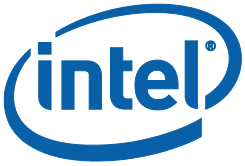


BIOS Implementation of UCSI

Technical White Paper

February 2016

Revision 001

Document: 333897-001

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.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. **No computer system can be absolutely secure.** Check with your system manufacturer or retailer or learn more at intel.com.

Intel technologies may require enabled hardware, specific software, or services activation. Check with your system manufacturer or retailer.

The products described 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.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or visit www.intel.com/design/literature.htm.

By using this document, in addition to any agreements you have with Intel, you accept the terms set forth below.

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

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.

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

Copyright © 2016, Intel Corporation. All rights reserved.

Contents

Figures

Tables

Revision History

Document Number	Revision Number	Description	Revision Date
333897	001	Initial release.	February 2016

§ §

1 Introduction

USB Type-C is a connector shape for USB. Type-C connector solves some issues present in all other USB connectors like:

- No “polarity” rules (make it easier for end users)
- USB Host and USB Function (Device) both use the same connector (Type-C), i.e. essentially all Type-C ports are “dual role” ports.
- Can be inserted right-side up or upside-down.
- Two types of cables: full-featured and USB 2.0 only
- Optional support for Power Delivery (PD). Provides wider range of power & charging options: up to 5A @ 20V. When Power Delivery is supported, it must use USB PD method (independent of Guest Protocol).
- Power and data on a single connector (removes need for Micro-AB combo) and bi-directional.
- Replaces modulated VBUS negotiation messaging (problematic) with baseband Configuration Channel (CC) signal.
- Supports negotiable pin re-definition for Alternate Interface functionality (i.e. non USB protocols for Guest Protocols) such as display port, PCIe, audio etc. This is optional.

Note: USB Type-C is supported by wide variety of USB-IF member companies.

1.1 Terminology

Term	Description
ACPI	Advanced Configuration and Power Interface
LPM	Local Policy Manager
OPM	OS Policy Manager
OSV	Operating System Vendors
PD	Power Delivery
PPM	Platform Policy Manager
SCI	System Control Interrupt
UCSI	USB Type-C connector System Software Interface
USB	Universal Serial Bus
VBUS	Pin of USB connector

1.2 Reference Documents

Document	Document No./Location
USB Type-C Connector System Software Interface [UCSI]	http://www.intel.com/content/dam/www/public/us/en/documents/technical-specifications/usb-type-c-ucsi-spec.pdf

