InsydeH2O® 5.4 Alder Lake OEM Chipset Services Technical Reference Guide

Revision 0.1
October 23

Disclaimer

Insyde Software provides this document and the programs "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose.

This document could contain technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in future revisions of this document. Insyde Software is under no obligation to notify any person of the changes.

The following trademarks are used in this document:

Insyde and InsydeH2O are registered trademarks or trademarks of Insyde Software. All other trademarks or trade names are property of their respective holders.

Content

Table of Contents

Revision History

Revision Number	Description	Author	Release Date
0.1	Initial release.	Evonne Li	October 23, 2020

1 Introduction

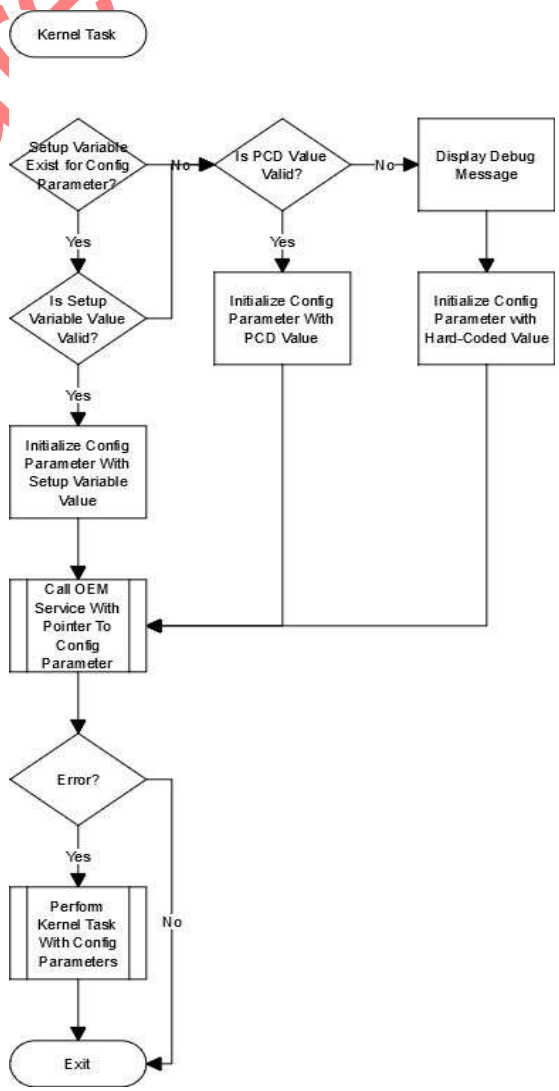
The Kernel represents the portion of H2O which manages the standard firmware behavior, including Insyde-specific features and industry standards. When the kernel code needs to determine project-specific or chipset-specific behavior, it calls the **chipset kernel services** or the **OEM kernel services**. It calls these functions to provide alternative behaviors or provide alternative initialization constants.

The Chipset represents the portion of H2O which configures the standard firmware to operate on a specific core ***chipset***, including the related CPU, memory controller and I/O controller. When the chipset code needs to determine project-specific behavior, it calls the **OEM chipset services**. It calls these functions to provide alternative behaviors or provide alternative initialization constants.

The Project represents the portion of H2O which configures the standard firmware to operate on a specific ***board***. It provides the necessary constants, OEM services and directives.

Each of the OEM Services follows a general pattern:

1. Initialize configuration parameters to defaults. In general, these should be initialized from setup variables or, if the setup variable does not exist, from PCDs with the name ***KernelXXX*** or ***CsXXXX***. This allows configuration to be changed easily in those cases where the exact configuration is known at build-time.
2. Pass the configuration parameters to a **Chipset** or **OEM service**.
3. The Chipset or OEM service may do one of three things:
   A. ***Nothing***. In this case, it will return EFI_UNSUPPORTED.
   B. ***Replace/Update Configuration Parameters***. In this case, it will alter the configuration parameters and return *EFI_MEDIA_CHANGED*.
   C. ***Replace the Kernel Behavior***. In this case, it will perform the same job as the kernel and return *EFI_SUCCESS*. The exact scope of the behavior replaced is specified by the individual function.
4. If the Chipset or OEM service returns with **an error**, then the kernel or chipset will **continue** with the configuration parameters as returned from the OEM or Chipset service.
5. If the Chipset or OEM service returns with **no error**, then the kernel or chipset will **skip** the specified behavior, assuming that it has been handled completely by the OEM or Chipset service.



1.1 Related Information

The following publications and sources of information are referred to by this document or may be useful to you:

- *UEFI Specification, Version 2.7 (May, 2017)*, http://www.uefi.org.

2 Library Description

2.1 Standard Operations

These are the services which allow the OEM to customize chipset behavior for a specific OEM board. The exact OEM chipset services are defined by Insyde on a **chipset-by-chipset** basis.

The **Intel\Xxxx\XxxxChipsetPkg** represents the package where Insyde created the code which interfaces between the reference code (R.C) provided by the silicon vendor and H2O.

2.1.1 Installation Instructions

The *phase* is the phase of H2O that the library is responsible for (DXE, PEI, SEC, BDS or SMM).

To use the OEM Chipset Services:

1. Add ***PhaseOemSvcChipsetLibDefault\|XxxxChipsetPkg\Library\PhaseOemSvcChipsetLib\PhaseOemSvcChipsetLibDefault.INF*** to the [LibraryClasses] section of the project DSC file.
2. Add ***PhaseOemSvcChipsetLib\|XxxxChipsetPkg\Library\PhaseOemSvcChipsetLib\PhaseOemSvcChipsetLib.INF*** to the [LibraryClasses] section of the project DSC file.
3. Add ***PhaseOemSvcChipsetLibDefault*** to the [LibraryClasses] section of the driver INF file.
4. Add ***#include <Library\PhaseOemSvcChipsetLib.h>*** to the driver source file.

