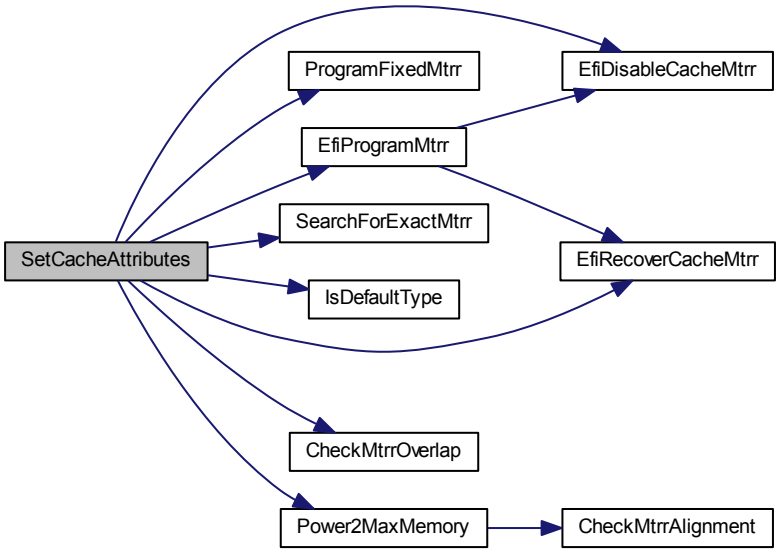
in	*MemoryAddress*	Base Address of Memory to program MTRR.
in	*MemoryLength*	Length of Memory to program MTRR.
in	*MemoryCache↩ Type*	Cache Type.

Return values

EFI_SUCCESS	Mtrr are set successfully.
EFI_LOAD_ERROR	No empty MTRRs to use.
EFI_INVALID_PARAMET↩ ER	The input parameter is not valid.
others	An error occurs when setting MTTR.

Definition at line 377 of file CacheLib.c.

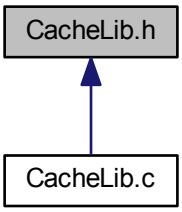
Here is the call graph for this function:



7.3 CacheLib.h File Reference

Copyright (c) 2014, Intel Corporation.

This graph shows which files directly or indirectly include this file:



Functions

- EFI_STATUS ResetCacheAttributes (VOID)

  Reset all the MTRRs to a known state.

- EFI_STATUS SetCacheAttributes (IN EFI_PHYSICAL_ADDRESS MemoryAddress, IN UINT64 Memory↩ Length, IN EFI_MEMORY_CACHE_TYPE MemoryCacheType)

  Given the memory range and cache type, programs the MTRRs.

7.3.1 Detailed Description

Copyright (c) 2014, 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.

7.3.2 Function Documentation

7.3.2.1 EFI_STATUS ResetCacheAttributes (VOID)

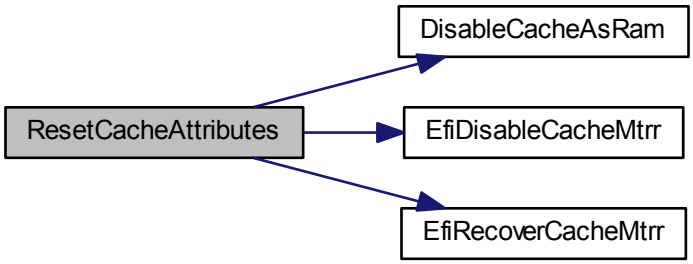
Reset all the MTRRs to a known state.

Return values

EFI_SUCCESS	All MTRRs have been reset successfully.

Definition at line 578 of file CacheLib.c.

Here is the call graph for this function:



7.3.2.2 EFI_STATUS SetCacheAttributes (IN EFI_PHYSICAL_ADDRESS *MemoryAddress,* IN UINT64 *MemoryLength,* IN EFI_MEMORY_CACHE_TYPE *MemoryCacheType*)

Given the memory range and cache type, programs the MTRRs.

Parameters

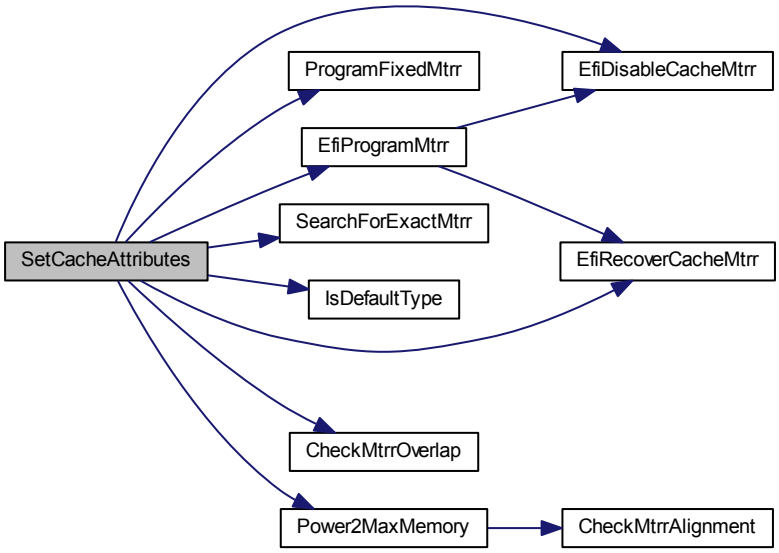
in	*MemoryAddress*	Base Address of Memory to program MTRR.
in	*MemoryLength*	Length of Memory to program MTRR.
in	*MemoryCache↩ Type*	Cache Type.

Return values

EFI_SUCCESS	Mtrr are set successfully.
EFI_LOAD_ERROR	No empty MTRRs to use.
EFI_INVALID_PARAMET↩ ER	The input parameter is not valid.
others	An error occurs when setting MTTR.

Definition at line 377 of file CacheLib.c.

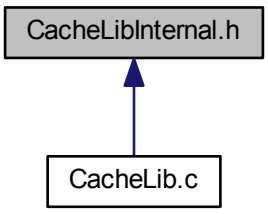
Here is the call graph for this function:



7.4 CacheLibInternal.h File Reference

Copyright (c) 2014, Intel Corporation.

This graph shows which files directly or indirectly include this file:



7.4.1 Detailed Description

Copyright (c) 2014, 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.

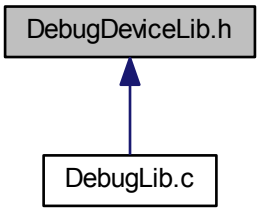
THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES

OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

7.5 DebugDeviceLib.h File Reference

Copyright (c) 2014, Intel Corporation.

This graph shows which files directly or indirectly include this file:



Functions

- UINT8 GetDebugPrintDeviceEnable (VOID)

  Returns the debug print device enable state.

7.5.1 Detailed Description

Copyright (c) 2014, 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.

7.5.2 Function Documentation

7.5.2.1 UINT8 GetDebugPrintDeviceEnable (VOID)

Returns the debug print device enable state.

Returns

Debug print device enable state.

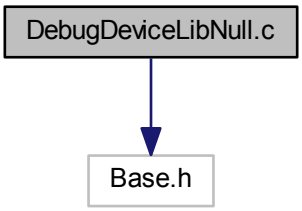
Definition at line 26 of file DebugDeviceLibNull.c.

7.6 DebugDeviceLibNull.c File Reference

Debug device library instance that retrieves the current enabling state for the platform debug output device.

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

Include dependency graph for DebugDeviceLibNull.c:



Functions

- UINT8 GetDebugPrintDeviceEnable (VOID)

  Returns the debug print device enable state.

7.6.1 Detailed Description

Debug device library instance that retrieves the current enabling state for the platform debug output device.

Copyright (c) 2014, 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.

7.6.2 Function Documentation

7.6.2.1 UINT8 GetDebugPrintDeviceEnable (VOID)

Returns the debug print device enable state.

Returns

Debug print device enable state.

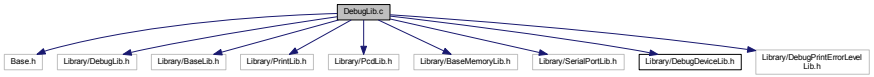
Definition at line 26 of file DebugDeviceLibNull.c.

7.7 DebugLib.c File Reference

Copyright (c) 2014 - 2015, Intel Corporation.

```
#include <Base.h>
#include <Library/DebugLib.h>
#include <Library/BaseLib.h>
#include <Library/PrintLib.h>
#include <Library/PcdLib.h>
#include <Library/BaseMemoryLib.h>
#include <Library/SerialPortLib.h>
#include <Library/DebugDeviceLib.h>
#include <Library/DebugPrintErrorLevelLib.h>
```

Include dependency graph for DebugLib.c:



Functions

- UINT32 * GetStackFramePointer (VOID)

  Get stack frame pointer of function call.
- VOID DebugPrint (IN UINTN ErrorLevel, IN CONST CHAR8 *Format,...)

  Prints a debug message to the debug output device if the specified error level is enabled.
- VOID FillHex (UINT32 Value, CHAR8 *Buffer)

  Convert an UINT32 value into HEX string sepcified by Buffer.
- VOID DebugAssertInternal (VOID)

  Prints an assert message containing a filename, line number, and description.
- VOID DebugAssert (IN CONST CHAR8 *FileName, IN UINTN LineNumber, IN CONST CHAR8 *Description)

  Prints an assert message containing a filename, line number, and description.
- VOID * DebugClearMemory (OUT VOID *Buffer, IN UINTN Length)

  Fills a target buffer with PcdDebugClearMemoryValue, and returns the target buffer.
- BOOLEAN DebugAssertEnabled (VOID)

  Returns TRUE if ASSERT() macros are enabled.
- BOOLEAN DebugPrintEnabled (VOID)

  Returns TRUE if DEBUG() macros are enabled.
- BOOLEAN DebugCodeEnabled (VOID)

  Returns TRUE if DEBUG_CODE() macros are enabled.
- BOOLEAN DebugClearMemoryEnabled (VOID)

  Returns TRUE if DEBUG_CLEAR_MEMORY() macro is enabled.
- BOOLEAN DebugPrintLevelEnabled (IN CONST UINTN ErrorLevel)

  Returns TRUE if any one of the bit is set both in ErrorLevel and PcdFixedDebugPrintErrorLevel.

7.7.1 Detailed Description

Copyright (c) 2014 - 2015, 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.

7.7.2 Function Documentation

7.7.2.1 VOID DebugAssert (IN CONST CHAR8 * *FileName,* IN UINTN *LineNumber,* IN CONST CHAR8 * *Description*)

Prints an assert message containing a filename, line number, and description.

This may be followed by a breakpoint or a dead loop.

Print a message of the form "ASSERT <FileName>(<LineNumber>): <Description>\n" to the debug output device. If DEBUG_PROPERTY_ASSERT_BREAKPOINT_ENABLED bit of PcdDebugProperyMask is set then CpuBreakpoint() is called. Otherwise, if DEBUG_PROPERTY_ASSERT_DEADLOOP_ENABLED bit of PcdDebugProperyMask is set then CpuDeadLoop() is called. If neither of these bits are set, then this function returns immediately after the message is printed to the debug output device. DebugAssert() must actively prevent recursion. If DebugAssert() is called while processing another DebugAssert(), then DebugAssert() must return immediately.

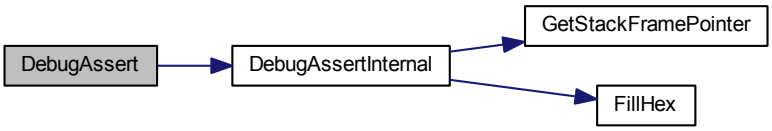
If FileName is NULL, then a <FileName> string of "(NULL) Filename" is printed. If Description is NULL, then a string of "(NULL) Description" is printed.

Parameters

FileName	The pointer to the name of the source file that generated the assert condition.
LineNumber	The line number in the source file that generated the assert condition
Description	The pointer to the description of the assert condition.

Definition at line 198 of file DebugLib.c.

Here is the call graph for this function:



7.7.2.2 BOOLEAN DebugAssertEnabled (VOID)

Returns TRUE if ASSERT() macros are enabled.

This function returns TRUE if the DEBUG_PROPERTY_DEBUG_ASSERT_ENABLED bit of PcdDebugProperyMask is set. Otherwise FALSE is returned.

Return values

TRUE	The DEBUG_PROPERTY_DEBUG_ASSERT_ENABLED bit of PcdDebugProperyMask is set.
FALSE	The DEBUG_PROPERTY_DEBUG_ASSERT_ENABLED bit of PcdDebugProperyMask is clear.

Definition at line 246 of file DebugLib.c.

7.7.2.3 VOID DebugAssertInternal (VOID)

Prints an assert message containing a filename, line number, and description.

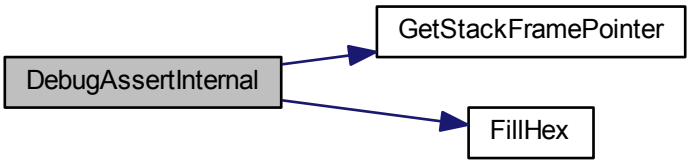
This may be followed by a breakpoint or a dead loop.

Print a message of the form "ASSERT <FileName>(<LineNumber>): <Description>\n" to the debug output device. If DEBUG_PROPERTY_ASSERT_BREAKPOINT_ENABLED bit of PcdDebugProperyMask is set then CpuBreakpoint() is called. Otherwise, if DEBUG_PROPERTY_ASSERT_DEADLOOP_ENABLED bit of PcdDebugProperyMask is set then CpuDeadLoop() is called. If neither of these bits are set, then this function returns immediately after the message is printed to the debug output device. DebugAssert() must actively prevent recursion. If DebugAssert() is called while processing another DebugAssert(), then DebugAssert() must return immediately.

If FileName is NULL, then a <FileName> string of "(NULL) Filename" is printed. If Description is NULL, then a string of "(NULL) Description" is printed.

Definition at line 139 of file DebugLib.c.

Here is the call graph for this function:



7.7.2.4 VOID* DebugClearMemory (OUT VOID * *Buffer,* IN UINTN *Length*)

Fills a target buffer with PcdDebugClearMemoryValue, and returns the target buffer.

This function fills Length bytes of Buffer with the value specified by PcdDebugClearMemoryValue, and returns Buffer.

If Buffer is NULL, then ASSERT(). If Length is greater than (MAX_ADDRESS - Buffer + 1), then ASSERT().

Parameters

Buffer	The pointer to the target buffer to be filled with PcdDebugClearMemoryValue.
Length	The number of bytes in Buffer to fill with zeros PcdDebugClearMemoryValue.

Returns

Buffer The pointer to the target buffer filled with PcdDebugClearMemoryValue.

Definition at line 225 of file DebugLib.c.

7.7.2.5 BOOLEAN DebugClearMemoryEnabled (VOID)

Returns TRUE if DEBUG_CLEAR_MEMORY() macro is enabled.

This function returns TRUE if the DEBUG_PROPERTY_CLEAR_MEMORY_ENABLED bit of PcdDebugProperyMask is set. Otherwise FALSE is returned.

Return values

TRUE	The DEBUG_PROPERTY_CLEAR_MEMORY_ENABLED bit of PcdDebug↩ ProperyMask is set.
FALSE	The DEBUG_PROPERTY_CLEAR_MEMORY_ENABLED bit of PcdDebug↩ ProperyMask is clear.

Definition at line 305 of file DebugLib.c.

7.7.2.6 BOOLEAN DebugCodeEnabled (VOID)

Returns TRUE if DEBUG_CODE() macros are enabled.

This function returns TRUE if the DEBUG_PROPERTY_DEBUG_CODE_ENABLED bit of PcdDebugProperyMask is set. Otherwise FALSE is returned.

Return values

TRUE	The DEBUG_PROPERTY_DEBUG_CODE_ENABLED bit of PcdDebug↩ ProperyMask is set.
FALSE	The DEBUG_PROPERTY_DEBUG_CODE_ENABLED bit of PcdDebug↩ ProperyMask is clear.

Definition at line 285 of file DebugLib.c.

7.7.2.7 VOID DebugPrint (IN UINTN *ErrorLevel,* IN CONST CHAR8 * *Format,* ...)

Prints a debug message to the debug output device if the specified error level is enabled.

If any bit in ErrorLevel is also set in DebugPrintErrorLevelLib function GetDebugPrintErrorLevel (), then print the message specified by Format and the associated variable argument list to the debug output device.

If Format is NULL, then ASSERT().

Parameters

ErrorLevel	The error level of the debug message.
Format	Format string for the debug message to print.
...	Variable argument list whose contents are accessed based on the format string specified by Format.

Definition at line 60 of file DebugLib.c.

Here is the call graph for this function:

7.7.2.8 BOOLEAN DebugPrintEnabled (VOID)

Returns TRUE if DEBUG() macros are enabled.

This function returns TRUE if the DEBUG_PROPERTY_DEBUG_PRINT_ENABLED bit of PcdDebugProperyMask is set. Otherwise FALSE is returned.

Return values

TRUE	The DEBUG_PROPERTY_DEBUG_PRINT_ENABLED bit of PcdDebug↩ ProperyMask is set.
FALSE	The DEBUG_PROPERTY_DEBUG_PRINT_ENABLED bit of PcdDebug↩ ProperyMask is clear.

Definition at line 266 of file DebugLib.c.

7.7.2.9 BOOLEAN DebugPrintLevelEnabled (IN CONST UINTN *ErrorLevel*)

Returns TRUE if any one of the bit is set both in ErrorLevel and PcdFixedDebugPrintErrorLevel.

This function compares the bit mask of ErrorLevel and PcdFixedDebugPrintErrorLevel.

Return values

TRUE	Current ErrorLevel is supported.
FALSE	Current ErrorLevel is not supported.