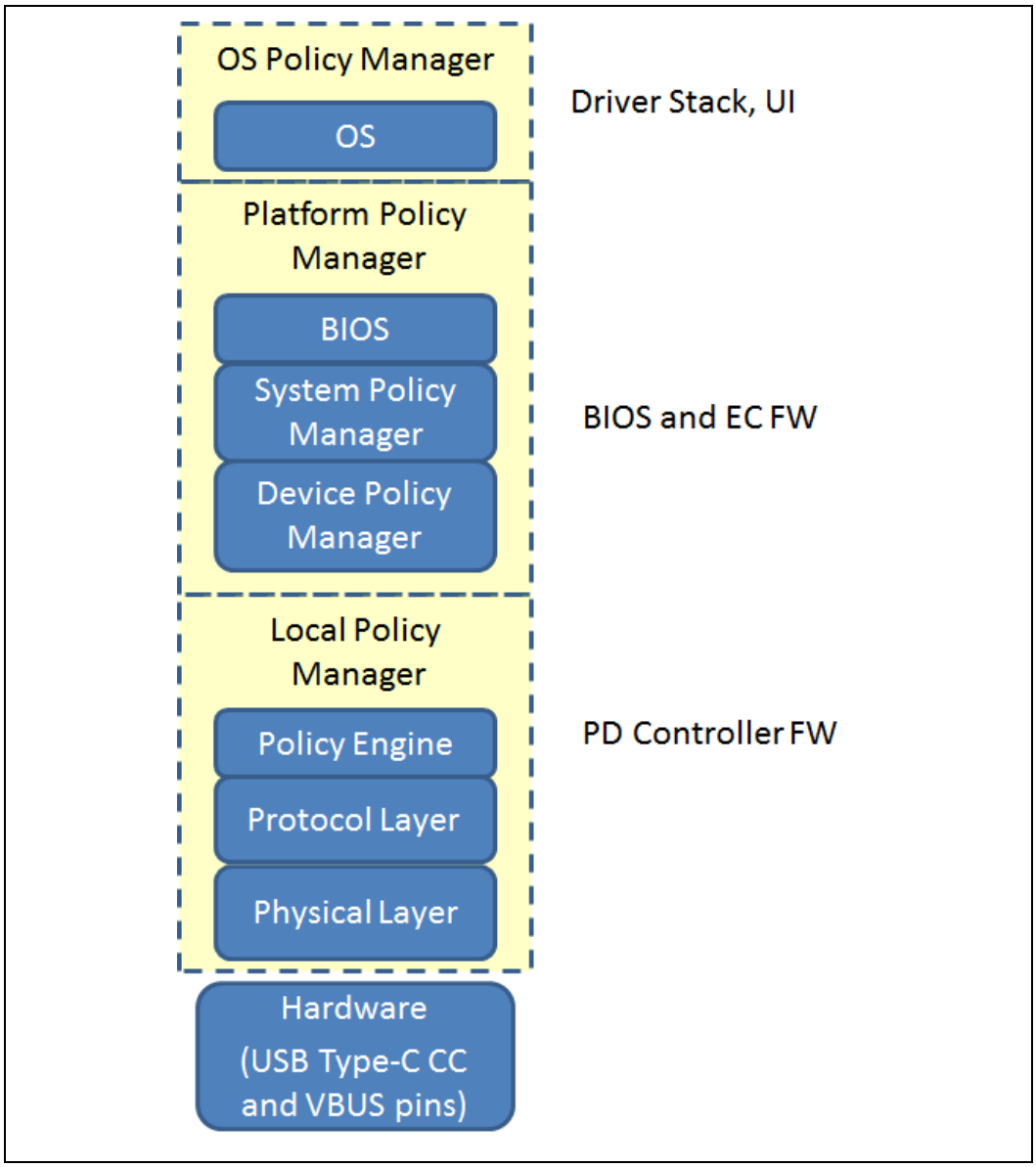
§ §

2 USB Type-C

2.1 USB Type-C System View

Figure 2-1. System View of USB Type-C shows the system view of USB Type-C block (yellow boxes) that map to the components in USB PD2.0 Specification (grey boxes).

Figure 2-1. System View of USB Type-C



Where from Figure 2-1:

- OPM (OS Policy Manager)
  - Communicates any OS based requests to USB Type-C connector/device and vice versa
  - OSV specific
- PPM (Platform Policy Manager)
  - Manage USB Type-C ports on the system and apply default policies to them (system policy, power delivery)
  - Platform specific (HW/FW/vendor provided OS SW)
- LPM (Local Policy Manager)
  - An individual USB Type-C policy manager on the platform
  - Can have more than one (depending on # of USB Type-C ports)

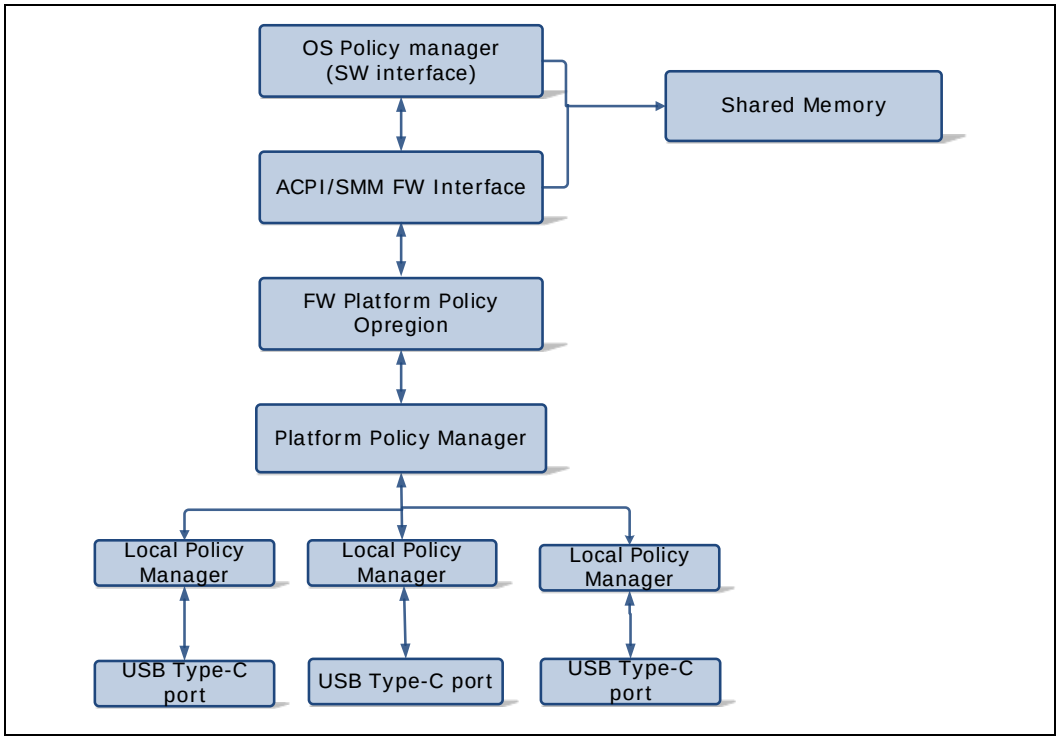
The interface between OS Policy Manager and Platform Policy Manager, Platform Policy Manager Interface, provides a means to implement the protocol that is described in the UCSI specification to allow communication from USB Type-C controller to the host or vice versa. The UCSI data structures can also be transitioned to hardware register definition easily in the future to allow system software directly communicate with and access the USB Type-C hardware.