2.1.2 To create a new OEM Chipset Services function

The *phase* is the phase of H2O execution that the library is responsible for. Valid values include Dxe, Pei, Sec, Bds or Smm.

The *project-package* is the project directory.

The *oem-function-name* is the name of the OEM Chipset Services function to implement.

The list of the OEM Chipset Services functions can be found in the Intel\Xxxx\XxxxChipsetPkg\Include\Library\PhaseOemSvcChipsetLib.h.

1. If the ***PhaseOemSvcChipsetLib*** directory does not exist under the Library sub-directory under the project directory, create it.
2. If the ***PhaseOemSvcChipsetLib.inf*** file does not exist under the *Phase*OemSvcChipsetLib directory under the project directory, copy it from Intel\Xxxx\XxxxChipsetPkg\Library\PhaseOemSvcChipsetLib\PhaseOemSvcChipsetLib.inf.
3. Edit the file ***project-package\Library\PhaseOemSvcChipsetLib.inf*** file and delete all files listed in the [sources] section.

4. If ***PhaseOemSvcChipsetLibDefault*** does not exist in the [LibraryClasses] section of the project DSC file, add ***PhaseOemSvcChipsetLibDefault|XxxxChipsetPkg\Library\PhaseOemSvcChipsetLib\PhaseOemSvcChipsetLibDefault.inf*** to the [LibraryClasses] section of the project DSC file.

5. If ***PhaseOemSvcChipsetLib*** does not exist in the [LibraryClasses] section of the project DSC file, add ***PhaseOemSvcChipsetLib|project-package\Library\PhaseOemSvcChipsetLib\PhaseOemSvcChipsetLib.inf*** to the [LibraryClasses] section of the project DSC file.

6. Copy the source file *oem-function-name.c* from ***Intel\Xxxx\XxxxChipsetPkg\Library\PhaseOemSvcChipsetLib*** to the ***project-package\Library\PhaseOemSvcChipsetLib***.

7. Add *oem-function-name.C* to the [Sources] section of the ***project-package\Library\PhaseOemSvcChipsetLib\PhaseOemSvcChipsetLib.inf***

2.2 SEC OEM Chipset Services for Alder Lake

The following is a list of the important file names for SEC OEM Chipset Services.

Library Class

SecOemSvcChipsetLibDefault

Library Files

XxxxChipsetPkg\Library\SecOemSvcChipsetLib\SecOemSvcChipsetLibDefault.inf

XxxxChipsetPkg\Library\SecOemSvcChipsetLib\SecOemSvcChipsetLib.inf

Source Files

XxxxChipsetPkg\Library\SecOemSvcChipsetLib\function-name.asm

XxxxChipsetPkg\Library\SecOemSvcChipsetLib\function-name.c

2.2.1 OemSvcHookBeforePeiCoreEntryPoint

Prototype

```
OemSvcHookBeforePeiCoreEntryPoint  PROC  NEAR  PUBLIC
```

Parameters

None

Description

This Function offers an interface for chipset code to call before "CallPeiCoreEntryPoint". This code needs to be written by Assembler and call by CALL_MMX (save ReturnAddress into MM7).

Return Status

EFI_UNSUPPORTED (al = 3)	Returns unsupported by default.
EFI_MEDIA_CHANGED (al = 13)	Alter the Configuration Parameter.
EFI_SUCCESS (al = 0)	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.3 PEI OEM Chipset Services for Alder Lake

The following is a list of the important file names for PEI OEM Chipset Services

Header File

XxxxChipsetPkg\Include\Library\PeiOemSvcChipsetLib.h

Library Class

PeiOemSvcChipsetLibDefault

Library Files

XxxxChipsetPkg\Library\PeiOemSvcChipsetLib\PeiOemSvcChipsetLibDefault.inf

XxxxChipsetPkg\Library\PeiOemSvcChipsetLib\PeiOemSvcChipsetLib.inf

Source Files

XxxxChipsetPkg\Library\PeiOemSvcChipsetLib\function-name.c

2.3.1 OemSvcModifyGpioSettingTable()

Prototype

```
OemSvcModifyGpioSettingTable (
  IN OUT  GPIO_INIT_CONFIG  **GpioTable,
  IN OUT  UINT16            *GpioTableCount
  );
```

Parameters

GpioTable

On entry, points to a structure that specifies the GPIO setting.

On exit, points to the updated structure.

GpioTableCount

On entry, points to a value that the length of GPIO table.

On exit, points to the updated value.

Description

This function offers an interface to dynamically modify GPIO table.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.

EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.3.2 OemSvcMxmDgpuPowerSequence()

Prototype

```
EFI_STATUS

OemSvcMxmDgpuPowerSequence (

  VOID

  );
```

Parameters

None

Description

This function offers an interface for OEM code to change the MXM GPU power enable sequence and modify the Switchable Graphics Information data HOB.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.3.3 OemSvcUpdatePeiPolicyInitPostMem()

Prototype

```
EFI_STATUS

OemSvcUpdatePeiPolicyInitPostMem (

  IN OUT SI_POLICY_PPI                    *SiPolicyPpi

  );
```

Parameters

SiPolicyPpi

On entry, points to SI_POLICY_PPI structure.

On exit, points to updated SI_POLICY_PPI structure.