Definition at line 323 of file DebugLib.c.

7.7.2.10 VOID FillHex (UINT32 *Value,* CHAR8 * *Buffer*)

Convert an UINT32 value into HEX string sepcified by Buffer.

Parameters

Value	The HEX value to convert to string
Buffer	The pointer to the target buffer to be filled with HEX string

Definition at line 109 of file DebugLib.c.

7.7.2.11 UINT32* GetStackFramePointer (VOID)

Get stack frame pointer of function call.

Returns

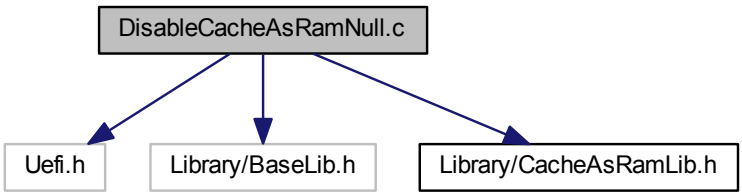
StackFramePointer stack frame pointer of function call.

7.8 DisableCacheAsRamNull.c File Reference

Copyright (c) 2014, Intel Corporation.

```
#include <Uefi.h>
#include <Library/BaseLib.h>
#include <Library/CacheAsRamLib.h>
```

Include dependency graph for DisableCacheAsRamNull.c:



Functions

- VOID DisableCacheAsRam (IN BOOLEAN DisableCar)

  This function disable CAR.

7.8.1 Detailed Description

Copyright (c) 2014, 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.

7.8.2 Function Documentation

7.8.2.1 VOID DisableCacheAsRam (IN BOOLEAN *DisableCar*)

This function disable CAR.

Parameters

in	*DisableCar*	TRUE means use INVD, FALSE means use WBINVD

Definition at line 26 of file DisableCacheAsRamNull.c.

7.9 DoxygenFspIntegrationGuide.h File Reference

This file contains doxygen ApolloLakeFspIntegration Guide.

7.9.1 Detailed Description

This file contains doxygen ApolloLakeFspIntegration Guide.

Copyright

Copyright (c) 2015 - 2016 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 such 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 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

7.10 FspApi.h File Reference

Intel FSP API definition from Intel Firmware Support Package External Architecture Specification v2.0, March 2016, revision 001.

Classes

- struct FSP_UPD_HEADER

  Fsp UPD HEADER Configuration.
- struct FSPM_ARCH_UPD

  FSPM_ARCH_UPD Configuration.
- struct FSPT_UPD_COMMON

  FSPT_UPD_COMMON Configuration.
- struct FSPM_UPD_COMMON

  FSPM_UPD_COMMON Configuration.
- struct FSPS_UPD_COMMON

  FSPS_UPD_COMMON Configuration.
- struct NOTIFY_PHASE_PARAMS

  Definition of NOTIFY_PHASE_PARAMS.

Typedefs

- typedef EFI_STATUS(∗ FSP_TEMP_RAM_INIT) (IN VOID ∗FsptUpdDataPtr)

  This FSP API is called soon after coming out of reset and before memory and stack is available.
- typedef EFI_STATUS(∗ FSP_NOTIFY_PHASE) (IN NOTIFY_PHASE_PARAMS ∗NotifyPhaseParamPtr)

  This FSP API is used to notify the FSP about the different phases in the boot process.
- typedef EFI_STATUS(∗ FSP_MEMORY_INIT) (IN VOID ∗FspmUpdDataPtr, OUT VOID ∗∗HobListPtr)

  This FSP API is called after TempRamInit and initializes the memory.
- typedef EFI_STATUS(∗ FSP_TEMP_RAM_EXIT) (IN VOID ∗TempRamExitParamPtr)

  This FSP API is called after FspMemoryInit API.
- typedef EFI_STATUS(∗ FSP_SILICON_INIT) (IN VOID ∗FspsUpdDataPtr)

  This FSP API is called after TempRamExit API.

Enumerations

- enum FSP_INIT_PHASE

7.10.1 Detailed Description

Intel FSP API definition from Intel Firmware Support Package External Architecture Specification v2.0, March 2016, revision 001.

Copyright (c) 2014 - 2016, 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.

7.10.2 Typedef Documentation

7.10.2.1 typedef EFI_STATUS(* FSP_MEMORY_INIT) (IN VOID *FspmUpdDataPtr, OUT VOID **HobListPtr)

This FSP API is called after TempRamInit and initializes the memory.

This FSP API accepts a pointer to a data structure that will be platform dependent and defined for each FSP binary. This will be documented in Integration guide with each FSP release. After FspMemInit completes its execution, it passes the pointer to the HobList and returns to the boot loader from where it was called. BootLoader is responsible to migrate it's stack and data to Memory. FspMemoryInit, TempRamExit and FspSiliconInit APIs provide an alternate method to complete the silicon initialization and provides bootloader an opportunity to get control after system memory is available and before the temporary RAM is torn down.

Parameters

in	*FspmUpdDataPtr*	Pointer to the FSPM_UPD data sructure.
out	*HobListPtr*	Pointer to receive the address of the HOB list.

Return values

EFI_SUCCESS	FSP execution environment was initialized successfully.
EFI_INVALID_PARAMETER	Input parameters are invalid.
EFI_UNSUPPORTED	The FSP calling conditions were not met.
EFI_DEVICE_ERROR	FSP initialization failed.
EFI_OUT_OF_RESOURCES	Stack range requested by FSP is not met.
FSP_STATUS_RESET_REQUIREDx	A reset is reuired. These status codes will not be returned during S3.

Definition at line 196 of file BroxtonFspBinPkg/Include/FspApi.h.

7.10.2.2 typedef EFI_STATUS(* FSP_NOTIFY_PHASE) (IN NOTIFY_PHASE_PARAMS *NotifyPhaseParamPtr)

This FSP API is used to notify the FSP about the different phases in the boot process.

This allows the FSP to take appropriate actions as needed during different initialization phases. The phases will be platform dependent and will be documented with the FSP release. The current FSP supports two notify phases: Post PCI enumeration Ready To Boot

Parameters

in	*NotifyPhaseParamPtr*	Address pointer to the NOTIFY_PHASE_PRAMS

Return values

EFI_SUCCESS	The notification was handled successfully.
EFI_UNSUPPORTED	The notification was not called in the proper order.
EFI_INVALID_PARAMET↩ ER	The notification code is invalid.

Definition at line 168 of file BroxtonFspBinPkg/Include/FspApi.h.

7.10.2.3 typedef EFI_STATUS(∗ FSP_SILICON_INIT) (IN VOID ∗FspsUpdDataPtr)

This FSP API is called after TempRamExit API.

FspMemoryInit, TempRamExit and FspSiliconInit APIs provide an alternate method to complete the silicon initialization.

Parameters

in	*FspsUpdDataPtr*	Pointer to the FSPS_UPD data structure. If NULL, FSP will use the default parameters.

Return values

EFI_SUCCESS	FSP execution environment was initialized successfully.
EFI_INVALID_PARAMET↩ ER	Input parameters are invalid.
EFI_UNSUPPORTED	The FSP calling conditions were not met.
EFI_DEVICE_ERROR	FSP initialization failed.
FSP_STATUS_RESET_R↩ EQUIREDx	A reset is reuired. These status codes will not be returned during S3.

Definition at line 243 of file BroxtonFspBinPkg/Include/FspApi.h.

7.10.2.4 typedef EFI_STATUS(∗ FSP_TEMP_RAM_EXIT) (IN VOID ∗TempRamExitParamPtr)

This FSP API is called after FspMemoryInit API.

This FSP API tears down the temporary memory setup by TempRamInit API. This FSP API accepts a pointer to a data structure that will be platform dependent and defined for each FSP binary. This will be documented in Integration Guide. FspMemoryInit, TempRamExit and FspSiliconInit APIs provide an alternate method to complete the silicon initialization and provides bootloader an opportunity to get control after system memory is available and before the temporary RAM is torn down.

Parameters

in	*TempRamExit↩ ParamPtr*	Pointer to the Temp Ram Exit parameters structure. This structure is normally defined in the Integration Guide. And if it is not defined in the Integration Guide, pass NULL.

Return values

EFI_SUCCESS	FSP execution environment was initialized successfully.
EFI_INVALID_PARAMET↩ ER	Input parameters are invalid.
EFI_UNSUPPORTED	The FSP calling conditions were not met.
EFI_DEVICE_ERROR	FSP initialization failed.

Definition at line 222 of file BroxtonFspBinPkg/Include/FspApi.h.

7.10.2.5 typedef EFI_STATUS(* FSP_TEMP_RAM_INIT) (IN VOID *FsptUpdDataPtr)

This FSP API is called soon after coming out of reset and before memory and stack is available.

This FSP API will load the microcode update, enable code caching for the region specified by the boot loader and also setup a temporary stack to be used until main memory is initialized.

A hardcoded stack can be set up with the following values, and the "esp" register initialized to point to this hardcoded stack.

1. The return address where the FSP will return control after setting up a temporary stack.
2. A pointer to the input parameter structure

However, since the stack is in ROM and not writeable, this FSP API cannot be called using the "call" instruction, but needs to be jumped to.

Parameters

in	*FsptUpdDataPtr*	Pointer to the FSPT_UPD data structure.

Return values

EFI_SUCCESS	Temporary RAM was initialized successfully.
EFI_INVALID_PARAMET↩ ER	Input parameters are invalid.
EFI_UNSUPPORTED	The FSP calling conditions were not met.
EFI_DEVICE_ERROR	Temp RAM initialization failed.

If this function is successful, the FSP initializes the ECX and EDX registers to point to a temporary but writeable memory range available to the boot loader and returns with FSP_SUCCESS in register EAX. Register ECX points to the start of this temporary memory range and EDX points to the end of the range. Boot loader is free to use the whole range described. Typically the boot loader can reload the ESP register to point to the end of this returned range so that it can be used as a standard stack.

Definition at line 148 of file BroxtonFspBinPkg/Include/FspApi.h.

7.10.3 Enumeration Type Documentation

7.10.3.1 enum FSP_INIT_PHASE

Enumerator

EnumInitPhaseAfterPciEnumeration This stage is notified when the bootloader completes the PCI enumeration and the resource allocation for the PCI devices is complete.

EnumInitPhaseReadyToBoot This stage is notified just before the bootloader hand-off to the OS loader.

EnumInitPhaseEndOfFirmware This stage is notified just before the firmware/Preboot environment transfers management of all system resources to the OS or next level execution environment.

EnumInitPhaseAfterPciEnumeration This stage is notified when the bootloader completes the PCI enumeration and the resource allocation for the PCI devices is complete.

EnumInitPhaseReadyToBoot This stage is notified just before the bootloader hand-off to the OS loader.

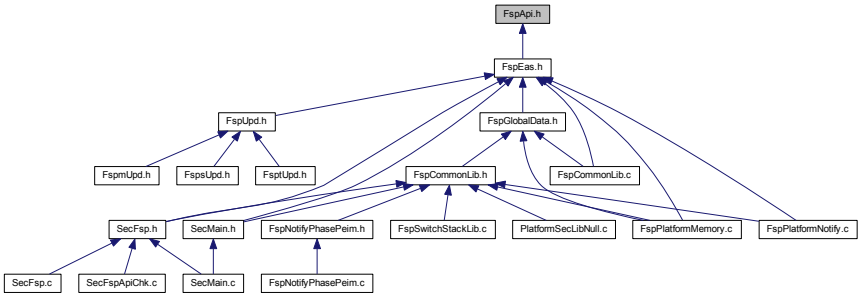
EnumInitPhaseEndOfFirmware This stage is notified just before the firmware/Preboot environment transfers management of all system resources to the OS or next level execution environment.

Definition at line 88 of file BroxtonFspBinPkg/Include/FspApi.h.

7.11 FspApi.h File Reference

Intel FSP API definition from Intel Firmware Support Package External Architecture Specification v2.0.

This graph shows which files directly or indirectly include this file:



Classes

- struct FSP_UPD_HEADER

  Fsp UPD HEADER Configuration.
- struct FSPM_ARCH_UPD

  FSPM_ARCH_UPD Configuration.
- struct FSPT_UPD_COMMON

  FSPT_UPD_COMMON Configuration.
- struct FSPM_UPD_COMMON

  FSPM_UPD_COMMON Configuration.
- struct FSPS_UPD_COMMON

  FSPS_UPD_COMMON Configuration.
- struct NOTIFY_PHASE_PARAMS

  Definition of NOTIFY_PHASE_PARAMS.

Macros

- #define FSP_STATUS_RESET_REQUIRED_COLD 0x40000001

  FSP Reset Status code These are defined in FSP EAS v2.0 section 11.2.2 - OEM Status Code.

Typedefs

- typedef EFI_STATUS(∗ FSP_TEMP_RAM_INIT) (IN VOID ∗FsptUpdDataPtr)

  This FSP API is called soon after coming out of reset and before memory and stack is available.
- typedef EFI_STATUS(∗ FSP_NOTIFY_PHASE) (IN NOTIFY_PHASE_PARAMS ∗NotifyPhaseParamPtr)

  This FSP API is used to notify the FSP about the different phases in the boot process.
- typedef EFI_STATUS(∗ FSP_MEMORY_INIT) (IN VOID ∗FspmUpdDataPtr, OUT VOID ∗∗HobListPtr)

  This FSP API is called after TempRamInit and initializes the memory.
- typedef EFI_STATUS(∗ FSP_TEMP_RAM_EXIT) (IN VOID ∗TempRamExitParamPtr)

  This FSP API is called after FspMemoryInit API.
- typedef EFI_STATUS(∗ FSP_SILICON_INIT) (IN VOID ∗FspsUpdDataPtr)

  This FSP API is called after TempRamExit API.

Enumerations

- enum FSP_INIT_PHASE

  Enumeration of FSP_INIT_PHASE for NOTIFY_PHASE.

7.11.1 Detailed Description

Intel FSP API definition from Intel Firmware Support Package External Architecture Specification v2.0.

Copyright (c) 2014 - 2016, 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.

7.11.2 Typedef Documentation

7.11.2.1 typedef EFI_STATUS(∗ FSP_MEMORY_INIT) (IN VOID ∗FspmUpdDataPtr, OUT VOID ∗∗HobListPtr)

This FSP API is called after TempRamInit and initializes the memory.

This FSP API accepts a pointer to a data structure that will be platform dependent and defined for each FSP binary. This will be documented in Integration guide with each FSP release. After FspMemInit completes its execution, it passes the pointer to the HobList and returns to the boot loader from where it was called. Boot Loader is responsible to migrate it's stack and data to Memory. FspMemoryInit, TempRamExit and FspSiliconInit APIs provide an alternate method to complete the silicon initialization and provides bootloader an opportunity to get control after system memory is available and before the temporary RAM is torn down.

Parameters

in	*FspmUpdDataPtr*	Pointer to the FSPM_UPD data sructure.
out	*HobListPtr*	Pointer to receive the address of the HOB list.

Return values

EFI_SUCCESS	FSP execution environment was initialized successfully.
EFI_INVALID_PARAMETER	Input parameters are invalid.
EFI_UNSUPPORTED	The FSP calling conditions were not met.
EFI_DEVICE_ERROR	FSP initialization failed.
EFI_OUT_OF_RESOURCES	Stack range requested by FSP is not met.
FSP_STATUS_RESET_REQUIREDx	A reset is reuired. These status codes will not be returned during S3.

Definition at line 237 of file IntelFsp2Pkg/Include/FspEas/FspApi.h.

7.11.2.2 typedef EFI_STATUS(∗ FSP_NOTIFY_PHASE) (IN NOTIFY_PHASE_PARAMS ∗NotifyPhaseParamPtr)

This FSP API is used to notify the FSP about the different phases in the boot process.

This allows the FSP to take appropriate actions as needed during different initialization phases. The phases will be platform dependent and will be documented with the FSP release. The current FSP supports two notify phases: Post PCI enumeration Ready To Boot

Parameters

in	*NotifyPhaseParamPtr*	Address pointer to the NOTIFY_PHASE_PRAMS

Return values

EFI_SUCCESS	The notification was handled successfully.
EFI_UNSUPPORTED	The notification was not called in the proper order.
EFI_INVALID_PARAMETER	The notification code is invalid.

Definition at line 209 of file IntelFsp2Pkg/Include/FspEas/FspApi.h.

7.11.2.3 typedef EFI_STATUS(* FSP_SILICON_INIT) (IN VOID *FspsUpdDataPtr)

This FSP API is called after TempRamExit API.

FspMemoryInit, TempRamExit and FspSiliconInit APIs provide an alternate method to complete the silicon initialization.

Parameters

in	*FspsUpdDataPtr*	Pointer to the FSPS_UPD data structure. If NULL, FSP will use the default parameters.

Return values

EFI_SUCCESS	FSP execution environment was initialized successfully.
EFI_INVALID_PARAMETER	Input parameters are invalid.
EFI_UNSUPPORTED	The FSP calling conditions were not met.
EFI_DEVICE_ERROR	FSP initialization failed.
FSP_STATUS_RESET_REQUIRED	A reset is reuired. These status codes will not be returned during S3.

Definition at line 284 of file IntelFsp2Pkg/Include/FspEas/FspApi.h.

7.11.2.4 typedef EFI_STATUS(* FSP_TEMP_RAM_EXIT) (IN VOID *TempRamExitParamPtr)

This FSP API is called after FspMemoryInit API.

This FSP API tears down the temporary memory setup by TempRamInit API. This FSP API accepts a pointer to a data structure that will be platform dependent and defined for each FSP binary. This will be documented in Integration Guide. FspMemoryInit, TempRamExit and FspSiliconInit APIs provide an alternate method to complete the silicon initialization and provides bootloader an opportunity to get control after system memory is available and before the temporary RAM is torn down.

Parameters

in	*TempRamExitParamPtr*	Pointer to the Temp Ram Exit parameters structure. This structure is normally defined in the Integration Guide. And if it is not defined in the Integration Guide, pass NULL.

Return values

EFI_SUCCESS	FSP execution environment was initialized successfully.
EFI_INVALID_PARAMETER	Input parameters are invalid.
EFI_UNSUPPORTED	The FSP calling conditions were not met.
EFI_DEVICE_ERROR	FSP initialization failed.

Definition at line 263 of file IntelFsp2Pkg/Include/FspEas/FspApi.h.

7.11.2.5 typedef EFI_STATUS(* FSP_TEMP_RAM_INIT) (IN VOID *FsptUpdDataPtr)

This FSP API is called soon after coming out of reset and before memory and stack is available.

This FSP API will load the microcode update, enable code caching for the region specified by the boot loader and also setup a temporary stack to be used until main memory is initialized.

A hardcoded stack can be set up with the following values, and the "esp" register initialized to point to this hardcoded stack.

1. The return address where the FSP will return control after setting up a temporary stack.
2. A pointer to the input parameter structure

However, since the stack is in ROM and not writeable, this FSP API cannot be called using the "call" instruction, but needs to be jumped to.

Parameters

in	*FsptUpdDataPtr*	Pointer to the FSPT_UPD data structure.

Return values

EFI_SUCCESS	Temporary RAM was initialized successfully.
EFI_INVALID_PARAMET↩ ER	Input parameters are invalid.
EFI_UNSUPPORTED	The FSP calling conditions were not met.
EFI_DEVICE_ERROR	Temp RAM initialization failed.

If this function is successful, the FSP initializes the ECX and EDX registers to point to a temporary but writeable memory range available to the boot loader and returns with FSP_SUCCESS in register EAX. Register ECX points to the start of this temporary memory range and EDX points to the end of the range. Boot loader is free to use the whole range described. Typically the boot loader can reload the ESP register to point to the end of this returned range so that it can be used as a standard stack.

Definition at line 189 of file IntelFsp2Pkg/Include/FspEas/FspApi.h.

7.11.3 Enumeration Type Documentation

7.11.3.1 enum FSP_INIT_PHASE

Enumeration of FSP_INIT_PHASE for NOTIFY_PHASE.

Enumerator

EnumInitPhaseAfterPciEnumeration This stage is notified when the bootloader completes the PCI enumeration and the resource allocation for the PCI devices is complete.

EnumInitPhaseReadyToBoot This stage is notified just before the bootloader hand-off to the OS loader.

EnumInitPhaseEndOfFirmware This stage is notified just before the firmware/Preboot environment transfers management of all system resources to the OS or next level execution environment.

EnumInitPhaseAfterPciEnumeration This stage is notified when the bootloader completes the PCI enumeration and the resource allocation for the PCI devices is complete.

EnumInitPhaseReadyToBoot This stage is notified just before the bootloader hand-off to the OS loader.

EnumInitPhaseEndOfFirmware This stage is notified just before the firmware/Preboot environment transfers management of all system resources to the OS or next level execution environment.

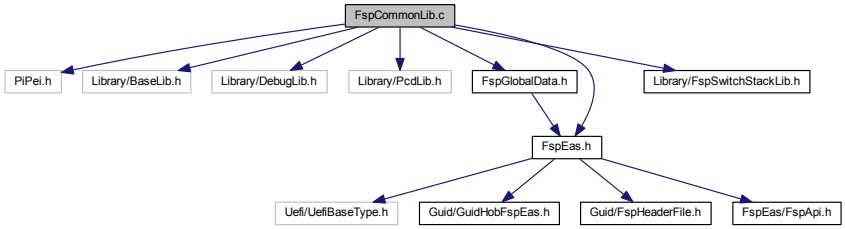
Definition at line 126 of file IntelFsp2Pkg/Include/FspEas/FspApi.h.

7.12 FspCommonLib.c File Reference

Copyright (c) 2014 - 2016, Intel Corporation.

```
#include <PiPei.h>
#include <Library/BaseLib.h>
#include <Library/DebugLib.h>
#include <Library/PcdLib.h>
#include <FspGlobalData.h>
#include <FspEas.h>
#include <Library/FspSwitchStackLib.h>
```

Include dependency graph for FspCommonLib.c:



Functions

- VOID SetFspGlobalDataPointer (IN FSP_GLOBAL_DATA *FspData)

  This function sets the FSP global data pointer.
- FSP_GLOBAL_DATA * GetFspGlobalDataPointer (VOID)

  This function gets the FSP global data pointer.
- UINT32 GetFspApiParameter (VOID)

  This function gets back the FSP API first parameter passed by the bootlaoder.
- UINT32 GetFspApiParameter2 (VOID)

  This function gets back the FSP API second parameter passed by the bootlaoder.
- VOID SetFspApiParameter (IN UINT32 Value)

  This function sets the FSP API parameter in the stack.
- VOID SetFspApiReturnStatus (IN UINT32 ReturnStatus)

  This function set the API status code returned to the BootLoader.
- VOID SetFspCoreStackPointer (IN VOID *NewStackTop)

  This function sets the context switching stack to a new stack frame.
- VOID SetFspPlatformDataPointer (IN VOID *PlatformData)

  This function sets the platform specific data pointer.
- VOID * GetFspPlatformDataPointer (VOID)

  This function gets the platform specific data pointer.
- VOID SetFspUpdDataPointer (IN VOID *UpdDataPtr)

  This function sets the UPD data pointer.
- VOID * GetFspUpdDataPointer (VOID)

  This function gets the UPD data pointer.
- VOID SetFspMemoryInitUpdDataPointer (IN VOID *MemoryInitUpdPtr)

  This function sets the FspMemoryInit UPD data pointer.
- VOID * GetFspMemoryInitUpdDataPointer (VOID)

  This function gets the FspMemoryInit UPD data pointer.
- VOID SetFspSiliconInitUpdDataPointer (IN VOID *SiliconInitUpdPtr)

This function sets the FspSiliconInit UPD data pointer.

- VOID * GetFspSiliconInitUpdDataPointer (VOID)

  This function gets the FspSiliconInit UPD data pointer.

- UINT64 SetFspMeasurePoint (IN UINT8 Id)

  Set FSP measurement point timestamp.

- FSP_INFO_HEADER * GetFspInfoHeader (VOID)

  This function gets the FSP info header pointer.

- VOID SetFspInfoHeader (FSP_INFO_HEADER *FspInfoHeader)

  This function sets the FSP info header pointer.

- FSP_INFO_HEADER * GetFspInfoHeaderFromApiContext (VOID)

  This function gets the FSP info header pointer using the API stack context.

- VOID * GetFspCfgRegionDataPointer (VOID)

  This function gets the CfgRegion data pointer.

- UINT8 GetFspApiCallingIndex (VOID)

  This function gets FSP API calling index.

- VOID SetFspApiCallingIndex (UINT8 Index)

  This function sets FSP API calling mode.

- UINT32 GetPhaseStatusCode (VOID)

  This function gets FSP Phase StatusCode.

- VOID SetPhaseStatusCode (UINT32 StatusCode)

  This function sets FSP Phase StatusCode.

- VOID FspApiReturnStatusReset (IN UINT32 FspResetType)

  This function updates the return status of the FSP API with requested reset type and returns to Boot Loader.

7.12.1 Detailed Description

Copyright (c) 2014 - 2016, 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.

7.12.2 Function Documentation

7.12.2.1 VOID FspApiReturnStatusReset (IN UINT32 *FspResetType*)

This function updates the return status of the FSP API with requested reset type and returns to Boot Loader.

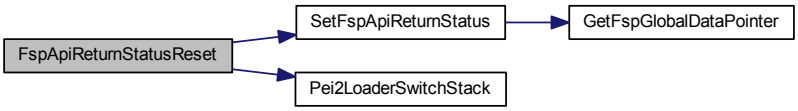
Parameters

in	*FspResetType*	Reset type that needs to returned as API return status

Below code is not an infinite loop.The control will go back to API calling function in BootLoader each time BootLoader calls the FSP API without honoring the reset request by FSP

Definition at line 514 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.2 UINT8 GetFspApiCallingIndex (VOID)

This function gets FSP API calling index.

This function gets FSP API calling mode.

Return values

API	calling index

Definition at line 451 of file FspCommonLib.c.

Here is the call graph for this function:

7.12.2.3 UINT32 GetFspApiParameter (VOID)

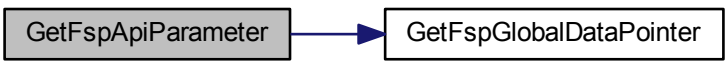
This function gets back the FSP API first parameter passed by the bootlaoder.

Return values

ApiParameter	FSP API first parameter passed by the bootlaoder.

Definition at line 96 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.4 UINT32 GetFspApiParameter2 (VOID)

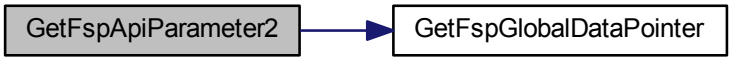
This function gets back the FSP API second parameter passed by the bootlaoder.

Return values

ApiParameter	FSP API second parameter passed by the bootlaoder.

Definition at line 113 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.5 VOID* GetFspCfgRegionDataPointer (VOID)

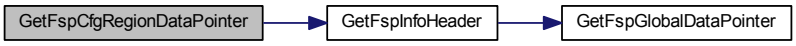
This function gets the CfgRegion data pointer.

Returns

CfgRegion data pointer.

Definition at line 434 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.6 FSP_INFO_HEADER* GetFspInfoHeader (VOID)

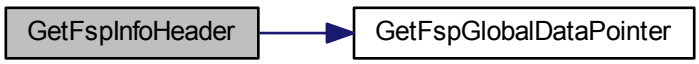
This function gets the FSP info header pointer.

Return values

FspInfoHeader	FSP info header pointer

Definition at line 389 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.7 FSP_INFO_HEADER* GetFspInfoHeaderFromApiContext (VOID)

This function gets the FSP info header pointer using the API stack context.

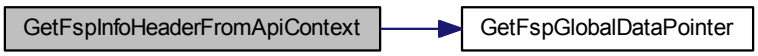
This function gets the FSP info header pointer from the API context.

Return values

FspInfoHeader	FSP info header pointer using the API stack context

Definition at line 417 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.8 VOID* GetFspMemoryInitUpdDataPointer (VOID)

This function gets the FspMemoryInit UPD data pointer.

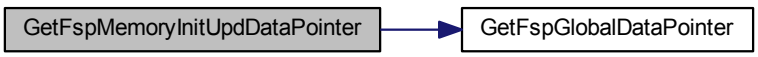
This function gets the memory init UPD data pointer.

Returns

FspMemoryInit UPD data pointer.

Definition at line 301 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.9 VOID* GetFspPlatformDataPointer (VOID)

This function gets the platform specific data pointer.

Parameters

in	*PlatformData*	FSP platform specific data pointer.

Definition at line 217 of file FspCommonLib.c.

Here is the call graph for this function:

7.12.2.10 VOID* GetFspSiliconInitUpdDataPointer (VOID)

This function gets the FspSiliconInit UPD data pointer.

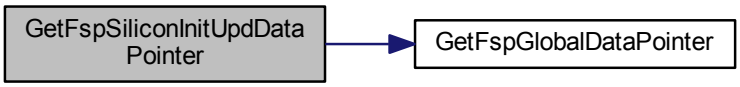
This function gets the silicon init UPD data pointer.

Returns

FspSiliconInit UPD data pointer.

Definition at line 343 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.11 VOID* GetFspUpdDataPointer (VOID)

This function gets the UPD data pointer.

Returns

UpdDataPtr UPD data pointer.

Definition at line 259 of file FspCommonLib.c.

Here is the call graph for this function:

7.12.2.12 UINT32 GetPhaseStatusCode (VOID)

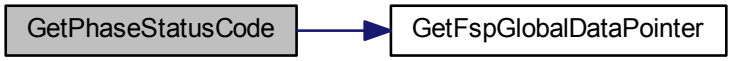
This function gets FSP Phase StatusCode.

Return values

StatusCode	

Definition at line 482 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.13 VOID SetFspApiCallingIndex (UINT8 *Index*)

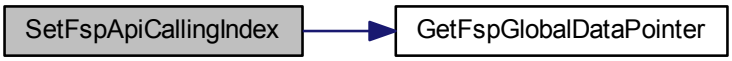
This function sets FSP API calling mode.

Parameters

in	*Index*	API calling index

Definition at line 465 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.14 VOID SetFspApiParameter (IN UINT32 *Value*)

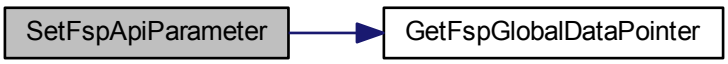
This function sets the FSP API parameter in the stack.

Parameters

in	*Value*	New parameter value.

Definition at line 131 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.15 VOID SetFspApiReturnStatus (IN UINT32 *ReturnStatus*)

This function set the API status code returned to the BootLoader.

Parameters

in	*ReturnStatus*	Status code to return.

Definition at line 149 of file FspCommonLib.c.

Here is the call graph for this function:

7.12.2.16 VOID SetFspCoreStackPointer (IN VOID * *NewStackTop*)

This function sets the context switching stack to a new stack frame.

Parameters

in	*NewStackTop*	New core stack to be set.

Definition at line 167 of file FspCommonLib.c.

Here is the call graph for this function:

7.12.2.17 VOID SetFspGlobalDataPointer (IN FSP_GLOBAL_DATA * *FspData*)

This function sets the FSP global data pointer.

Parameters

in	*FspData*	FSP global data pointer.

Definition at line 65 of file FspCommonLib.c.

7.12.2.18 VOID SetFspInfoHeader (FSP_INFO_HEADER * *FspInfoHeader*)

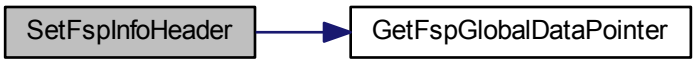
This function sets the FSP info header pointer.

Parameters

in	*FspInfoHeader*	FSP info header pointer

Definition at line 403 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.19 UINT64 SetFspMeasurePoint (IN UINT8 *Id*)

Set FSP measurement point timestamp.

Parameters

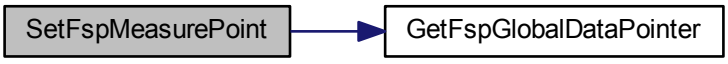
in	*Id*	Measurement point ID.

Returns

performance timestamp.

Definition at line 363 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.20 VOID SetFspMemoryInitUpdDataPointer (IN VOID * *MemoryInitUpdPtr*)

This function sets the FspMemoryInit UPD data pointer.

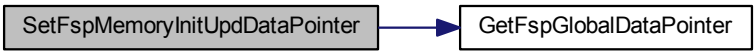
This function sets the memory init UPD data pointer.

Parameters

in	*MemoryInit↩ UpdPtr*	FspMemoryInit UPD data pointer.

Definition at line 277 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.21 VOID SetFspPlatformDataPointer (IN VOID * *PlatformData*)

This function sets the platform specific data pointer.

Parameters

in	*PlatformData*	FSP platform specific data pointer.

Definition at line 198 of file FspCommonLib.c.

Here is the call graph for this function:

7.12.2.22 VOID SetFspSiliconInitUpdDataPointer (IN VOID * *SiliconInitUpdPtr*)

This function sets the FspSiliconInit UPD data pointer.

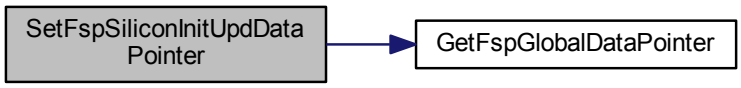
This function sets the silicon init UPD data pointer.

Parameters

in	*SiliconInitUpdPtr*	FspSiliconInit UPD data pointer.

Definition at line 319 of file FspCommonLib.c.

Here is the call graph for this function:



7.12.2.23 VOID SetFspUpdDataPointer (IN VOID * *UpdDataPtr*)

This function sets the UPD data pointer.

Parameters

in	*UpdDataPtr*	UPD data pointer.

Definition at line 235 of file FspCommonLib.c.

Here is the call graph for this function:

7.12.2.24 VOID SetPhaseStatusCode (UINT32 *StatusCode*)

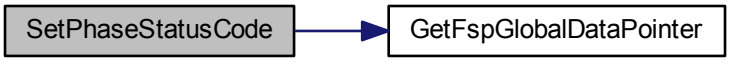
This function sets FSP Phase StatusCode.

Parameters

in	*Mode*	Phase StatusCode

Definition at line 496 of file FspCommonLib.c.