Description

```
This function offers an interface to update SI_POLICY_PPI data before the system
installs gSiPolicyReadyPpiGuid.
```

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.3.4 OemSvcUpdatePeiPolicyInitPreMem()

Prototype

```
EFI_STATUS

OemSvcUpdatePeiPolicyInitPreMem (

  IN OUT SI_PREMEM_POLICY_PPI          *SiPreMemPolicyPpi

  );
```

Parameters

SiPreMemPolicyPpi

On entry, points to SI_PREMEM_POLICY_PPI structure.

On exit, points to updated SI_PREMEM_POLICY_PPI structure.

Description

This function offers an interface to update SI_PREMEM_POLICY_PPI data before the system installs gSiPreMemPolicyReadyPpiGuid.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.3.5 OemSvcModifyGpioSettingTablePreMem()

Prototype

```
EFI_STATUS
```

```
OemSvcModifyGpioSettingTablePreMem (
  IN OUT  GPIO_INIT_CONFIG  **GpioTablePreMem,
  IN OUT  UINT16            *GpioTableCountPreMem
  );
```

Parameters

GpioTablePreMem

On entry, points to a structure that specifies the GPIO setting.

On exit, points to the updated structure.

GpioTableCountPreMem

On entry, points to a value that the length of GPIO table.

On exit, points to the updated value.

Description

This function offers an interface to dynamically modify GPIO table before memory initialization.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.3.6 OemSvcEarlyGpioSettingTable()

Prototype

```
EFI_STATUS
OemSvcEarlyGpioSettingTable (
  IN OUT  GPIO_INIT_CONFIG  **EarlyGpioTable,
  IN OUT  UINT16            *EarlyGpioTableCount
  );
```

Parameters

EarlyGpioTable

On entry, points to a structure that specifies the GPIO setting (default is empty).

On exit, points to the updated structure.

EarlyGpioTableCount

On entry, points to a value that the length of GPIO table (default is 0).

On exit, points to the updated value.

Description

This function offers an interface on early stage to program GPIO.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.3.7 OemSvcGetInitCmosTable()

Prototype

```
EFI_STATUS
OemSvcGetInitCmosTable (
  OUT CMOS_DEFAULT_DATA      **CmosDefaultTable
  );
```

Parameters

CmosDefaultTable

The CMOS default table from OEM.

Description

This function will provide OEM CMOS default table.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.3.8 OemSvcUpdateFspmUpd()

Prototype

```
EFI_STATUS
OemSvcUpdateFspmUpd (
  IN OUT VOID **FspUpdDataPtr
  );
```

Parameters

FspUpdDataPtr

A pointer to the UPD data region data structure address.

It must convert to FSPM_UPD structure for update FSPM UPD.

If you set this pointer to NULL, FSP API will use the default FSP-M UPD settings in the binary.

Description

This function offers an interface to modify FSPM_UPD data before the FSP-M API be called.

(The interface is only used in FSP API mode)

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.3.9 OemSvcUpdateFspsUpd()

Prototype

```
EFI_STATUS
OemSvcUpdateFspsUpd (
  IN OUT VOID **FspUpdDataPtr
  );
```

Parameters

FspUpdDataPtr

A pointer to the UPD data region data structure address.

It must convert to FSPS_UPD structure for update FSPS UPD.

If you set this pointer to NULL, FSP API will use the default FSP-S UPD settings in the binary.

Description

This function offers an interface to modify FSPS_UPD data before the FSP-S API be called.

(The interface is only used in FSP API mode)

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.3.10 OemSvcHookInitSio()

Prototype

```
EFI_STATUS

OemSvcHookInitSio (

  IN OUT EFI_SIO_TABLE       *SioTable,

  IN OUT EFI_SIO_GPIO_TABLE  *SioGpioTable

  );
```

Parameters

SioTable

On entry, points to a structure that specifies the SIO register and value.On exit, points to the updated structure.

The default value is decided by gChipsetPkgTokenSpaceGuid.PcdPeiSioTable.

SioGpioTable

On entry, points to a structure that specifies the SIO GPIO register and value.On exit, points to the updated structure.

The default value is decided by gChipsetPkgTokenSpaceGuid.PcdPeiSioGpioTable1.

Description

This function offers an interface to Dynamically modify gChipsetPkgTokenSpaceGuid.PcdPeiSioGpioTable, gChipsetPkgTokenSpaceGuid.PcdPeiSioTable or change procedure about SIO initial..

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.4 DXE OEM Chipset Services for Alder Lake

The following is a list of the important file names for DXE OEM Chipset Services.

Header File

XxxxChipsetPkg\Include\Library\DxeOemSvcChipsetLib.h

Library Class

DxeOemSvcChipsetLibDefault

Library Files

XxxxChipsetPkg\Library\DxeOemSvcChipsetLib\DxeOemSvcChipsetLibDefault.inf

XxxxChipsetPkg\Library\DxeOemSvcChipsetLib\DxeOemSvcChipsetLib.inf

Source Files

XxxxChipsetPkg\Library\DxeOemSvcChipsetLib\function-name.c

2.4.1 OemSvcSetIgdOpRegion()

Prototype

```
EFI_STATUS

OemSvcSetIgdOpRegion (

  IN OUT IGD_OPREGION_PROTOCOL     *IgdOpRegion

);
```

Parameters

IgdOpRegion

On entry, points to memory buffer for Internal graphics device, this buffer set aside communicate between ACPI code and POST.

IGD_OPREGION_PROTOCOL is defined in "Related Definitions" below.

Description

This function offers an interface to update IGD OpRegion content. This area is created on POST.

This operation region is used for wrote data (VBT data or default setting) on POST and is passed for Intel VGA driver or ACPI code.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.