Here is the call graph for this function:

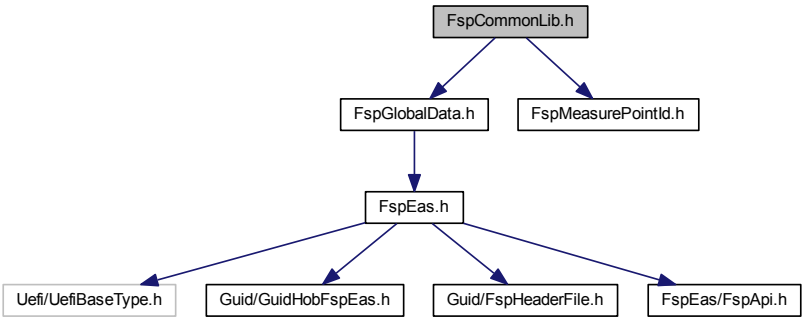


7.13 FspCommonLib.h File Reference

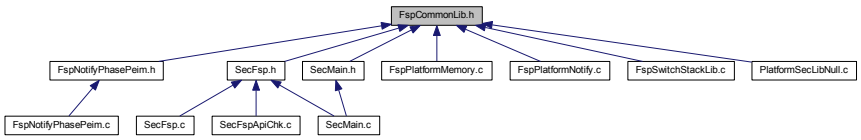
Copyright (c) 2014 - 2016, Intel Corporation.

```
#include <FspGlobalData.h>
#include <FspMeasurePointId.h>
```

Include dependency graph for FspCommonLib.h:



This graph shows which files directly or indirectly include this file:



Functions

- VOID SetFspGlobalDataPointer (IN FSP_GLOBAL_DATA *FspData)

  This function sets the FSP global data pointer.
- FSP_GLOBAL_DATA * GetFspGlobalDataPointer (VOID)

  This function gets the FSP global data pointer.
- UINT32 GetFspApiParameter (VOID)

  This function gets back the FSP API first parameter passed by the bootlaoder.
- UINT32 GetFspApiParameter2 (VOID)

  This function gets back the FSP API second parameter passed by the bootlaoder.
- VOID SetFspApiParameter (IN UINT32 Value)

  This function sets the FSP API parameter in the stack.
- VOID SetFspApiReturnStatus (IN UINT32 ReturnStatus)

  This function set the API status code returned to the BootLoader.
- VOID SetFspCoreStackPointer (IN VOID *NewStackTop)

  This function sets the context switching stack to a new stack frame.
- VOID SetFspPlatformDataPointer (IN VOID *PlatformData)

  This function sets the platform specific data pointer.
- VOID * GetFspPlatformDataPointer (VOID)

  This function gets the platform specific data pointer.

- VOID SetFspUpdDataPointer (IN VOID *UpdDataPtr)

  This function sets the UPD data pointer.
- VOID * GetFspUpdDataPointer (VOID)

  This function gets the UPD data pointer.
- VOID SetFspMemoryInitUpdDataPointer (IN VOID *MemoryInitUpdPtr)

  This function sets the memory init UPD data pointer.
- VOID * GetFspMemoryInitUpdDataPointer (VOID)

  This function gets the memory init UPD data pointer.
- VOID SetFspSiliconInitUpdDataPointer (IN VOID *SiliconInitUpdPtr)

  This function sets the silicon init UPD data pointer.
- VOID * GetFspSiliconInitUpdDataPointer (VOID)

  This function gets the silicon init UPD data pointer.
- UINT64 SetFspMeasurePoint (IN UINT8 Id)

  Set FSP measurement point timestamp.
- FSP_INFO_HEADER * GetFspInfoHeader (VOID)

  This function gets the FSP info header pointer.
- VOID SetFspInfoHeader (FSP_INFO_HEADER *FspInfoHeader)

  This function sets the FSP info header pointer.
- FSP_INFO_HEADER * GetFspInfoHeaderFromApiContext (VOID)

  This function gets the FSP info header pointer from the API context.
- VOID * GetFspCfgRegionDataPointer (VOID)

  This function gets the CfgRegion data pointer.
- UINT8 GetFspApiCallingIndex (VOID)

  This function gets FSP API calling mode.
- VOID SetFspApiCallingIndex (UINT8 Index)

  This function sets FSP API calling mode.
- UINT32 GetPhaseStatusCode (VOID)

  This function gets FSP Phase StatusCode.
- VOID SetPhaseStatusCode (UINT32 StatusCode)

  This function sets FSP Phase StatusCode.
- VOID FspApiReturnStatusReset (IN UINT32 FspResetType)

  This function updates the return status of the FSP API with requested reset type and returns to Boot Loader.

7.13.1 Detailed Description

Copyright (c) 2014 - 2016, 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.

7.13.2 Function Documentation

7.13.2.1 VOID FspApiReturnStatusReset (IN UINT32 *FspResetType*)

This function updates the return status of the FSP API with requested reset type and returns to Boot Loader.

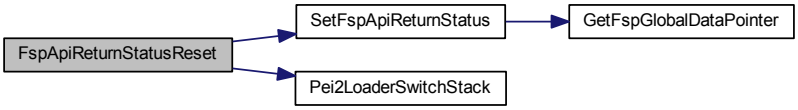
Parameters

in	*FspResetType*	Reset type that needs to returned as API return status

Below code is not an infinite loop.The control will go back to API calling function in BootLoader each time BootLoader calls the FSP API without honoring the reset request by FSP

Definition at line 514 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.2 UINT8 GetFspApiCallingIndex (VOID)

This function gets FSP API calling mode.

Return values

API	calling mode

This function gets FSP API calling mode.

Return values

API	calling index

Definition at line 451 of file FspCommonLib.c.

Here is the call graph for this function:

7.13.2.3 UINT32 GetFspApiParameter (VOID)

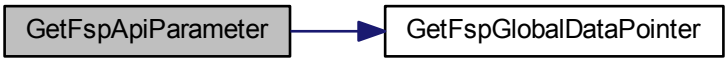
This function gets back the FSP API first parameter passed by the bootlaoder.

Return values

ApiParameter	FSP API first parameter passed by the bootlaoder.

Definition at line 96 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.4 UINT32 GetFspApiParameter2 (VOID)

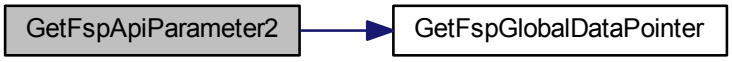
This function gets back the FSP API second parameter passed by the bootlaoder.

Return values

ApiParameter	FSP API second parameter passed by the bootlaoder.

Definition at line 113 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.5 VOID* GetFspCfgRegionDataPointer (VOID)

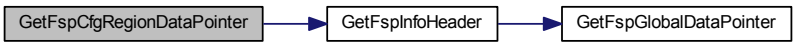
This function gets the CfgRegion data pointer.

Returns

CfgRegion data pointer.

Definition at line 434 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.6 FSP_INFO_HEADER* GetFspInfoHeader (VOID)

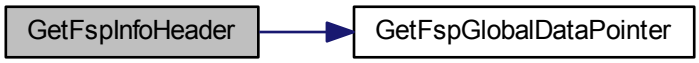
This function gets the FSP info header pointer.

Return values

FspInfoHeader	FSP info header pointer

Definition at line 389 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.7 FSP_INFO_HEADER* GetFspInfoHeaderFromApiContext (VOID)

This function gets the FSP info header pointer from the API context.

Return values

FspInfoHeader	FSP info header pointer

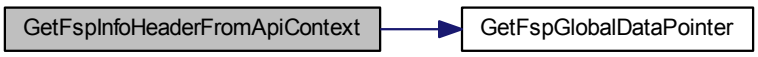
This function gets the FSP info header pointer from the API context.

Return values

FspInfoHeader	FSP info header pointer using the API stack context

Definition at line 417 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.8 VOID* GetFspMemoryInitUpdDataPointer (VOID)

This function gets the memory init UPD data pointer.

Returns

memory init UPD data pointer.

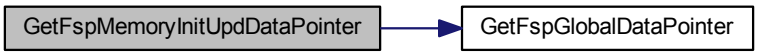
This function gets the memory init UPD data pointer.

Returns

FspMemoryInit UPD data pointer.

Definition at line 301 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.9 VOID* GetFspPlatformDataPointer (VOID)

This function gets the platform specific data pointer.

Parameters

in	*PlatformData*	Fsp platform specific data pointer.
in	*PlatformData*	FSP platform specific data pointer.

Definition at line 217 of file FspCommonLib.c.

Here is the call graph for this function:

7.13.2.10 VOID* GetFspSiliconInitUpdDataPointer (VOID)

This function gets the silicon init UPD data pointer.

Returns

silicon init UPD data pointer.

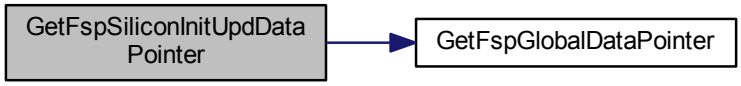
This function gets the silicon init UPD data pointer.

Returns

FspSiliconInit UPD data pointer.

Definition at line 343 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.11 VOID* GetFspUpdDataPointer (VOID)

This function gets the UPD data pointer.

Returns

UpdDataPtr UPD data pointer.

Definition at line 259 of file FspCommonLib.c.

Here is the call graph for this function:

7.13.2.12 UINT32 GetPhaseStatusCode (VOID)

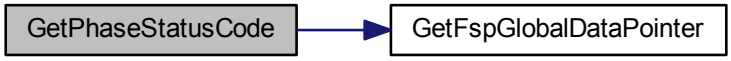
This function gets FSP Phase StatusCode.

Return values

StatusCode	

Definition at line 482 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.13 VOID SetFspApiCallingIndex (UINT8 *Index*)

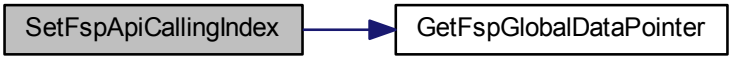
This function sets FSP API calling mode.

Parameters

in	*Index*	API calling index

Definition at line 465 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.14 VOID SetFspApiParameter (IN UINT32 *Value*)

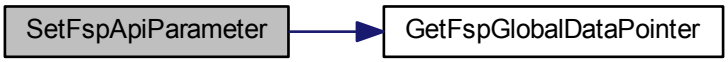
This function sets the FSP API parameter in the stack.

Parameters

in	*Value*	New parameter value.

Definition at line 131 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.15 VOID SetFspApiReturnStatus (IN UINT32 *ReturnStatus*)

This function set the API status code returned to the BootLoader.

Parameters

in	*ReturnStatus*	Status code to return.

Definition at line 149 of file FspCommonLib.c.

Here is the call graph for this function:

7.13.2.16 VOID SetFspCoreStackPointer (IN VOID * *NewStackTop*)

This function sets the context switching stack to a new stack frame.

Parameters

in	*NewStackTop*	New core stack to be set.

Definition at line 167 of file FspCommonLib.c.

Here is the call graph for this function:

7.13.2.17 VOID SetFspGlobalDataPointer (IN FSP_GLOBAL_DATA * *FspData*)

This function sets the FSP global data pointer.

Parameters

in	*FspData*	Fsp global data pointer.
in	*FspData*	FSP global data pointer.

Definition at line 65 of file FspCommonLib.c.

7.13.2.18 VOID SetFspInfoHeader (FSP_INFO_HEADER * *FspInfoHeader*)

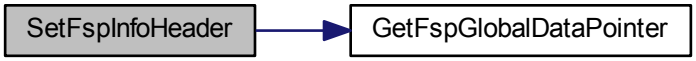
This function sets the FSP info header pointer.

Parameters

in	*FspInfoHeader*	FSP info header pointer

Definition at line 403 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.19 UINT64 SetFspMeasurePoint (IN UINT8 *Id*)

Set FSP measurement point timestamp.

Parameters

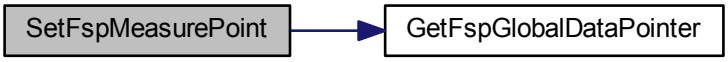
in	*Id*	Measurement point ID.

Returns

performance timestamp.

Definition at line 363 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.20 VOID SetFspMemoryInitUpdDataPointer (IN VOID * *MemoryInitUpdPtr*)

This function sets the memory init UPD data pointer.

Parameters

in	*MemoryInit↩ UpdPtr*	memory init UPD data pointer.

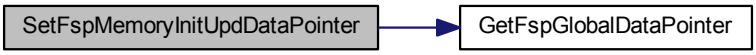
This function sets the memory init UPD data pointer.

Parameters

in	*MemoryInit↩ UpdPtr*	FspMemoryInit UPD data pointer.

Definition at line 277 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.21 VOID SetFspPlatformDataPointer (IN VOID * *PlatformData*)

This function sets the platform specific data pointer.

Parameters

in	*PlatformData*	Fsp platform specific data pointer.
in	*PlatformData*	FSP platform specific data pointer.

Definition at line 198 of file FspCommonLib.c.

Here is the call graph for this function:

7.13.2.22 VOID SetFspSiliconInitUpdDataPointer (IN VOID * *SiliconInitUpdPtr*)

This function sets the silicon init UPD data pointer.

Parameters

in	*SiliconInitUpdPtr*	silicon init UPD data pointer.

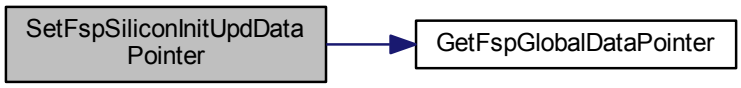
This function sets the silicon init UPD data pointer.

Parameters

in	*SiliconInitUpdPtr*	FspSiliconInit UPD data pointer.

Definition at line 319 of file FspCommonLib.c.

Here is the call graph for this function:



7.13.2.23 VOID SetFspUpdDataPointer (IN VOID * *UpdDataPtr*)

This function sets the UPD data pointer.

Parameters

in	*UpdDataPtr*	UPD data pointer.

Definition at line 235 of file FspCommonLib.c.

Here is the call graph for this function:

7.13.2.24 VOID SetPhaseStatusCode (UINT32 *StatusCode*)

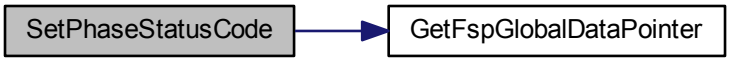
This function sets FSP Phase StatusCode.

Parameters

in	*Mode*	Phase StatusCode

Definition at line 496 of file FspCommonLib.c.

Here is the call graph for this function:

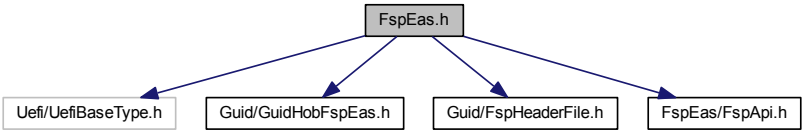


7.14 FspEas.h File Reference

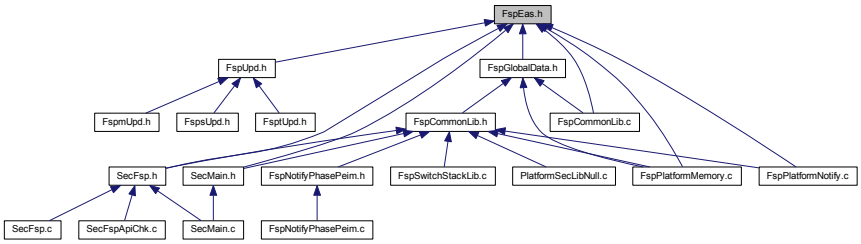
Intel FSP definition from Intel Firmware Support Package External Architecture Specification v2.0.

```
#include <Uefi/UefiBaseType.h>
#include <Guid/GuidHobFspEas.h>
#include <Guid/FspHeaderFile.h>
#include <FspEas/FspApi.h>
```

Include dependency graph for FspEas.h:



This graph shows which files directly or indirectly include this file:



7.14.1 Detailed Description

Intel FSP definition from Intel Firmware Support Package External Architecture Specification v2.0.

Copyright (c) 2014 - 2016, 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.

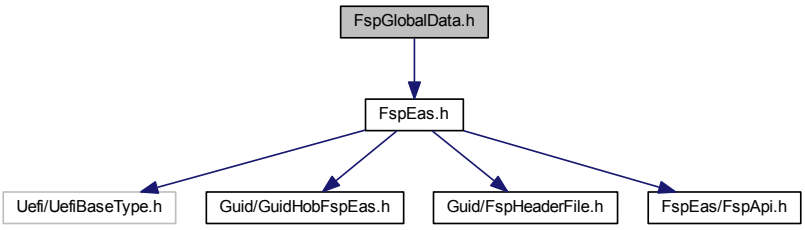
THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

7.15 FspGlobalData.h File Reference

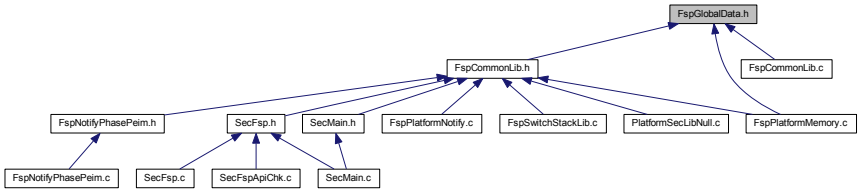
Copyright (c) 2014 - 2016, Intel Corporation.

```
#include <FspEas.h>
```

Include dependency graph for FspGlobalData.h:



This graph shows which files directly or indirectly include this file:



7.15.1 Detailed Description

Copyright (c) 2014 - 2016, Intel Corporation.

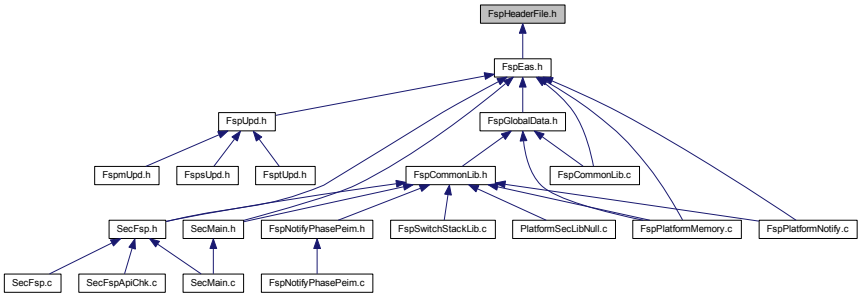
All rights reserved.
This program and the accompanying materials are licensed and made available under the terms and conditions of the BSD License which accompanies this distribution. The full text of the license may be found at http://opensource.org/licenses/bsd-license.php.

THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

7.16 FspHeaderFile.h File Reference

Intel FSP Header File definition from Intel Firmware Support Package External Architecture Specification v2.0.

This graph shows which files directly or indirectly include this file:



Classes

- struct FSP_INFO_HEADER

  FSP Information Header as described in FSP v2.0 Spec section 5.1.1.

- struct FSP_INFO_EXTENDED_HEADER

  FSP Information Extended Header as described in FSP v2.0 Spec section 5.1.2.

- struct FSP_PATCH_TABLE

  FSP Patch Table as described in FSP v2.0 Spec section 5.1.5.

Macros

- #define FSP_INFO_HEADER_OFF 0x94

  Fixed FSP header offset in the FSP image.

- #define FSP_INFO_EXTENDED_HEADER_SIGNATURE SIGNATURE_32 ('F', 'S', 'P', 'E')

  Signature of the FSP Extended Header.

7.16.1 Detailed Description

Intel FSP Header File definition from Intel Firmware Support Package External Architecture Specification v2.0.

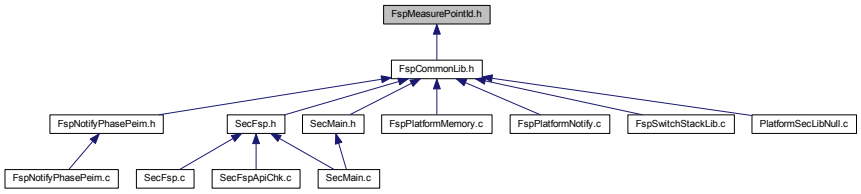
Copyright (c) 2014 - 2016, Intel Corporation. All rights reserved.
This program and the accompanying materials are licensed and made available under the terms and conditions of the BSD License which accompanies this distribution. The full text of the license may be found at http://opensource.org/licenses/bsd-license.php.

THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

7.17 FspMeasurePointId.h File Reference

Copyright (c) 2014 - 2016, Intel Corporation.

This graph shows which files directly or indirectly include this file:



7.17.1 Detailed Description

Copyright (c) 2014 - 2016, Intel Corporation.

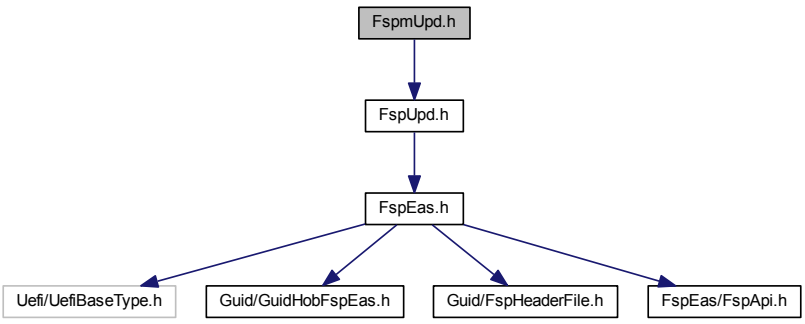
All rights reserved.
This program and the accompanying materials are licensed and made available under the terms and conditions of the BSD License which accompanies this distribution. The full text of the license may be found at http://opensource.org/licenses/bsd-license.php.

THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

7.18 FspmUpd.h File Reference

Copyright (c) 2020, Intel Corporation.

```
#include <FspUpd.h>
```
Include dependency graph for FspmUpd.h:



Classes

- struct FSP_M_CONFIG

  Fsp M Configuration.

- struct FSPM_UPD

  Fsp M UPD Configuration.

7.18.1 Detailed Description

Copyright (c) 2020, 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 SH↩ ALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCURE↩ MENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INT↩ ERRUPTION) 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.

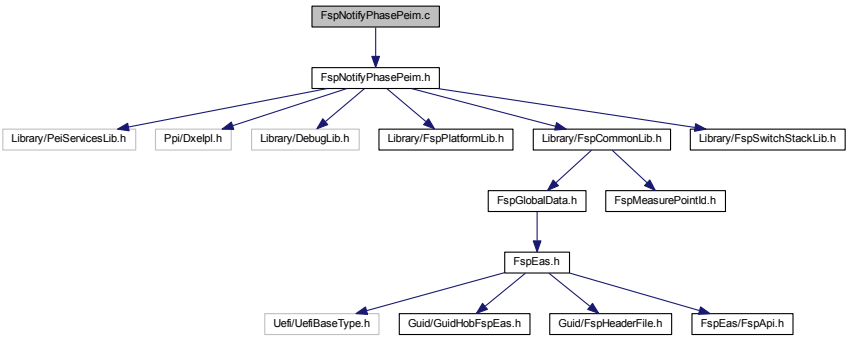
This file is automatically generated. Please do NOT modify !!!

7.19 FspNotifyPhasePeim.c File Reference

Source file for FSP notify phase PEI module.

```
#include "FspNotifyPhasePeim.h"
```

Include dependency graph for FspNotifyPhasePeim.c:



Functions

- EFI_STATUS WaitForNotify (IN CONST EFI_DXE_IPL_PPI ∗This, IN EFI_PEI_SERVICES ∗∗PeiServices, IN EFI_PEI_HOB_POINTERS HobList)

  This function waits for FSP notify.

- EFI_STATUS FspNotifyPhasePeimEntryPoint (IN EFI_PEI_FILE_HANDLE FileHandle, IN CONST EFI_P↩ EI_SERVICES ∗∗PeiServices)

  FSP notify phase PEI module entry point.

7.19.1 Detailed Description

Source file for FSP notify phase PEI module.

Copyright (c) 2016, 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.

7.19.2 Function Documentation

7.19.2.1 EFI_STATUS FspNotifyPhasePeimEntryPoint (IN EFI_PEI_FILE_HANDLE *FileHandle*, IN CONST EFI_PEI_SERVICES ** *PeiServices*)

FSP notify phase PEI module entry point.

Parameters

in	*FileHandle*	Not used.
in	*PeiServices*	General purpose services available to every PEIM.

Return values

EFI_SUCCESS	The function completes successfully
EFI_OUT_OF_RESOURCES	Insufficient resources to create database

Definition at line 113 of file FspNotifyPhasePeim.c.

7.19.2.2 EFI_STATUS WaitForNotify (IN CONST EFI_DXE_IPL_PPI * *This*, IN EFI_PEI_SERVICES ** *PeiServices*, IN EFI_PEI_HOB_POINTERS *HobList*)

This function waits for FSP notify.

Parameters

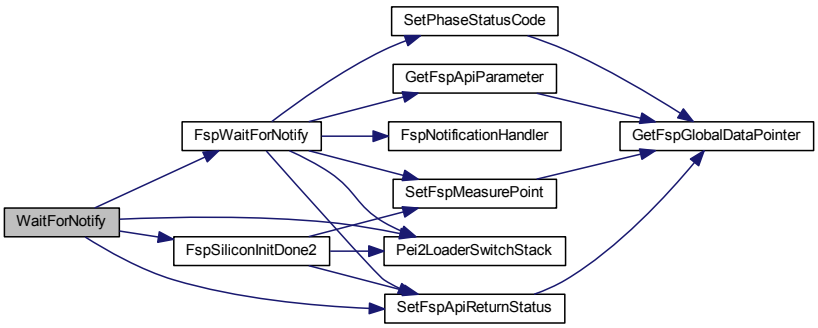
This	Entry point for DXE IPL PPI.
PeiServices	General purpose services available to every PEIM.
HobList	Address to the Pei HOB list.

Returns

EFI_SUCCESS This function never returns.

Definition at line 64 of file FspNotifyPhasePeim.c.

Here is the call graph for this function:

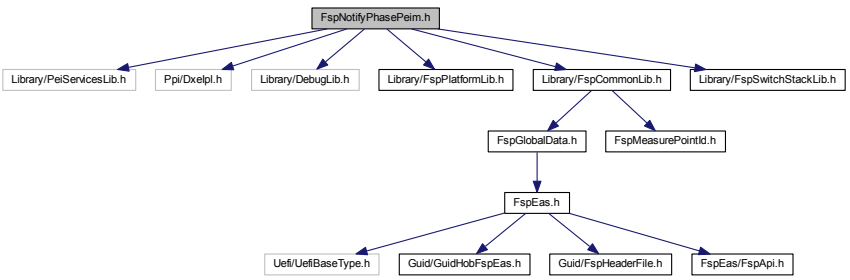


7.20 FspNotifyPhasePeim.h File Reference

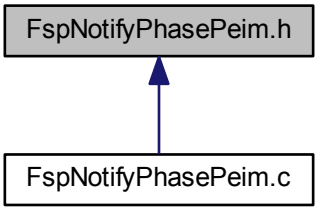
Header file for FSP notify phase PEI module.

```
#include <Library/PeiServicesLib.h>
#include <Ppi/DxeIpl.h>
#include <Library/DebugLib.h>
#include <Library/FspPlatformLib.h>
#include <Library/FspCommonLib.h>
#include <Library/FspSwitchStackLib.h>
```

Include dependency graph for FspNotifyPhasePeim.h:



This graph shows which files directly or indirectly include this file:



7.20.1 Detailed Description

Header file for FSP notify phase PEI module.

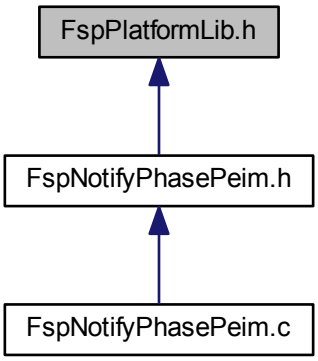
Copyright (c) 2016 Intel Corporation. All rights reserved. This program and the accompanying materials are licensed and made available under the terms and conditions of the BSD License which accompanies this distribution. The full text of the license may be found at http://opensource.org/licenses/bsd-license.php.

THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

7.21 FspPlatformLib.h File Reference

Copyright (c) 2014 - 2016, Intel Corporation.

This graph shows which files directly or indirectly include this file:



Functions

- EFI_HOB_RESOURCE_DESCRIPTOR * FspGetResourceDescriptorByOwner (IN EFI_GUID *OwnerGuid)

Get system memory resource descriptor by owner.

- VOID FspGetSystemMemorySize (IN OUT UINT64 *LowMemoryLength, IN OUT UINT64 *HighMemoryLength)

  Get system memory from HOB.
- VOID FspSetNewStackFrame (VOID)

  Set a new stack frame for the continuation function.
- VOID FspSiliconInitDone (VOID)

  This function transfer control back to BootLoader after FspSiliconInit.
- VOID FspMemoryInitDone (IN OUT VOID **HobListPtr)

  This function returns control to BootLoader after MemoryInitApi.
- VOID FspTempRamExitDone (VOID)

  This function returns control to BootLoader after TempRamExitApi.
- VOID FspWaitForNotify (VOID)

  This function handle NotifyPhase API call from the BootLoader.
- VOID FspSiliconInitDone2 (IN EFI_STATUS Status)

  This function transfer control back to BootLoader after FspSiliconInit.
- VOID FspMemoryInitDone2 (IN EFI_STATUS Status, IN OUT VOID **HobListPtr)

  This function returns control to BootLoader after MemoryInitApi.
- VOID FspTempRamExitDone2 (IN EFI_STATUS Status)

  This function returns control to BootLoader after TempRamExitApi.

7.21.1 Detailed Description

Copyright (c) 2014 - 2016, 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.

7.21.2 Function Documentation

7.21.2.1 EFI_HOB_RESOURCE_DESCRIPTOR* FspGetResourceDescriptorByOwner (IN EFI_GUID * *OwnerGuid*)

Get system memory resource descriptor by owner.

Parameters

in	*OwnerGuid*	resource owner guid

Definition at line 33 of file FspPlatformMemory.c.

7.21.2.2 VOID FspGetSystemMemorySize (IN OUT UINT64 * *LowMemoryLength*, IN OUT UINT64 * *HighMemoryLength*)

Get system memory from HOB.

Parameters

in,out	*LowMemoryLength*	less than 4G memory length

in,out	*HighMemory↩ Length*	greater than 4G memory length

Definition at line 68 of file FspPlatformMemory.c.

7.21.2.3 VOID FspMemoryInitDone (IN OUT VOID ∗∗ *HobListPtr*)

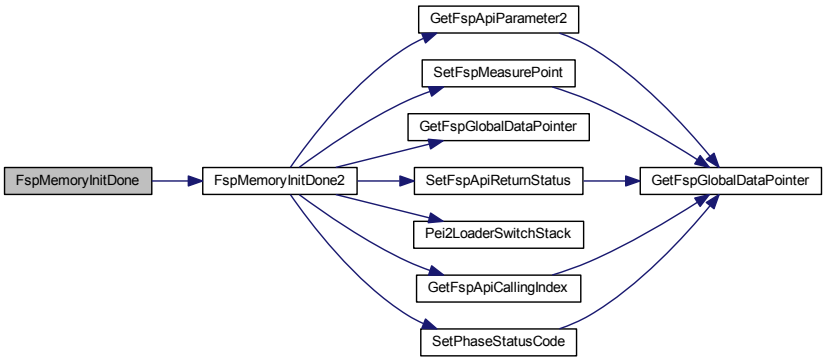
This function returns control to BootLoader after MemoryInitApi.

Parameters

in,out	*HobListPtr*	The address of HobList pointer.

Definition at line 379 of file FspPlatformNotify.c.

Here is the call graph for this function:



7.21.2.4 VOID FspMemoryInitDone2 (IN EFI_STATUS *Status,* IN OUT VOID ∗∗ *HobListPtr*)

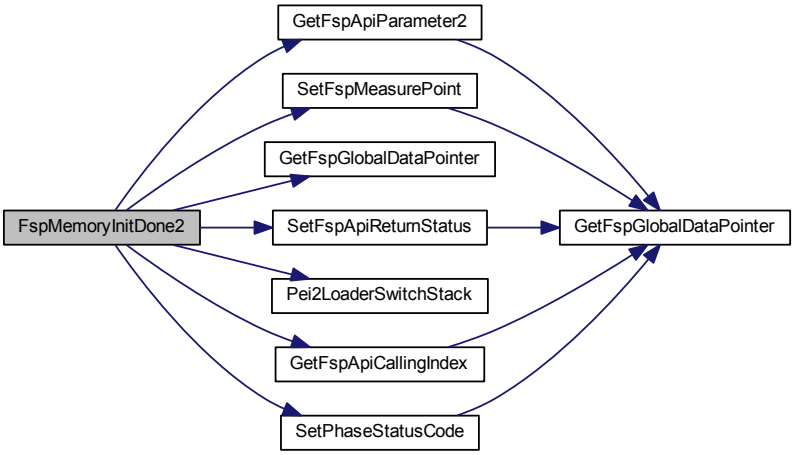
This function returns control to BootLoader after MemoryInitApi.

Parameters

in	*Status*	return status for the MemoryInitApi.
in,out	*HobListPtr*	The address of HobList pointer.
in	*Status*	return status for the MemoryInitApi.
in,out	*HobListPtr*	The address of HobList pointer, if NULL, will get value from GetFspApi↩ Parameter2 ()

Definition at line 159 of file FspPlatformNotify.c.

Here is the call graph for this function:



7.21.2.5 VOID FspSiliconInitDone2 (IN EFI_STATUS *Status*)

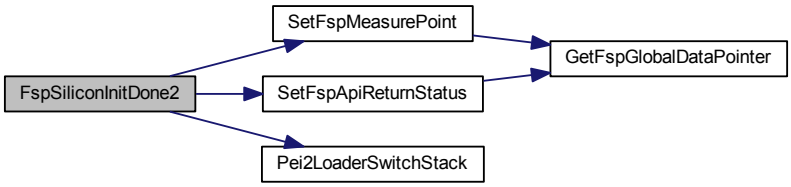
This function transfer control back to BootLoader after FspSiliconInit.

Parameters

in	*Status*	return status for the FspSiliconInit.

Definition at line 116 of file FspPlatformNotify.c.

Here is the call graph for this function:



7.21.2.6 VOID FspTempRamExitDone2 (IN EFI_STATUS *Status*)

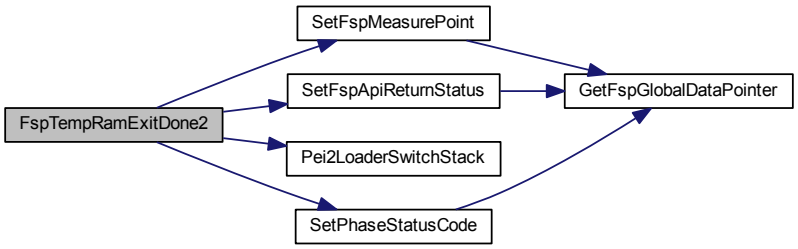
This function returns control to BootLoader after TempRamExitApi.

Parameters

in	*Status*	return status for the TempRamExitApi.

Definition at line 236 of file FspPlatformNotify.c.

Here is the call graph for this function:



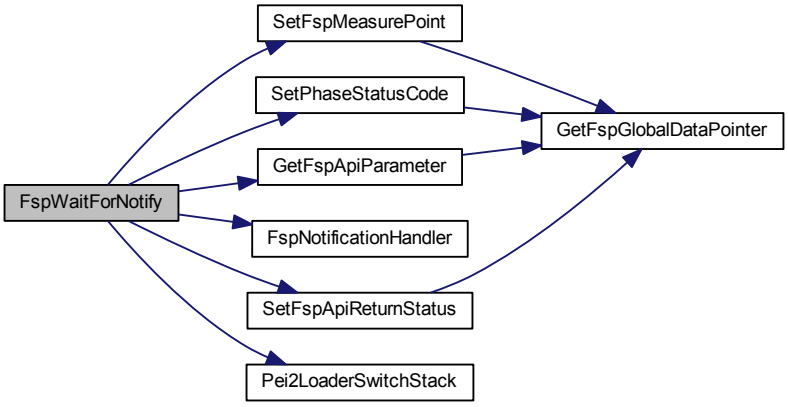
7.21.2.7 VOID FspWaitForNotify (VOID)

This function handle NotifyPhase API call from the BootLoader.