Related Definitions

```
struct _IGD_OPREGION_PROTOCOL {
  IGD_OPREGION_STRUCTURE  *OpRegion;
};

typedef struct {
  IGD_OPREGION_HEADER Header; ///< OpRegion header (Offset 0x0, Size 0x100)
  IGD_OPREGION_MBOX1  MBox1;  ///< Mailbox 1: Public ACPI Methods (Offset 0x100,
Size 0x100)
  IGD_OPREGION_MBOX2  MBox2;  ///< Mailbox 2: Software SCI Interface (Offset 0x200,
Size 0x100)
  IGD_OPREGION_MBOX3  MBox3;  ///< Mailbox 3: BIOS to Driver Notification (Offset
0x300, Size 0x100)
  IGD_OPREGION_MBOX4  MBox4;  ///< Mailbox 4: Video BIOS Table (VBT) (Offset 0x400,
Size 0x1800)
  IGD_OPREGION_MBOX5  MBox5;  ///< Mailbox 5: BIOS to Driver Notification Extension
(Offset 0x1C00, Size 0x400)
} IGD_OPREGION_STRUCTURE;
```

2.4.2 OemSvcHookPlatformDxe()

Prototype

```
EFI_STATUS

OemSvcHookPlatformDxe (

  IN CHIPSET_CONFIGURATION              *SetupVariable,

  IN EFI_PCI_ROOT_BRIDGE_IO_PROTOCOL  *PciRootBridgeIo,

  IN BOOLEAN                            Flag

  );
```

Parameters

SetupVariable

On entry, points to CHIPSET_CONFIGURATION instance.

On exit, points to updated CHIPSET_CONFIGURATION instance.

Type `CHIPSET_CONFIGURATION` is defined in **Intel\Xxxx\XxxxChipsetPkg\Include\ChipsetSetupConfig.h**.

PciRootBridgeIo

On entry, points to EFI_PCI_ROOT_BRIDGE_IO_PROTOCOL instance.

On exit, points to updated EFI_PCI_ROOT_BRIDGE_IO_PROTOCOL instance.

Type `EFI_PCI_ROOT_BRIDGE_IO_PROTOCOL` is defined in **Include\Protocol\PciRootBridgeIo.h**.

Flag

TRUE: Before DxePlatformEntryPoint.

FALSE: After DxePlatformEntryPoint.

Description

This function offers an interface to hook before and after DxePlatformEntryPoint (PlatformDxe.inf).

This hook does some setting between configure GPIO for SATA, Initialize PCH register, check chipset TXT capability and Install Legacy USB setup policy protocol.

It is the DXE hook for OEM to customize Platform related code. For Example, it can set EC shut down temperature.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.4.3 OemSvcSetUsbLegacyPlatformOptions()

Prototype

```
EFI_STATUS

OemSvcSetUsbLegacyPlatformOptions (

  IN OUT USB_LEGACY_MODIFIERS                *UsbLegacyModifiers

  );
```

Parameters

UsbLegacyModifiers

On entry, points to USB_LEGACY_MODIFIERS instance.

On exit, points to updated USB_LEGACY_MODIFIERS instance.

USB_LEGACY_MODIFIERS is defined in “Related Definitions” below.

Description

This function offers an interface to modify USB legacy options.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.

Related Definitions

```
typedef struct {
  UINT32            UsbLegacyEnable          :1;
  UINT32            UsbBoot                  :2;
  UINT32            UsbZip                   :2;
```

```
  UINT32            UsbZipEmulation          :2;
  UINT32            UsbFixedDiskWpBootSector :2;
  UINT32            UsbEhciSupport           :1;
  UINT32            UsbSupportSingleLunOnly  :1;
  UINT32            UsbUefiOnly              :1;
  UINT32            Reserved                 :20;
  UINT32            UsbIgnoreDevicesPtr;
} USB_LEGACY_MODIFIERS;
```

2.4.4 OemSvcUpdateOemBadgingLogoData()

Prototype

```
EFI_STATUS

OemSvcUpdateOemBadingLogoData (

  IN OUT EFI_OEM_BADGING_LOGO_DATA     **EfiOemBadingLogoData,

  IN OUT UINTN                           *BadgingDataSize,

  IN OUT OEM_BADGING_STRING            **OemBadgingString,

  IN OUT OEM_BADGING_STRING            **OemBadgingStringInTextMode,

  IN OUT UINTN                           *StringCount,

  IN OUT OEM_BADGING_STRING            **OemBadgingStringAfterSelectWithMe,

  IN OUT OEM_BADGING_STRING            **OemBadgingStringAfterSelectWithMeInTextMode

  );
```

Parameters

EfiOemBadingLogoData

On entry, points to a structure that specifies image data.

On exit, points to updated structure.

`EFI_OEM_BADGING_LOGO_DATA` is defined in "Related Definitions" in this section below.

BadgingDataSize

On entry, the size of EFI_OEM_BADGING_LOGO_DATA matrix, in bytes.

On exit, the size of updated EFI_OEM_BADGING_LOGO_DATA matrix, in bytes.

OemBadgingString

On entry, points to OEM_BADGING_STRING matrix.

On exit, points to updated OEM_BADGING_STRING matrix.

`OEM_BADGING_STRING` is defined in "Related Definitions" in this section below.

OemBadgingStringInTextMode

On entry, points to OEM_BADGING_STRING matrix in text mode.

On exit, points to updated OEM_BADGING_STRING matrix in text mode.

StringCount

The number is POST string count.

On entry, based on SetupVariablei→QuietBoot

1: The number of entries in OemBadgingString.

0: The number of entries in OemBadgingStringInTextMode.

On exit, based on SetupVariablei→QuietBoots

1: The number of entries in updated OemBadgingString,

0: The number of entries in updated OemBadgingStringInTextMode.

OemBadgingStringAfterSelectWithMe

On entry, points to OEM_BADGING_STRING matrix after selected.

On exit, points to updated OEM_BADGING_STRING matrix after selected.

OemBadgingStringAfterSelectWithMeInTextMode

On entry, points to OEM_BADGING_STRING matrix after selected in text mode.

On exit, points to updated OEM_BADGING_STRING matrix after selected in text mode.

Description

This function offers an interface to modify OEM Logo and POST String.

Before press hotkey

- When QuietBoot enable, system will use OemBadgingString
- When QuietBoot disable, system will use OemBadgingStringInTextMode After press hotkey
- When QuietBoot enable, system will use OemBadgingStringAfterSelectWithMePointer
- When QuietBoot disable, system will use OemBadgingStringAfterSelectWithMeInTextModePointers

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

Related Definitions

```
typedef struct {
  EFI_GUID                                FileName;
  EFI_BADGING_SUPPORT_FORMAT              Format;
  EFI_BADGING_SUPPORT_DISPLAY_ATTRIBUTE   Attribute;
  UINTN                                   CoordinateX;
  UINTN                                   CoordinateY;
  BOOLEAN                                 Visible;
  EFI_BADGING_SUPPORT_IMAGE_TYPE          Type;
} EFI_OEM_BADGING_LOGO_DATA;
```

FileName

Guid of Image.

Format

Format of Image.

Attribute

Location of Image displayed.

CoordinateX

Desired X resolution when displaying image.

CoordinateY

Desired Y resolution when displaying image.

Visible

Image display on NULL or TRUE

Image not display on FALSE

Type

Type of image.

```
typedef enum {
  EfiBadgingSupportFormatBMP,
  EfiBadgingSupportFormatJPEG,
  EfiBadgingSupportFormatTIFF,
  EfiBadgingSupportFormatGIF,
  EfiBadgingSupportFormatPCX,
  EfiBadgingSupportFormatTGA,
  EfiBadgingSupportFormatPNG,
  EfiBadgingSupportFormatUnknown
} EFI_BADGING_SUPPORT_FORMAT;
```

```
typedef enum {
  EfiBadgingSupportDisplayAttributeLeftTop,
  EfiBadgingSupportDisplayAttributeCenterTop,
  EfiBadgingSupportDisplayAttributeRightTop,
  EfiBadgingSupportDisplayAttributeCenterRight,
  EfiBadgingSupportDisplayAttributeRightBottom,
  EfiBadgingSupportDisplayAttributeCenterBottom,
  EfiBadgingSupportDisplayAttributeLeftBottom,
```

```
  EfiBadgingSupportDisplayAttributeCenterLeft,
  EfiBadgingSupportDisplayAttributeCenter,
  EfiBadgingSupportDisplayAttributeCustomized
} EFI_BADGING_SUPPORT_DISPLAY_ATTRIBUTE;

typedef enum {
  EfiBadgingSupportImageLogo,
  EfiBadgingSupportImageBadge,
  EfiBadgingSupportImageBoot
} EFI_BADGING_SUPPORT_IMAGE_TYPE;

typedef struct _OEM_BADGING_STRING {
  UINTN                                 X;// Location X
  UINTN                                 Y;// Location Y
  EFI_UGA_PIXEL                         Foreground;// Strings Foreground
  EFI_UGA_PIXEL                         Background;// Strings Background
  STRING_REF                            StringToken;
  BOOLEAN                               Fun;
} OEM_BADGING_STRING;
```

X

Desired X resolution when displaying string.

Y

Desired Y resolution when displaying string.

Foreground

Desired foreground when displaying string.

Background

Desired background when displaying string.

StringToken

Value of String.

Fun

Executed function when displaying string.

Note

If OEM badging string has been modified, user also modified POST string on "XXXBoardpkg\Project.uni".

EX:

```
#string STR_OEM_BADGING_STR_ESC          #language en-US  "Press Esc for boot
options"
```

2.4.5 OemSvcUpdatePlatformNvs()

Prototype

```
EFI_STATUS
```

```
OemSvcUpdatePlatformNvs (
  PLATFORM_NVS_AREA                    *mPlatformNvsArea,
  OEM_PLATFORM_NVS_AREA               *mOemPlatformNvsArea
  );
```

Parameters

mPlatformNvsArea

On entry, points to **PLATFORM_NVS_AREA** instance.

On exit, points to updated **PLATFORM_NVS_AREA** instance.

Type **PLATFORM_NVS_AREA** is defined in **Intel\Xxxx\XxxxChipsetPkg\Include\PlatformNvsAreaDef.h**.

mOemPlatformNvsArea

On entry, points to **OEM_PLATFORM_NVS_AREA** instance.

On exit, points to updated **OEM_PLATFORM_NVS_AREA** instance.

OEM_PLATFORM_NVS_AREA is defined in "Related Definitions" in this section below.

Description

This function offers an interface to update GlobalNvs table and OEM GlobalNvs table content.

Global NVS table and OEM Global NVS table is set aside communicate between ACPI code and POST.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.

Related Definitions

```
typedef struct {
  //
  // The definitions below need to be matched OGNS definitions in OemPlatformNvs.ASL
  // and can be modified by OEM
  //
  UINT8       OemPlatformNvsData00;    // 00
  UINT8       OemPlatformNvsData01;    // 01
  UINT8       OemPlatformNvsData02;    // 02
  UINT8       OemPlatformNvsData03;    // 03
  UINT8       OemPlatformNvsData04;    // 04
  UINT8       OemPlatformNvsData05;    // 05
  UINT8       OemPlatformNvsData06;    // 06
  UINT8       OemPlatformNvsData07;    // 07
  UINT8       OemPlatformNvsData08;    // 08
  UINT8       OemPlatformNvsData09;    // 09
  UINT8       OemPlatformNvsData10;    // 10
  UINT8       OemEmbeddedControllerFlag;
} OEM_PLATFORM_NVS_AREA;
```

2.4.6 OemSvcUpdateAmtPlatformPolicy()

Prototype

```
EFI_STATUS

OemSvcUpdateAmtPlatformPolicy (

  IN OUT AMT_POLICY_PROTOCOL        *AmtPlatformPolicy

  );
```

Parameters

AmtPlatformPolicy

On entry, points to AMT_POLICY_PROTOCOL structure.