It gives control back to the BootLoader after it is handled. If the Notification code is a ReadyToBoot event, this function will return and FSP continues the remaining execution until it reaches the DxeIpl.

Definition at line 284 of file FspPlatformNotify.c.

Here is the call graph for this function:

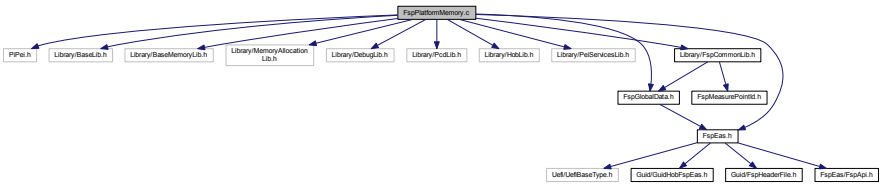


7.22 FspPlatformMemory.c File Reference

Copyright (c) 2014 - 2016, Intel Corporation.

```
#include <PiPei.h>
#include <Library/BaseLib.h>
#include <Library/BaseMemoryLib.h>
#include <Library/MemoryAllocationLib.h>
#include <Library/DebugLib.h>
#include <Library/PcdLib.h>
#include <Library/HobLib.h>
#include <Library/PeiServicesLib.h>
#include <Library/FspCommonLib.h>
#include <FspGlobalData.h>
#include <FspEas.h>
```

Include dependency graph for FspPlatformMemory.c:



Functions

- EFI_HOB_RESOURCE_DESCRIPTOR * FspGetResourceDescriptorByOwner (IN EFI_GUID *OwnerGuid)

  Get system memory resource descriptor by owner.
- VOID FspGetSystemMemorySize (IN OUT UINT64 *LowMemoryLength, IN OUT UINT64 *HighMemoryLength)

  Get system memory from HOB.

7.22.1 Detailed Description

Copyright (c) 2014 - 2016, 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.

7.22.2 Function Documentation

7.22.2.1 EFI_HOB_RESOURCE_DESCRIPTOR* FspGetResourceDescriptorByOwner (IN EFI_GUID * *OwnerGuid*)

Get system memory resource descriptor by owner.

Parameters

in	*OwnerGuid*	resource owner guid

Definition at line 33 of file FspPlatformMemory.c.

7.22.2.2 VOID FspGetSystemMemorySize (IN OUT UINT64 * *LowMemoryLength*, IN OUT UINT64 * *HighMemoryLength*)

Get system memory from HOB.

Parameters

in,out	*LowMemory↩ Length*	less than 4G memory length
in,out	*HighMemory↩ Length*	greater than 4G memory length

Definition at line 68 of file FspPlatformMemory.c.

7.23 FspPlatformNotify.c File Reference

Copyright (c) 2014 - 2016, Intel Corporation.

```
#include <PiPei.h>
#include <Library/PeiServicesLib.h>
#include <Library/PeiServicesTablePointerLib.h>
#include <Library/BaseLib.h>
#include <Library/BaseMemoryLib.h>
#include <Library/PcdLib.h>
#include <Library/DebugLib.h>
#include <Library/HobLib.h>
#include <Library/FspSwitchStackLib.h>
#include <Library/FspCommonLib.h>
#include <Guid/EventGroup.h>
#include <FspEas.h>
#include <FspStatusCode.h>
#include <Protocol/PciEnumerationComplete.h>
#include <Library/ReportStatusCodeLib.h>
#include <Library/PerformanceLib.h>
```

Include dependency graph for FspPlatformNotify.c:

Functions

- EFI_STATUS FspNotificationHandler (IN UINT32 NotificationCode)

  Install FSP notification.
- VOID FspSiliconInitDone2 (IN EFI_STATUS Status)

  This function transfer control back to BootLoader after FspSiliconInit.
- VOID FspMemoryInitDone2 (IN EFI_STATUS Status, IN OUT VOID **HobListPtr)

  This function returns control to BootLoader after MemoryInitApi.
- VOID FspTempRamExitDone2 (IN EFI_STATUS Status)

  This function returns control to BootLoader after TempRamExitApi.
- VOID FspWaitForNotify (VOID)

  This function handle NotifyPhase API call from the BootLoader.
- VOID FspSiliconInitDone (VOID)

  This function transfer control back to BootLoader after FspSiliconInit.
- VOID FspMemoryInitDone (IN OUT VOID **HobListPtr)

  This function returns control to BootLoader after MemoryInitApi.
- VOID FspTempRamExitDone (VOID)

  This function returns control to BootLoader after TempRamExitApi.

7.23.1 Detailed Description

Copyright (c) 2014 - 2016, 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.

7.23.2 Function Documentation

7.23.2.1 VOID FspMemoryInitDone (IN OUT VOID ** *HobListPtr*)

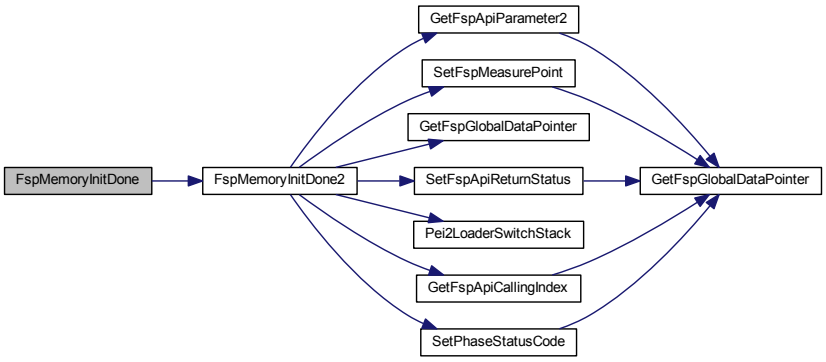
This function returns control to BootLoader after MemoryInitApi.

Parameters

in,out	*HobListPtr*	The address of HobList pointer.

Definition at line 379 of file FspPlatformNotify.c.

Here is the call graph for this function:



7.23.2.2 VOID FspMemoryInitDone2 (IN EFI_STATUS *Status*, IN OUT VOID ** *HobListPtr*)

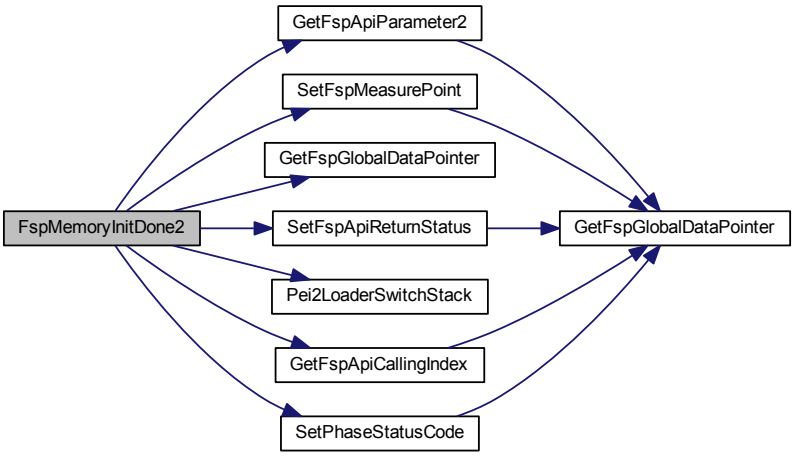
This function returns control to BootLoader after MemoryInitApi.

Parameters

in	*Status*	return status for the MemoryInitApi.
in,out	*HobListPtr*	The address of HobList pointer, if NULL, will get value from GetFspApiParameter2 ()

Definition at line 159 of file FspPlatformNotify.c.

Here is the call graph for this function:



7.23.2.3 EFI_STATUS FspNotificationHandler (IN UINT32 *NotificationCode*)

Install FSP notification.

Parameters

in	*NotificationCode*	FSP notification code

Return values

EFI_SUCCESS	Notify FSP successfully
EFI_INVALID_PARAMET↩ ER	NotificationCode is invalid

Definition at line 67 of file FspPlatformNotify.c.

7.23.2.4 VOID FspSiliconInitDone2 (IN EFI_STATUS *Status*)

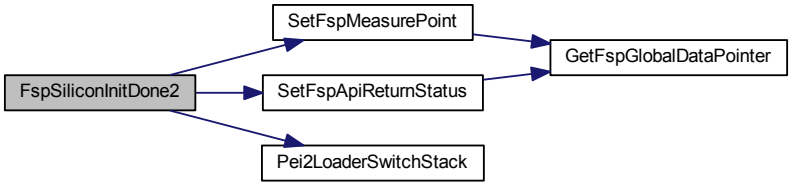
This function transfer control back to BootLoader after FspSiliconInit.

Parameters

in	*Status*	return status for the FspSiliconInit.

Definition at line 116 of file FspPlatformNotify.c.

Here is the call graph for this function:



7.23.2.5 VOID FspTempRamExitDone2 (IN EFI_STATUS *Status*)

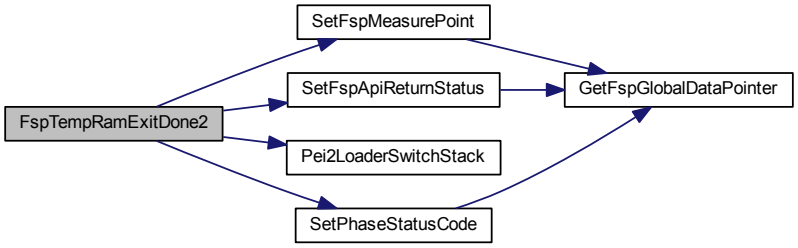
This function returns control to BootLoader after TempRamExitApi.

Parameters

in	*Status*	return status for the TempRamExitApi.

Definition at line 236 of file FspPlatformNotify.c.

Here is the call graph for this function:



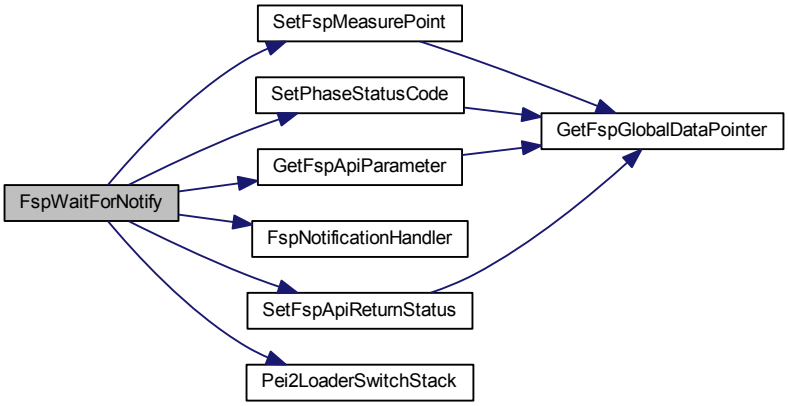
7.23.2.6 VOID FspWaitForNotify (VOID)

This function handle NotifyPhase API call from the BootLoader.

It gives control back to the BootLoader after it is handled. If the Notification code is a ReadyToBoot event, this function will return and FSP continues the remaining execution until it reaches the DxeIpl.

Definition at line 284 of file FspPlatformNotify.c.

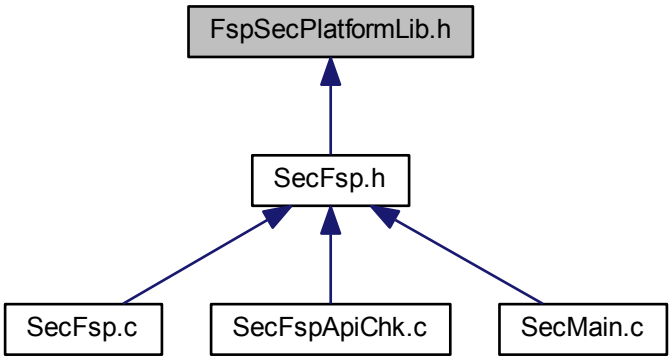
Here is the call graph for this function:



7.24 FspSecPlatformLib.h File Reference

Copyright (c) 2015 - 2016, Intel Corporation.

This graph shows which files directly or indirectly include this file:



Functions

- UINT32 SecPlatformInit (VOID)

  This function performs platform level initialization.
- UINT32 LoadMicrocode (IN VOID *FsptUpdDataPtr)

  This function loads Microcode.
- UINT32 SecCarInit (IN VOID *FsptUpdDataPtr)

  This function initializes the CAR.
- EFI_STATUS FspUpdSignatureCheck (IN UINT32 ApiIdx, IN VOID *ApiParam)

This function check the signture of UPD.

7.24.1 Detailed Description

Copyright (c) 2015 - 2016, 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.

7.24.2 Function Documentation

7.24.2.1 EFI_STATUS FspUpdSignatureCheck (IN UINT32 *ApiIdx*, IN VOID * *ApiParam*)

This function check the signture of UPD.

Parameters

in	*ApiIdx*	Internal index of the FSP API.
in	*ApiParam*	Parameter of the FSP API.

Definition at line 27 of file PlatformSecLibNull.c.

7.24.2.2 UINT32 LoadMicrocode (IN VOID * *FsptUpdDataPtr*)

This function loads Microcode.

This function must be in ASM file, because stack is not established yet. This function is optional. If a library instance does not provide this function, the default one will be used.

The callee should not use XMM6/XMM7. The return address is saved in MM7.

Parameters

in	*FsptUpdDataPtr*	Address pointer to the FSPT_UPD data structure. It is saved in ESP.

Return values

in	saved in EAX - 0 means Microcode is loaded successfully. other means Microcode is not loaded successfully.

7.24.2.3 UINT32 SecCarInit (IN VOID * *FsptUpdDataPtr*)

This function initializes the CAR.

This function must be in ASM file, because stack is not established yet.

The callee should not use XMM6/XMM7. The return address is saved in MM7.

Parameters

in	*FsptUpdDataPtr*	Address pointer to the FSPT_UPD data structure. It is saved in ESP.

Return values

in	saved in EAX - 0 means CAR initialization success. other means CAR initialization fail.

7.24.2.4 UINT32 SecPlatformInit (VOID)

This function performs platform level initialization.

This function must be in ASM file, because stack is not established yet. This function is optional. If a library instance does not provide this function, the default empty one will be used.

The callee should not use XMM6/XMM7. The return address is saved in MM7.

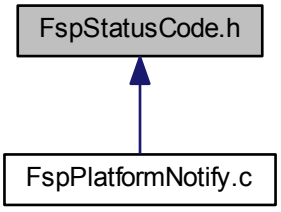
Return values

in	saved in EAX - 0 means platform initialization success. other means platform initialization fail.

7.25 FspStatusCode.h File Reference

Intel FSP status code definition.

This graph shows which files directly or indirectly include this file:



7.25.1 Detailed Description

Intel FSP status code definition.

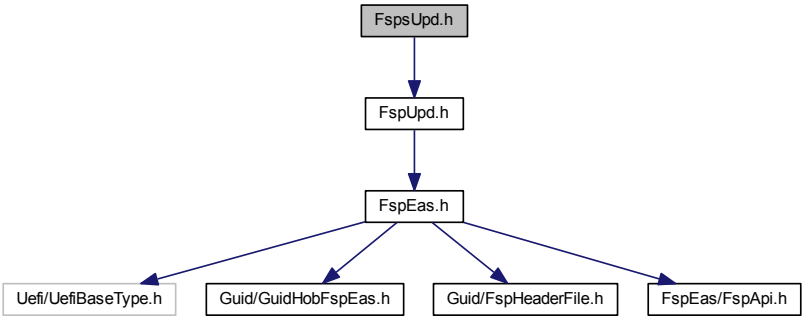
Copyright (c) 2016, Intel Corporation. All rights reserved.
This program and the accompanying materials are licensed and made available under the terms and conditions of the BSD License which accompanies this distribution. The full text of the license may be found at http://opensource.org/licenses/bsd-license.php.

THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

7.26 FspsUpd.h File Reference

Copyright (c) 2020, Intel Corporation.

`#include <FspUpd.h>`
Include dependency graph for FspsUpd.h:



Classes

- struct FSP_S_CONFIG

  Fsp S Configuration.
- struct FSPS_UPD

  Fsp S UPD Configuration.

7.26.1 Detailed Description

Copyright (c) 2020, 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 SH↩ ALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCURE↩ MENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INT↩ ERRUPTION) 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.

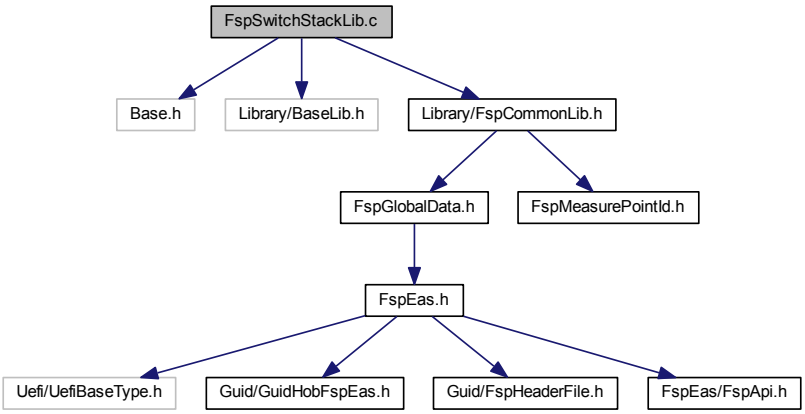
This file is automatically generated. Please do NOT modify !!!

7.27 FspSwitchStackLib.c File Reference

Copyright (c) 2014, Intel Corporation.

```
#include <Base.h>
#include <Library/BaseLib.h>
#include <Library/FspCommonLib.h>
```

Include dependency graph for FspSwitchStackLib.c:



Functions

- UINT32 SwapStack (IN UINT32 NewStack)

  Switch the current stack to the previous saved stack.