On exit, points to updated AMT_POLICY_PROTOCOL structure.

Description

This function offers an interface to modify AMT_POLICY_PROTOCOL data before the system installs AMT_POLICY_PROTOCOL.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.4.6 OemSvcUpdateDxeMePolicy()

Prototype

```
EFI_STATUS

OemSvcUpdateDxeMePolicy (

  IN OUT ME_POLICY_PROTOCOL         *MePlatformPolicy

  );
```

Parameters

MePlatformPolicy

On entry, points to ME_POLICY_PROTOCOL structure.

On exit, points to updated ME_POLICY_PROTOCOL structure.

Description

This function offers an interface to modify ME_POLICY_PROTOCOL data before the system installs ME_POLICY_PROTOCOL.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.4.7 OemSvcUpdateDxePlatformSaPolicy()

Prototype

```
EFI_STATUS
OemSvcUpdateDxePlatformSaPolicy (
  IN OUT SA_POLICY_PROTOCOL          *SaPlatformPolicy
  );
```

Parameters

SaPlatformPolicy

On entry, points to SA_POLICY_PROTOCOL structure.

On exit, points to updated SA_POLICY_PROTOCOL structure.

Description

This function offers an interface to modify SA_POLICY_PROTOCOL data before the system installs SA_POLICY_PROTOCOL.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.4.8 OemSvcUpdateBiosProtectTable()

Prototype

```
EFI_STATUS
OemSvcUpdateBiosProtectTable (
  OUT BIOS_PROTECT_REGION        **BiosRegionTable,
  OUT UINT8                      *ProtectRegionNum
  );
```

Parameters

BiosRegionTable

Pointer to BiosRegion Table.

BIOS_PROTECT_REGION is defined in "Related Definitions" below.

ProtectRegionNum

The number of Bios protect region instances.

Description

Customize BIOS protect region before boot to OS.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

Related Definitions

```
typedef struct {
  UINT32                  Base;
  UINT32                  Size;
} BIOS_PROTECT_REGION;
```

2.4.9 OemSvcSetBootDisplayDevice()

Prototype

```
EFI_STATUS

OemSvcSetBootDisplayDevice (

  IN OUT CHIPSET_CONFIGURATION          *SetupNVRam,

  IN OUT SA_SETUP                       *SaSetupNVRam

  );
```

Parameters

SetupNVRam

Pointer to CHIPSET_CONFIGURATION instance.

SaSetupNVRam

Points to SA_SETUP instance.

Description

This function provides an interface to set Boot Display Device.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.4.10 OemSvcUpdateDsdtAcpiTable()

Prototype

```
EFI_STATUS
OemSvcUpdateDsdtAcpiTable (
  CONST UINT32                  *Signature,
  );
```

Parameters

Signature

Pointer to ACPI Common Header Signature.

Description

This function offers an interface to update the ACPI DSDT table.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.4.11 OemSvcHookRouteConfig()

Prototype

```
EFI_STATUS
OemSvcHookRouteConfig (
  IN OUT CHIPSET_CONFIGURATION          *ScBuffer,
  IN     UINT32                         BufferSize,
  IN OUT VOID                           *RcScBuffer,
  IN     UINT32                         ScuRecord
```

```
);
```

Parameters

ScBuffer

A pointer to CHIPSET_CONFIGURATION struct.

BufferSize

System configuration size.

RcScBuffer

A pointer to RC relative variables struct.

ScuRecord

The bit mask of the currently SCU record. Bit 0 = 1 (SCU_ACTION_LOAD_DEFAULT), It indicates system do load default action.

Description

This function provides an interface to hook GenericRouteConfig.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.4.12 OemSvcInitTbtFunc()

Prototype

```
EFI_STATUS
OemSvcInitTbtFunc (
  IN CHIPSET_CONFIGURATION            *SetupData,
  IN SETUP_DATA                       *RcSetupData,
  IN SA_SETUP                         *SaSetup,
  IN PCH_SETUP                        *PchSetup
  );
```

Parameters

SetupData

A pointer to CHIPSET_CONFIGURATION struct.

RcSetupData

Points to SETUP_DATA instance.

SaSetup

Points to SA_SETUP instance.

PchSetup

Points to PCH_SETUP instance.

Description

This function offers an interface to initial Thunderbolt Setup data in SetupRuntimeUpdateEveryBoot.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.4.13 OemSvcSetBacklightControl()

Prototype

```
EFI_STATUS
OemSvcSetBacklightControl(
  VOID
  );
```

Parameters

None

Description

This function provides an interface to update backlight brightness value.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.4.14 OemSvcHookPlatformReset()

Prototype

```
EFI_STATUS
OemSvcHookPlatformReset(
    IN EFI_RESET_TYPE           ResetType,
    IN EFI_STATUS               ResetStatus
  );
```

Parameters

ResetType

Warm or cold.

ResetStatus

Possible cause of reset.

Description

This function provides an interface to update backlight brightness value.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.5 SMM OEM Chipset Services for Alder Lake

The following is a list of the important file names for SMM OEM Chipset Services.