7.27.1 Detailed Description

Copyright (c) 2014, 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.

7.27.2 Function Documentation

7.27.2.1 UINT32 SwapStack (IN UINT32 *NewStack*)

Switch the current stack to the previous saved stack.

Parameters

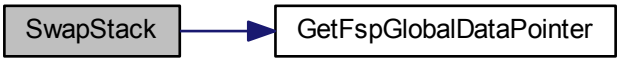
in	*NewStack*	The new stack to be switched.

Returns

OldStack After switching to the saved stack, this value will be saved in eax before returning.

Definition at line 30 of file FspSwitchStackLib.c.

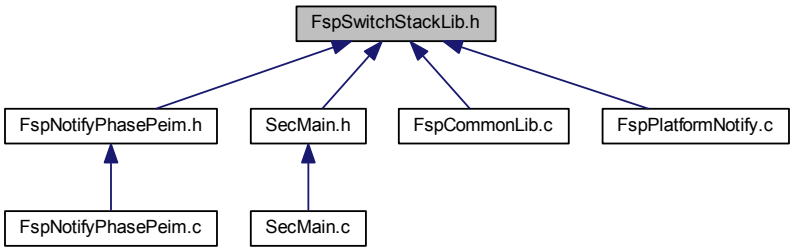
Here is the call graph for this function:



7.28 FspSwitchStackLib.h File Reference

Copyright (c) 2014, Intel Corporation.

This graph shows which files directly or indirectly include this file:



Functions

- UINT32 Pei2LoaderSwitchStack (VOID)

  This function will switch the current stack to the previous saved stack.

7.28.1 Detailed Description

Copyright (c) 2014, 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.

7.28.2 Function Documentation

7.28.2.1 UINT32 Pei2LoaderSwitchStack (VOID)

This function will switch the current stack to the previous saved stack.

Before calling the previous stack has to be set in FSP_GLOBAL_DATA.CoreStack. EIP FLAGS 16 bit FLAGS 16 bit EDI ESI EBP ESP EBX EDX ECX EAX DWORD IDT base1 StackPointer: DWORD IDT base2

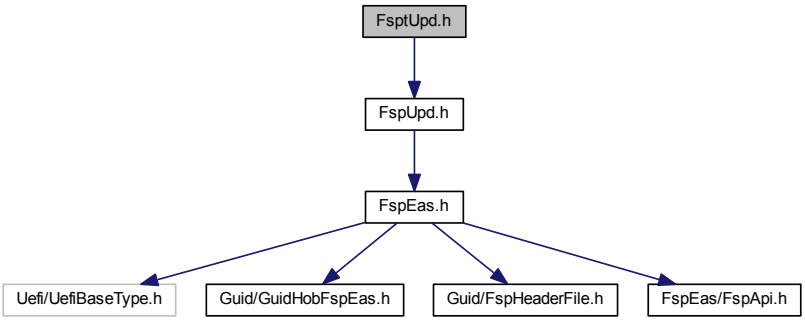
Returns

ReturnKey After switching to the saved stack, this value will be saved in eax before returning.

7.29 FsptUpd.h File Reference

Copyright (c) 2020, Intel Corporation.

`#include <FspUpd.h>`
Include dependency graph for FsptUpd.h:



Classes

- struct FSPT_COMMON_UPD

  Fsp T Common UPD.
- struct FSPT_UPD

  Fsp T UPD Configuration.

7.29.1 Detailed Description

Copyright (c) 2020, 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 SH↩ ALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,

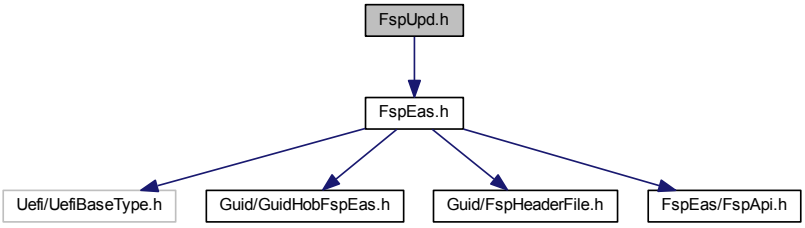
SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCURE↩ MENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INT↩ ERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR 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 !!!

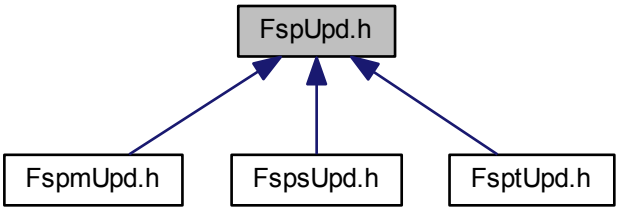
7.30 FspUpd.h File Reference

Copyright (c) 2020, Intel Corporation.

`#include <FspEas.h>`
Include dependency graph for FspUpd.h:



This graph shows which files directly or indirectly include this file:



7.30.1 Detailed Description

Copyright (c) 2020, 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 SH↩ ALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCURE↩ MENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INT↩ ERRUPTION) 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.

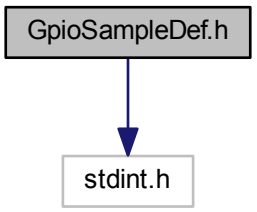
This file is automatically generated. Please do NOT modify !!!

7.31 GpioSampleDef.h File Reference

Copyright (c) 2015, Intel Corporation.

```
#include <stdint.h>
```

Include dependency graph for GpioSampleDef.h:



7.31.1 Detailed Description

Copyright (c) 2015, 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 SH↩ ALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCURE↩ MENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INT↩

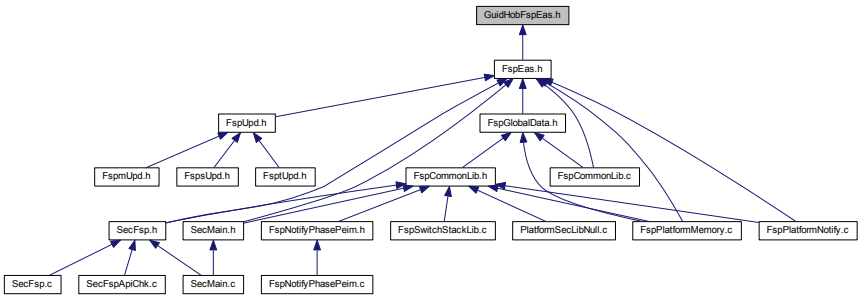
ERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This file is automatically generated. Please do NOT modify !!!

7.32 GuidHobFspEas.h File Reference

Intel FSP Hob Guid definition from Intel Firmware Support Package External Architecture Specification v2.0.

This graph shows which files directly or indirectly include this file:



7.32.1 Detailed Description

Intel FSP Hob Guid definition from Intel Firmware Support Package External Architecture Specification v2.0.

Copyright (c) 2014 - 2016, 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.

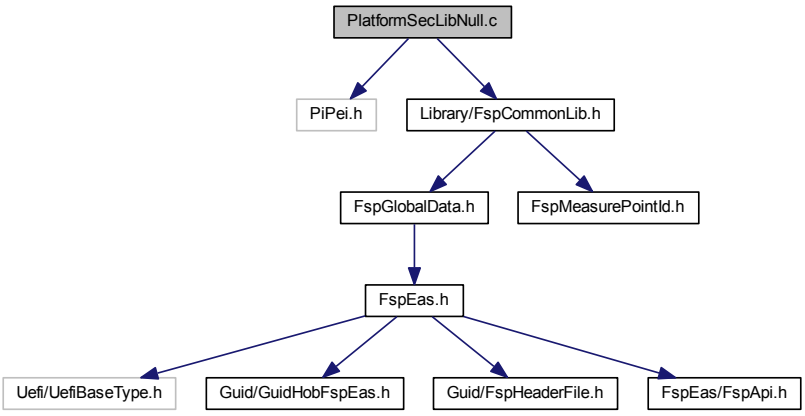
THE PROGRAM IS DISTRIBUTED UNDER THE BSD LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR REPRESENTATIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED.

7.33 PlatformSecLibNull.c File Reference

Null instance of Platform Sec Lib.

```
#include <PiPei.h>
#include <Library/FspCommonLib.h>
```

Include dependency graph for PlatformSecLibNull.c:



Functions

- EFI_STATUS FspUpdSignatureCheck (IN UINT32 ApiIdx, IN VOID *ApiParam)

  This function check the signture of UPD.

7.33.1 Detailed Description

Null instance of Platform Sec Lib.

Copyright (c) 2014 - 2016, 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.

7.33.2 Function Documentation

7.33.2.1 EFI_STATUS FspUpdSignatureCheck (IN UINT32 *ApiIdx*, IN VOID * *ApiParam*)

This function check the signture of UPD.

Parameters

in	*ApiIdx*	Internal index of the FSP API.
in	*ApiParam*	Parameter of the FSP API.

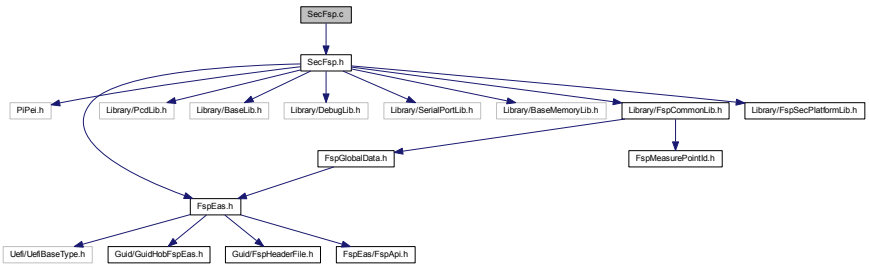
Definition at line 27 of file PlatformSecLibNull.c.

7.34 SecFsp.c File Reference

Copyright (c) 2014 - 2016, Intel Corporation.

```
#include "SecFsp.h"
```

Include dependency graph for SecFsp.c:



Functions

- UINT64 FspGetExceptionHandler (IN UINT64 IdtEntryTemplate)

  Calculate the FSP IDT gate descriptor.

- VOID SecGetPlatformData (IN OUT FSP_GLOBAL_DATA *FspData)

  This interface fills platform specific data.

- VOID FspGlobalDataInit (IN OUT FSP_GLOBAL_DATA *PeiFspData, IN UINT32 BootLoaderStack, IN UINT8 ApiIdx)

  Initialize the FSP global data region.

- VOID FspDataPointerFixUp (IN UINT32 OffsetGap)

  Adjust the FSP data pointers after the stack is migrated to memory.

7.34.1 Detailed Description

Copyright (c) 2014 - 2016, 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.

7.34.2 Function Documentation

7.34.2.1 VOID FspDataPointerFixUp (IN UINT32 *OffsetGap*)

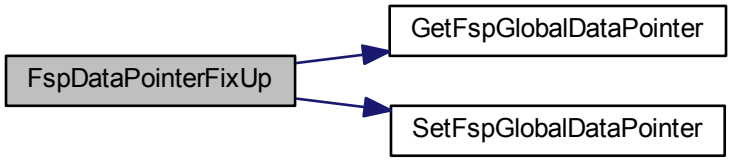
Adjust the FSP data pointers after the stack is migrated to memory.

Parameters

in	*OffsetGap*	The offset gap between the old stack and the new stack.

Definition at line 206 of file SecFsp.c.

Here is the call graph for this function:



7.34.2.2 UINT64 FspGetExceptionHandler (IN UINT64 *IdtEntryTemplate*)

Calculate the FSP IDT gate descriptor.

Parameters

in	*IdtEntryTemplate*	IDT gate descriptor template.

Returns

FSP specific IDT gate descriptor.

Definition at line 26 of file SecFsp.c.

Here is the call graph for this function:

7.34.2.3 VOID FspGlobalDataInit (IN OUT FSP_GLOBAL_DATA * *PeiFspData*, IN UINT32 *BootLoaderStack*, IN UINT8 *ApiIdx*)

Initialize the FSP global data region.

It needs to be done as soon as possible after the stack is setup.

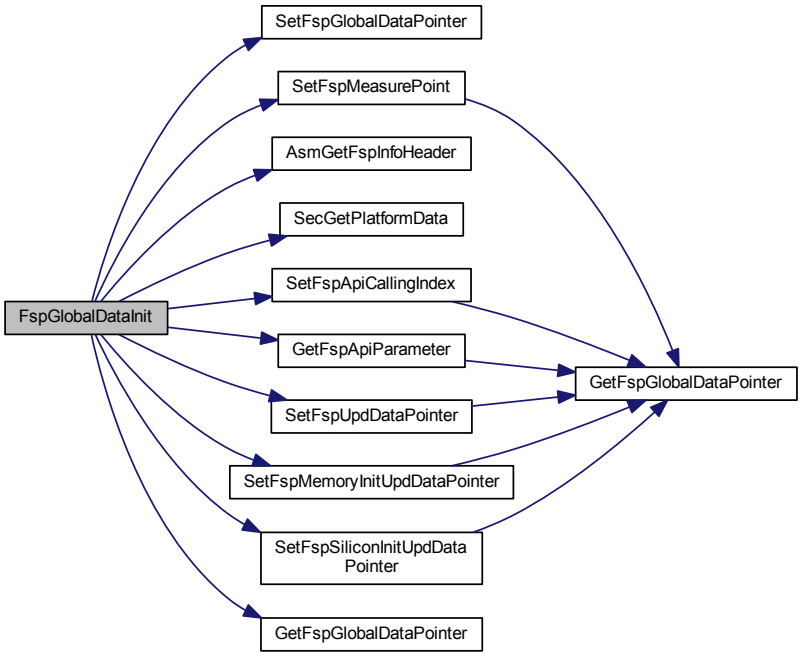
Parameters

in,out	*PeiFspData*	Pointer of the FSP global data.
in	*BootLoaderStack*	BootLoader stack.

in	*ApiIdx*	The index of the FSP API.

Definition at line 122 of file SecFsp.c.

Here is the call graph for this function:



7.34.2.4 VOID SecGetPlatformData (IN OUT FSP_GLOBAL_DATA * *FspData*)

This interface fills platform specific data.

Parameters

in,out	*FspData*	Pointer to the FSP global data.

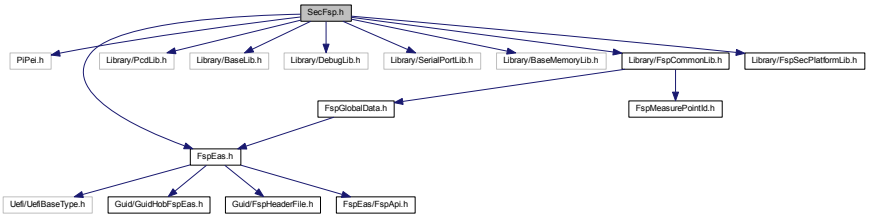
Definition at line 54 of file SecFsp.c.

7.35 SecFsp.h File Reference

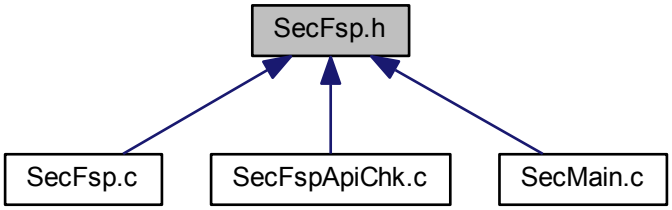
Copyright (c) 2014 - 2016, Intel Corporation.

```
#include <PiPei.h>
#include <FspEas.h>
#include <Library/PcdLib.h>
#include <Library/BaseLib.h>
#include <Library/DebugLib.h>
#include <Library/SerialPortLib.h>
#include <Library/BaseMemoryLib.h>
#include <Library/FspCommonLib.h>
#include <Library/FspSecPlatformLib.h>
```

Include dependency graph for SecFsp.h:



This graph shows which files directly or indirectly include this file:



Functions

- UINT64 FspGetExceptionHandler (IN UINT64 IdtEntryTemplate)

  Calculate the FSP IDT gate descriptor.
- VOID FspGlobalDataInit (IN OUT FSP_GLOBAL_DATA *PeiFspData, IN UINT32 BootLoaderStack, IN UINT8 ApiIdx)

  Initialize the FSP global data region.
- VOID FspDataPointerFixUp (IN UINT32 OffsetGap)

  Adjust the FSP data pointers after the stack is migrated to memory.
- UINT32 AsmGetFspBaseAddress (VOID)

  This interface returns the base address of FSP binary.
- UINT32 AsmGetFspInfoHeader (VOID)

  This interface gets FspInfoHeader pointer.

7.35.1 Detailed Description

Copyright (c) 2014 - 2016, 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.

7.35.2 Function Documentation

7.35.2.1 UINT32 AsmGetFspBaseAddress (VOID)

This interface returns the base address of FSP binary.

Returns

FSP binary base address.

7.35.2.2 UINT32 AsmGetFspInfoHeader (VOID)

This interface gets FspInfoHeader pointer.

Returns

FSP binary base address.

7.35.2.3 VOID FspDataPointerFixUp (IN UINT32 *OffsetGap*)

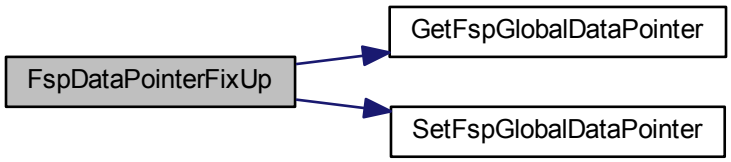
Adjust the FSP data pointers after the stack is migrated to memory.

Parameters

in	*OffsetGap*	The offset gap between the old stack and the new stack.

Definition at line 206 of file SecFsp.c.

Here is the call graph for this function:



7.35.2.4 UINT64 FspGetExceptionHandler (IN UINT64 *IdtEntryTemplate*)

Calculate the FSP IDT gate descriptor.