Header File

XxxxChipsetPkg\Include\Library\SmmOemSvcChipsetLib.h

Library Class

SmmOemSvcChipsetLibDefault

Library Files

XxxxChipsetPkg\Library\SmmOemSvcChipsetLib\SmmOemSvcChipsetLibDefault.inf

XxxxChipsetPkg\Library\SmmOemSvcChipsetLib\SmmOemSvcChipsetLib.inf

Source Files

XxxxChipsetPkg\Library\SmmOemSvcChipsetLib\function-name.c

2.5.1 OemSvcGetSaveRestorePciDeviceOemList()

Prototype

```
EFI_STATUS

OemSvcGetSaveRestorePciDeviceOemList (

  OUT SR_OEM_DEVICE     **PciDeviceOemList

  );
```

Parameters

PciDeviceOemList

Points to the list of OEM PCI device registers which must be saved on an S3.

SR_OEM_DEVICE is defined in "Related Definitions" below.

Description

This function provides OEM to add additional PCI registers which is not listed on global register table PciDeviceSubResList[]. It will replace global register table PciDeviceOemSubResList[].

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.

Related Definitions

```
typedef struct {
  UINTN               RegNum;
  UINT8               PciBus;
  UINT8               PciDev;
  UINT8               PciFun;
  UINT8               *PciRegTable;
  UINT32              *PciRegTableSave;
} SR_DEVICE;
```

RegNum

Save and restore register count.

PciBus

Pci Bus number.

PciDev

Pci Device number.

PciFunc

Pci Function number.

PciRegTable

A point to an array of save/restore register offset.

PciRegTableSave

Points to an array of save/restore register value.

```
typedef struct {
  UINT8                                 Bus;
  UINT8                                 Dev;
  UINT8                                 Fun;
} P2P_BRIDGE;
```

Bus

Bridge Bus number.

Dev

Bridge Device number.

Fun

Bridge Function number.

```
typedef struct {
  SR_DEVICE                         Device;
  P2P_BRIDGE                        P2PB;
} SR_OEM_DEVICE;



typedef struct {
  SR_DEVICE                         *Chipset;
  SR_OEM_DEVICE                     *Oem;
} SR_TABLE;
```

2.5.2 OemSvcVbiosHookCallBack()

Prototype

```
EFI_STATUS

OemSvcVbiosHookCallBack (

  IN UINT32                         Int15FunNum,

  IN OUT EFI_IA32_REGISTER_SET       *CpuRegs,

  IN VOID                           *Context

  );
```

Parameters

Int15FunNum

Int15 function number.

CpuzRegs

The structure containing CPU Registers (AX, BX, CX, DX etc.).

`EFI_IA32_REGISTER_SET` is defined in "Related Definitions" below.

Context

Context.

Description

This function provides an interface to do additional VbiosHookCallBack function that are listed in OemInt15VbiosFunctionHook array.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

Related Definitions

```
///
/// EFI_IA32_REGISTER_SET
///
typedef union {
  EFI_DWORD_REGS    E;
  EFI_WORD_REGS     X;
  EFI_BYTE_REGS     H;
} EFI_IA32_REGISTER_SET;

///
/// EFI_DWORD_REGS
///
typedef struct {
    UINT32            EAX;
    UINT32            EBX;
    UINT32            ECX;
    UINT32            EDX;
    UINT32            ESI;
    UINT32            EDI;
    EFI_EFLAGS_REG    EFlags;
    UINT16            ES;
    UINT16            CS;
    UINT16            SS;
    UINT16            DS;
    UINT16            FS;
    UINT16            GS;
    UINT32            EBP;
    UINT32            ESP;
} EFI_DWORD_REGS;

///
/// EFI_FLAGS_REG
///
typedef struct {
  UINT16     CF:1;
  UINT16     Reserved1:1;
```

```
  UINT16      PF:1;
  UINT16      Reserved2:1;
  UINT16      AF:1;
  UINT16      Reserved3:1;
  UINT16      ZF:1;
  UINT16      SF:1;
  UINT16      TF:1;
  UINT16      IF:1;
  UINT16      DF:1;
  UINT16      OF:1;
  UINT16      IOPL:2;
  UINT16      NT:1;
  UINT16      Reserved4:1;
} EFI_FLAGS_REG;

///
/// EFI_WORD_REGS
///
typedef struct {
    UINT16           AX;
    UINT16           ReservedAX;
    UINT16           BX;
    UINT16           ReservedBX;
    UINT16           CX;
    UINT16           ReservedCX;
    UINT16           DX;
    UINT16           ReservedDX;
    UINT16           SI;
    UINT16           ReservedSI;
    UINT16           DI;
    UINT16           ReservedDI;
    EFI_FLAGS_REG    Flags;
    UINT16           ReservedFlags;
    UINT16           ES;
    UINT16           CS;
    UINT16           SS;
    UINT16           DS;
    UINT16           FS;
    UINT16           GS;
    UINT16           BP;
    UINT16           ReservedBP;
    UINT16           SP;
    UINT16           ReservedSP;
} EFI_WORD_REGS;

///
/// EFI_BYTE_REGS
///
typedef struct {
    UINT8   AL, AH;
    UINT16  ReservedAX;
    UINT8   BL, BH;
    UINT16  ReservedBX;
    UINT8   CL, CH;
    UINT16  ReservedCX;
    UINT8   DL, DH;
    UINT16  ReservedDX;
} EFI_BYTE_REGS;
```

2.5.3 OemSvcGetOemInt15VbiosFunctionlist()

Prototype

```
EFI_STATUS
OemSvcGetOemInt15VbiosFunctionlist(
  IN OUT UINT16              **OemInt15VbiosFunctionlist,
  OUT    UINT16              *Size
  );
```

Parameters

OemInt15VbiosFunctionlist

Point to OemInt15VbiosFunctionHook array.

This array record EAX register value on INT15 trigger

Size

The number of entries in OemInt15VbiosFunctionHook array

Description

This function provides an interface to get OemInt15VbiosFunctionHook.

If function is added on INT15 trigger, OemInt15VbiosFunctionlist and Size are updated.

When user triggered INT15, it compared EAX register and OemInt15VbiosFunctionlist data. If data match, it will call VbiosHookCallBack function.

Caller will use this to install additional VbiosHookCallBack function.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.6 BASE OEM Chipset Services for Alder Lake

The following is a list of the important file names for BASE OEM Chipset Services.

Header File

XxxxChipsetPkg\Include\Library\BaseOemSvcChipsetLib.h

Library Class

BaseOemSvcChipsetLibDefault

Library Files

XxxxChipsetPkg\Library\BaseOemSvcChipsetLib\BaseOemSvcChipsetLibDefault.inf

XxxxChipsetPkg\Library\BaseOemSvcChipsetLib\BaseOemSvcChipsetLib.inf

Source Files

XxxxChipsetPkg\Library\BaseOemSvcChipsetLib\function-name.c

2.6.1 OemSvcEcGetLidState()

Prototype

```
OemSvcEcGetLidState (
  OUT EFI_STATUS      *EcGetLidState,
  OUT UINT8           *LidIsOpen
  );
```

Parameters

EcGetLidState

The status of get Lid.

LidIsOpen

TRUE: Lid is open.

FALSE: Lid is close.

Description

Get Lid state from EC.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.6.2 OemSvcEcPowerState()

Prototype

```
EFI_STATUS
  OemSvcEcPowerState (
  IN OUT BOOLEAN      *PowerStateIsAc
```

```
);
```

Parameters

PowerStateIsAc

A Boolean pointer.

TRUE: AC power.

FALSE: Battery power.

Description

Get power state from EC. If power state cannot be determined, battery powered is assumed.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.6.3 OemSvcEcSaveRestoreKbc()

Prototype

```
EFI_STATUS
OemSvcEcSaveRestoreKbc (
  IN BOOLEAN      SaveRestoreFlag
  );
```

Parameters

SaveRestoreFlag

Save/Restore Flag.

TRUE: restore Keyboard command byte

FALSE: Save Keyboard command byte.

Description

Provide hook function for OEM to implement save and restore KBC.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled

	completely by this function.

2.6.4 OemSvcEcSetDswMode()

Prototype

```
EFI_STATUS
OemSvcEcSetDswMode (
  OUT EFI_STATUS      *SetDswModeStatus,
  IN UINT8            DswMode
  );
```

Parameters

SetDswModeStatus

The status of set DSW mode.

DswMode

DSW mode.

0x0 Disable DSW mode

0x1 Enable DSW in S5 (DC)

0x3 Enable DSW in S4-S5 (DC)

0x5 Enable DSW in S3-S4-S5 (DC)

Description

Set DSW mode to EC.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.6.5 OemSvcEcVersion()

Prototype

```
EFI_STATUS
OemSvcEcVersion (
  OUT EFI_STATUS      *ReadEcVersionStatus,
```

```
  OUT UINT8           *MajorNum,
  OUT UINT8           *MinorNum
  );
```

Parameters

ReadEcVersionStatus

The status of read EC version.

MajorNum

EC major number.

MinorNum

EC minor number.

Description

Get major/minor version from EC.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.6.6 OemSvcEcSetCriticalThermal()

Prototype

```
EFI_STATUS
OemSvcEcSetCriticalThermal (
  IN UINT8      Temperature
  );
```

Parameters

Temperature

The temperature of Critical Thermal.

Description

Send Critical thermal to EC.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.

EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.6.7 OemSvcEcSetLowPowerMode()

Prototype

```
EFI_STATUS
OemSvcEcSetDswMode (
  IN BOOLEAN      LowPowerMode
  );
```

Parameters

LowPowerMode

Low Power Mode.

TRUE: Enable Low Power Mode.

FALSE: Disable Low Power Mode.

Description

Set Low Power Mode to EC.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.6.8 OemSvcEcDetectEcPresent()

Prototype

```
EFI_STATUS
OemSvcEcDetectEcPresent (
  OUT EFI_STATUS      *CommandStatus,
  OUT BOOLEAN         *Present
  );
```

Parameters

CommandStatus

Command status.

Present

TRUE: EC is present

FALSE: EC is absent.

Description

Detect EC Present.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.6.9 OemSvcEcGetPcieDockStatus()

Prototype

```
EFI_STATUS
OemSvcEcGetPcieDockStatus (
  OUT EFI_STATUS      *CommandStatus,
  OUT UINT8           *Dock
  );
```

Parameters

CommandStatus

Command status.

Dock

Dock Status.

The bit0 is PCIe Dock Status, 1 = docked.

Description

Get PCIe dock status from EC.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.

EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.

2.6.10 OemSvcEcReadEcRam()

Prototype

```
EFI_STATUS
OemSvcEcReadEcRam (
  OUT EFI_STATUS *ReadEcRamStatus,
  IN  OUT UINT8  *DataBuffer
  );
```

Parameters

ReadEcRamStatus

Read EC ram status.

DataBuffer

For input, DataBuffer means offset, while it means data for output.

Description

Read the EC memory.

Return Status

EFI_UNSUPPORTED	Returns unsupported by default.
EFI_MEDIA_CHANGED	Alter the Configuration Parameter.
EFI_SUCCESS	The function performs the same operation as caller. The caller will skip the specified behavior and assuming that it has been handled completely by this function.