Parameters

in	*IdtEntryTemplate*	IDT gate descriptor template.

Returns

FSP specific IDT gate descriptor.

Definition at line 26 of file SecFsp.c.

Here is the call graph for this function:

7.35.2.5 VOID FspGlobalDataInit (IN OUT FSP_GLOBAL_DATA * *PeiFspData*, IN UINT32 *BootLoaderStack*, IN UINT8 *ApiIdx*)

Initialize the FSP global data region.

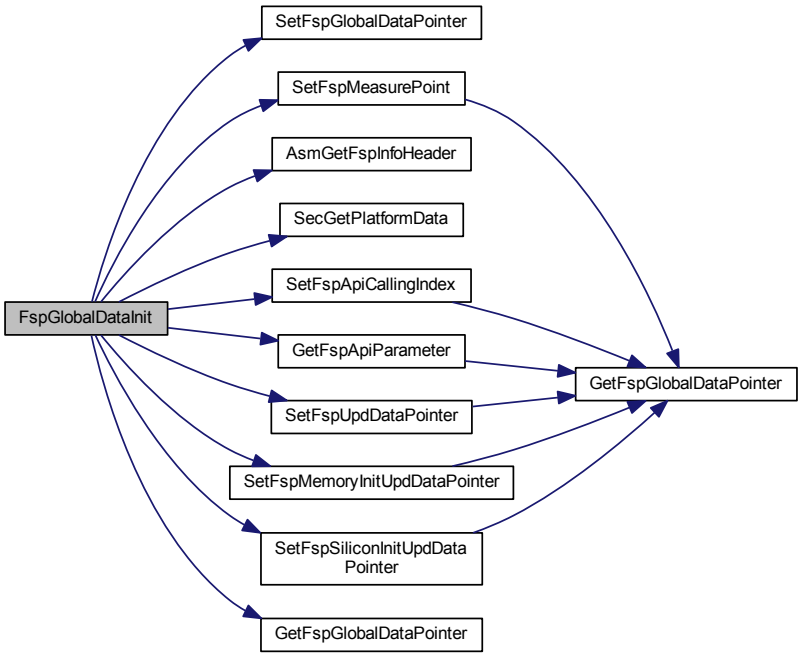
It needs to be done as soon as possible after the stack is setup.

Parameters

in,out	*PeiFspData*	Pointer of the FSP global data.
in	*BootLoader↩ Stack*	BootLoader stack.
in	*ApiIdx*	The index of the FSP API.

Definition at line 122 of file SecFsp.c.

Here is the call graph for this function:

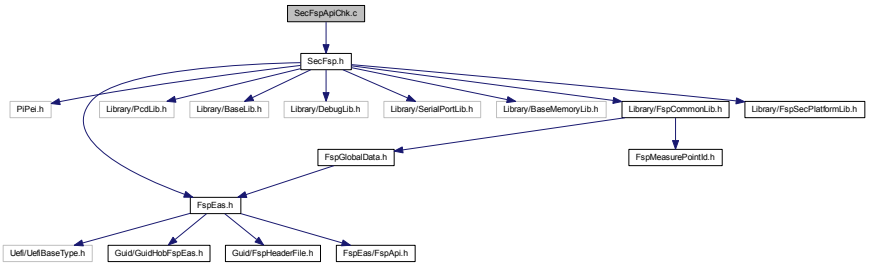


7.36 SecFspApiChk.c File Reference

Copyright (c) 2016, Intel Corporation.

```
#include "SecFsp.h"
```

Include dependency graph for SecFspApiChk.c:



Functions

- EFI_STATUS FspApiCallingCheck (IN UINT8 ApiIdx, IN VOID *ApiParam)

  This function check the FSP API calling condition.

7.36.1 Detailed Description

Copyright (c) 2016, 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.

7.36.2 Function Documentation

7.36.2.1 EFI_STATUS FspApiCallingCheck (IN UINT8 *ApiIdx*, IN VOID * *ApiParam*)

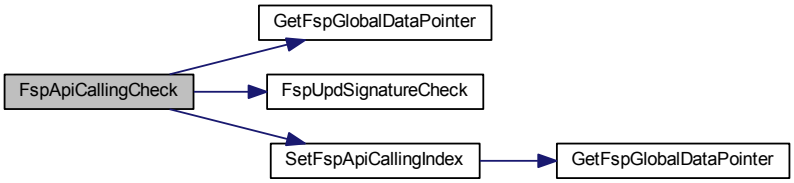
This function check the FSP API calling condition.

Parameters

in	*ApiIdx*	Internal index of the FSP API.
in	*ApiParam*	Parameter of the FSP API.

Definition at line 26 of file SecFspApiChk.c.

Here is the call graph for this function:

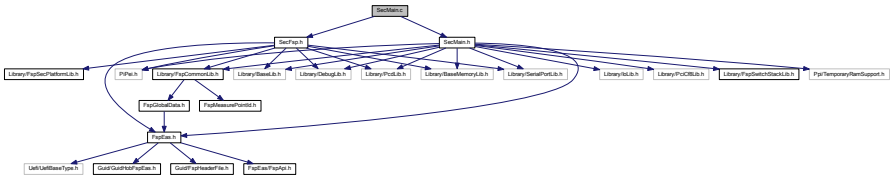


7.37 SecMain.c File Reference

Copyright (c) 2014 - 2016, Intel Corporation.

```
#include "SecMain.h"
#include "SecFsp.h"
```

Include dependency graph for SecMain.c:



Functions

- VOID SecStartup (IN UINT32 SizeOfRam, IN UINT32 TempRamBase, IN VOID *BootFirmwareVolume, IN PEI_CORE_ENTRY PeiCore, IN UINT32 BootLoaderStack, IN UINT32 ApiIdx)

  Entry point to the C language phase of SEC.
- EFI_STATUS SecTemporaryRamSupport (IN CONST EFI_PEI_SERVICES **PeiServices, IN EFI_PHYSICAL_ADDRESS TemporaryMemoryBase, IN EFI_PHYSICAL_ADDRESS PermanentMemoryBase, IN UINTN CopySize)

  This service of the TEMPORARY_RAM_SUPPORT_PPI that migrates temporary RAM into permanent memory.

7.37.1 Detailed Description

Copyright (c) 2014 - 2016, 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.

7.37.2 Function Documentation

7.37.2.1 VOID SecStartup (IN UINT32 *SizeOfRam*, IN UINT32 *TempRamBase*, IN VOID * *BootFirmwareVolume*, IN PEI_CORE_ENTRY *PeiCore*, IN UINT32 *BootLoaderStack*, IN UINT32 *ApiIdx*)

Entry point to the C language phase of SEC.

After the SEC assembly code has initialized some temporary memory and set up the stack, the control is transferred to this function.

Parameters

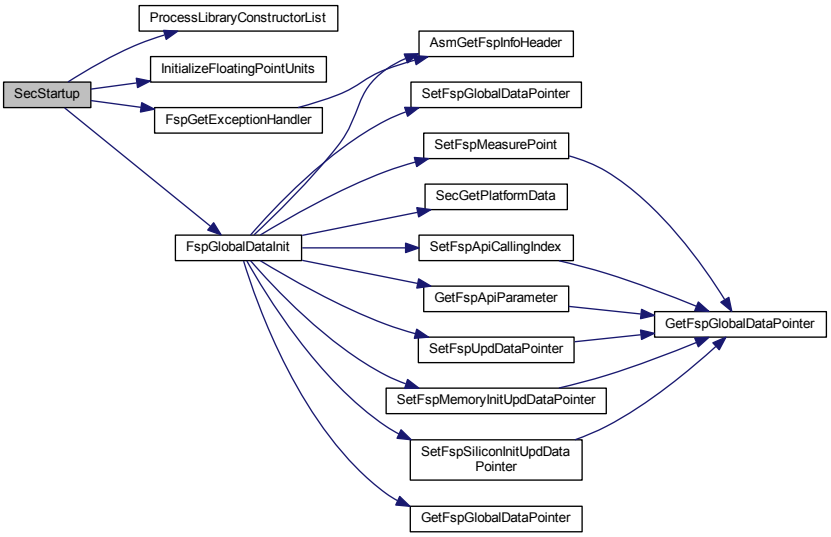
in	*SizeOfRam*	Size of the temporary memory available for use.
in	*TempRamBase*	Base address of temporary ram
in	*BootFirmware↩ Volume*	Base address of the Boot Firmware Volume.
in	*PeiCore*	PeiCore entry point.
in	*BootLoader↩ Stack*	BootLoader stack.
in	*ApiIdx*	the index of API.

Returns

This function never returns.

Definition at line 53 of file SecMain.c.

Here is the call graph for this function:



7.37.2.2 EFI_STATUS SecTemporaryRamSupport (IN CONST EFI_PEI_SERVICES ** *PeiServices*, IN EFI_PHYSICAL_ADDRESS *TemporaryMemoryBase*, IN EFI_PHYSICAL_ADDRESS *PermanentMemoryBase*, IN UINTN *CopySize*)

This service of the TEMPORARY_RAM_SUPPORT_PPI that migrates temporary RAM into permanent memory.

Parameters

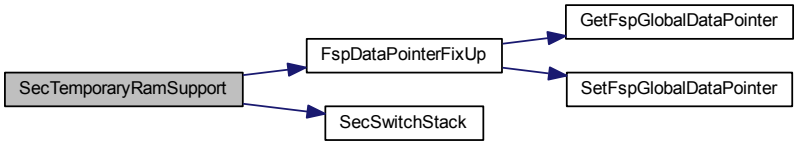
in	*PeiServices*	Pointer to the PEI Services Table.
in	*Temporary↩ MemoryBase*	Source Address in temporary memory from which the SEC or PEIM will copy the Temporary RAM contents.
in	*Permanent↩ MemoryBase*	Destination Address in permanent memory into which the SEC or PEIM will copy the Temporary RAM contents.
in	*CopySize*	Amount of memory to migrate from temporary to permanent memory.

Return values

EFI_SUCCESS	The data was successfully returned.
EFI_INVALID_PARAMET↩ ER	PermanentMemoryBase + CopySize > TemporaryMemoryBase when TemporaryMemoryBase > PermanentMemoryBase.

Definition at line 163 of file SecMain.c.

Here is the call graph for this function:

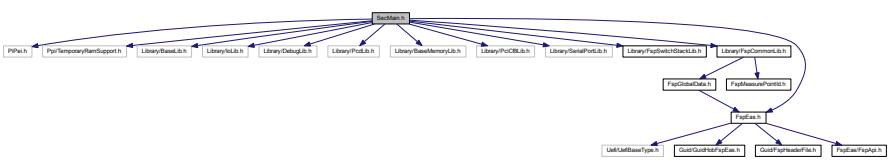


7.38 SecMain.h File Reference

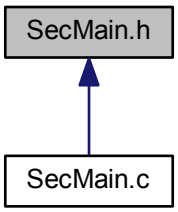
Copyright (c) 2014 - 2016, Intel Corporation.

```
#include <PiPei.h>
#include <Ppi/TemporaryRamSupport.h>
#include <Library/BaseLib.h>
#include <Library/IoLib.h>
#include <Library/DebugLib.h>
#include <Library/PcdLib.h>
#include <Library/BaseMemoryLib.h>
#include <Library/PciCf8Lib.h>
#include <Library/SerialPortLib.h>
#include <Library/FspSwitchStackLib.h>
#include <Library/FspCommonLib.h>
#include <FspEas.h>
```

Include dependency graph for SecMain.h:



This graph shows which files directly or indirectly include this file:



Functions

- VOID SecSwitchStack (IN UINT32 TemporaryMemoryBase, IN UINT32 PermenentMemoryBase)

  Switch the stack in the temporary memory to the one in the permanent memory.
- EFI_STATUS SecTemporaryRamSupport (IN CONST EFI_PEI_SERVICES **PeiServices, IN EFI_PHYS↩ ICAL_ADDRESS TemporaryMemoryBase, IN EFI_PHYSICAL_ADDRESS PermanentMemoryBase, IN UI↩ NTN CopySize)

  This service of the TEMPORARY_RAM_SUPPORT_PPI that migrates temporary RAM into permanent memory.
- VOID InitializeFloatingPointUnits (VOID)

  Initializes floating point units for requirement of UEFI specification.
- VOID SecStartup (IN UINT32 SizeOfRam, IN UINT32 TempRamBase, IN VOID *BootFirmwareVolume, IN PEI_CORE_ENTRY PeiCore, IN UINT32 BootLoaderStack, IN UINT32 ApIdx)

  Entry point to the C language phase of SEC.
- VOID ProcessLibraryConstructorList (VOID)

  Autogenerated function that calls the library constructors for all of the module's dependent libraries.

7.38.1 Detailed Description

Copyright (c) 2014 - 2016, 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.

7.38.2 Function Documentation

7.38.2.1 VOID InitializeFloatingPointUnits (VOID)

Initializes floating point units for requirement of UEFI specification.

This function initializes floating-point control word to 0x027F (all exceptions masked,double-precision, round-to-nearest) and multimedia-extensions control word (if supported) to 0x1F80 (all exceptions masked, round-to-nearest, flush to zero for masked underflow).

7.38.2.2 VOID ProcessLibraryConstructorList (VOID)

Autogenerated function that calls the library constructors for all of the module's dependent libraries.

This function must be called by the SEC Core once a stack has been established.

7.38.2.3 VOID SecStartup (IN UINT32 *SizeOfRam*, IN UINT32 *TempRamBase*, IN VOID * *BootFirmwareVolume*, IN PEI_CORE_ENTRY *PeiCore*, IN UINT32 *BootLoaderStack*, IN UINT32 *ApiIdx*)

Entry point to the C language phase of SEC.

After the SEC assembly code has initialized some temporary memory and set up the stack, the control is transferred to this function.

Parameters

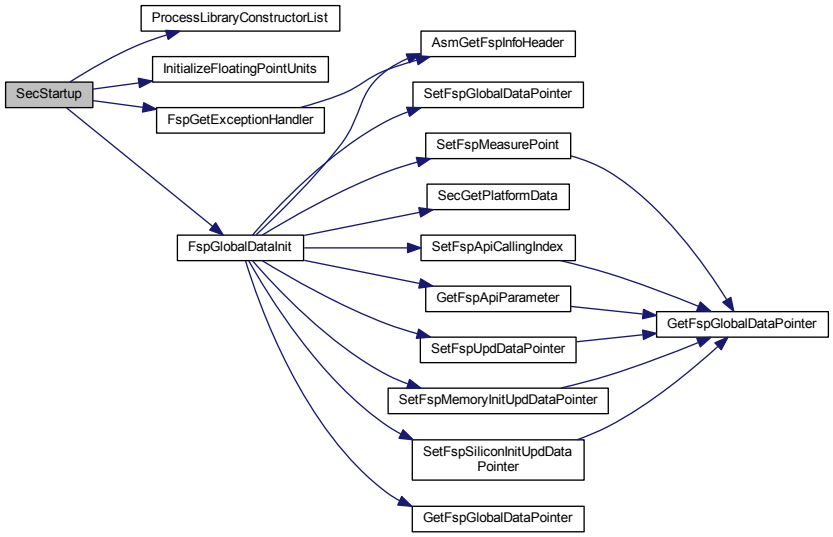
in	*SizeOfRam*	Size of the temporary memory available for use.
in	*TempRamBase*	Base address of temporary ram
in	*BootFirmware↩ Volume*	Base address of the Boot Firmware Volume.
in	*PeiCore*	PeiCore entry point.
in	*BootLoader↩ Stack*	BootLoader stack.
in	*ApiIdx*	the index of API.

Returns

This function never returns.

Definition at line 53 of file SecMain.c.

Here is the call graph for this function:



7.38.2.4 VOID SecSwitchStack (IN UINT32 *TemporaryMemoryBase*, IN UINT32 *PermenentMemoryBase*)

Switch the stack in the temporary memory to the one in the permanent memory.

This function must be invoked after the memory migration immediately. The relative position of the stack in the temporary and permanent memory is same.

Parameters

in	*Temporary↩ MemoryBase*	Base address of the temporary memory.
in	*Permenent↩ MemoryBase*	Base address of the permanent memory.

7.38.2.5 EFI_STATUS SecTemporaryRamSupport (IN CONST EFI_PEI_SERVICES ** *PeiServices*, IN EFI_PHYSICAL_ADDRESS *TemporaryMemoryBase*, IN EFI_PHYSICAL_ADDRESS *PermanentMemoryBase*, IN UINTN *CopySize*)

This service of the TEMPORARY_RAM_SUPPORT_PPI that migrates temporary RAM into permanent memory.

Parameters

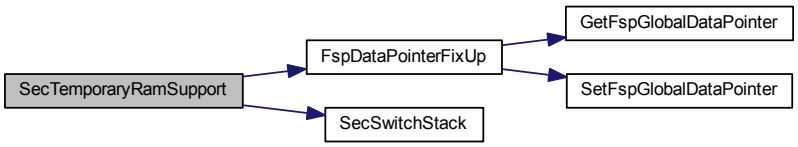
in	*PeiServices*	Pointer to the PEI Services Table.
in	*Temporary↩ MemoryBase*	Source Address in temporary memory from which the SEC or PEIM will copy the Temporary RAM contents.
in	*Permanent↩ MemoryBase*	Destination Address in permanent memory into which the SEC or PEIM will copy the Temporary RAM contents.
in	*CopySize*	Amount of memory to migrate from temporary to permanent memory.

Return values

EFI_SUCCESS	The data was successfully returned.
EFI_INVALID_PARAMET↩ ER	PermanentMemoryBase + CopySize > TemporaryMemoryBase when TemporaryMemoryBase > PermanentMemoryBase.

Definition at line 163 of file SecMain.c.

Here is the call graph for this function:



Index