2.2 USB Type-C Platform Policy Manager Interface

OPM (OS Policy Manager) block represents the Operating System's software interface in Figure 2-2. USB Type-C SW Policy Block Diagram. The end point connector is attached to the main USB Type-C Power Delivery Controller on the board. A Local Policy Manager for the USB Type-C hardware is the firmware that runs on the USB Type-C Power Delivery Controller and maintains the specific port information and communication. The Platform policy manager is the embedded controller firmware on the platform that maintains information of different USB Type-C ports on the platform and manages all communication to these ports through the LPM.

Note: *Be aware this is a reference implementation and it could be implemented differently depending on the platform design. The following sections assume the usage of ACPI for PPM interface and EC for PPM.*

Figure 2-2. USB Type-C SW Policy Block Diagram



2.2.1 Shared Mailbox Layout

A shared mailbox is established to exchange information between the OPM and the BIOS interface at Table 2-1. The mailbox is 48 bytes of contiguous data and is one common structure for both agents with the below layout. (Source: UCSI Specification)

Table 2-1. Shared Mailbox for OPM/BIOS Interfaces

Offset (Bytes)	Mnemonic	Memory Location Name	Direction	Size (bits)
0	**VERSION**	UCSI Version Number	PPM->OPM	16
2	**RESERVED**	Reserved	N/A	16
4	**CCI**	USB Type-C Command Status and Connector Change Indication	PPM->OPM	32
8	**CONTROL**	USB Type-C Control	OPM->PPM	64
16	**MESSAGE IN**	USB Type-C Message In	PPM->OPM	128
32	**MESSAGE OUT**	USB Type-C Message Out	OPM->PPM	128

Since all the fields are already unidirectional, there is no other explicit mechanism needed to maintain mutual exclusion between BIOS and OPM accesses.

2.2.2 OPM / ACPI / EC FW Responsibilities

This section will call out the specifics roles and responsibilities of each FW Interface involved in the USB Type-C FW stack communication.

2.2.2.1 OPM Responsibilities

1. Enumerate USB Type-C ACPI device with _HID
2. Update shared mailbox with required command and parameters
3. Call _DSM for notifying ACPI of memory update
4. Enable and Acknowledge alerts/command complete indicators from EC
5. Handle notification event

2.2.2.2 ACPI Responsibilities

1. Report USB Type-C ACPI device with _HID
2. Define and initialize shared mailbox structure
3. Define _DSM method to read shared mailbox when invoked
4. Define and initialize EC OpRegion to communicate with USB Type-C
5. Trigger notification alert to OPM on command completion and response or device alert

2.2.2.3 EC Responsibilities

1. Communicate OPM command request to USB Type-C controller
2. Communicate USB Type-C response and/or alert to OPM
3. Clear indicator bits based on OPM acknowledgement

2.2.3 USB Type-C BIOS Requirements

The below requirements needs to be covered by BIOS:

1. The USB Type-C device needs to be reported as ACPI device to Operating System with its own unique ID.

   Example:

   ```
   Name (_HID, EISAID("USBC000"))
   Name (_CID, EISAID("PNP0CA0"))
   Name (_UID, 1)
   Name (_DDN, "USB Type-C")
   Name (_ADR, 0x0)
   ```

2. ACPI code must also define _DSM method that is invoked by OPM driver every time it updates the shared mailbox.

   Example:

   ```
   Method (_DSM, 4, Serialized, 0, UnknownObj, {BuffObj, IntObj,
   IntObj, PkgObj})
   {
   ```

```
  // Compare passed in UUID to supported UUID.
  If (LEqual(Arg0, ToUUID ("6f8398c2-7ca4-11e4-ad36-631042b5008f")))
// UUID for USB type-C
  {
      ...
  }  // End UUID check
} // End _DSM Method
```

3. In addition this ACPI code defines a shared mailbox in main memory to share with the OPM driver.

   Example:

```
OperationRegion (USBC, SystemMemory, 0x7DF76000, 0x30)
Field (USBC,AnyAcc,Lock,Preserve)
{
  // USB C Mailbox Interface
  VERS,  16,       // PPM->OPM Version
      ,  16,       // Reserved
  CCI,   32,       // PPM->OPM CCI indicator
  CTRL,  64,       // OPM->PPM Control message
  MSGI, 128,       // OPM->PPM Message In
  MSGO, 128,       // PPM->OPM Message Out
}
```

2.2.4 OPM/BIOS Communication Flow

The communication between OPM and BIOS in any USB Type-C SW flow is primarily through ACPI _DSM method.

OPM will write the command to the shared mailbox Message In and Control fields depending upon the request and invoke the _DSM method for USB Type-C device.

The _DSM does not do any evaluation or processing of the OPM request but just communicates it down to the EC through EC OpRegion.

When BIOS receives a completion notification or an alert from EC, it copies the USB Type-C data from EC OpRegion to the shared mailbox issues ACPI notify 0x80 on USB Type-C device to the OPM driver. On this notification, OPM driver reads the shared mailbox to see the specific alert from PPM.

Depending on BIOS/EC FW implementations, there may not be a contiguous 48 bytes available in EC OpRegion to replicate the contents of shared mailbox as it is in EC OpRegion. In such cases, BIOS/EC FW can spread the USB Type-C data in EC OpRegion, while BIOS maintains the shared mailbox with the format as per UCSI spec.

2.2.5 BIOS/EC Communication Flow

The underlying communication between BIOS and EC will be same as legacy method of using IO ports 0x66/0x62 for BIOS to EC command and using SCI for EC to BIOS alert.

It is recommended avoid interpretation USB Type-C data by BIOS. When sending data to EC, BIOS sends all data marked as OPM->PPM and when receiving data BIOS will receive all data marked as PPM->OPM. Also, BIOS can get static data (such as VERSION) once per boot. It is highly recommended that EC OpRegion bytes used for USB Type-C are exclusive.

Note: *The BIOS-EC OpRegion is a shared mailbox between BIOS and EC in the non-volatile memory that is used for multiple purposes. For the USB Type-C specifically, ACPI BIOS needs to add 48 bytes with the described structure at Table 2-1.*

Example:

```
OperationRegion (ECF2,EmbeddedControl,0,0xFF)
Field (ECF2,ByteAcc,Lock,Preserve)
{
    ...
    VERS,  16,       // PPM->OPM Version
        ,  16,       // Reserved
    CCI,   32,       // PPM->OPM CCI indicator
    CTRL,  64,       // OPM->PPM Control message
    MSGI, 128,       // OPM->PPM Message In
    MSGO, 128,       // PPM->OPM Message Out
    ...
}
```

When BIOS_DSM is invoked due to a write to OPM-BIOS mailbox, the _DSM method does a copy of the Message In, Control and Acknowledge fields to the corresponding EC OpRegion field. Following this, BIOS issues a command to request EC to read the OpRegion fields.

Once EC has processed the request and executed the command, it updates the Message Out and CCI register in the OpRegion with command complete notification back to OPM. To pass this information to the mailbox that OPM will read, the EC issues an SCI to BIOS.

When BIOS SCI handler sees alert or response from USB Type-C, it updates the OPM-BIOS mailbox with the Message Out, CCI and Status fields from the BIOS-EC OpRegion and then notifies the OPM so that OPM can read the mailbox.

Figure 2-3. [OPM->BIOS ->EC] High Level Communication Flow Block Diagram

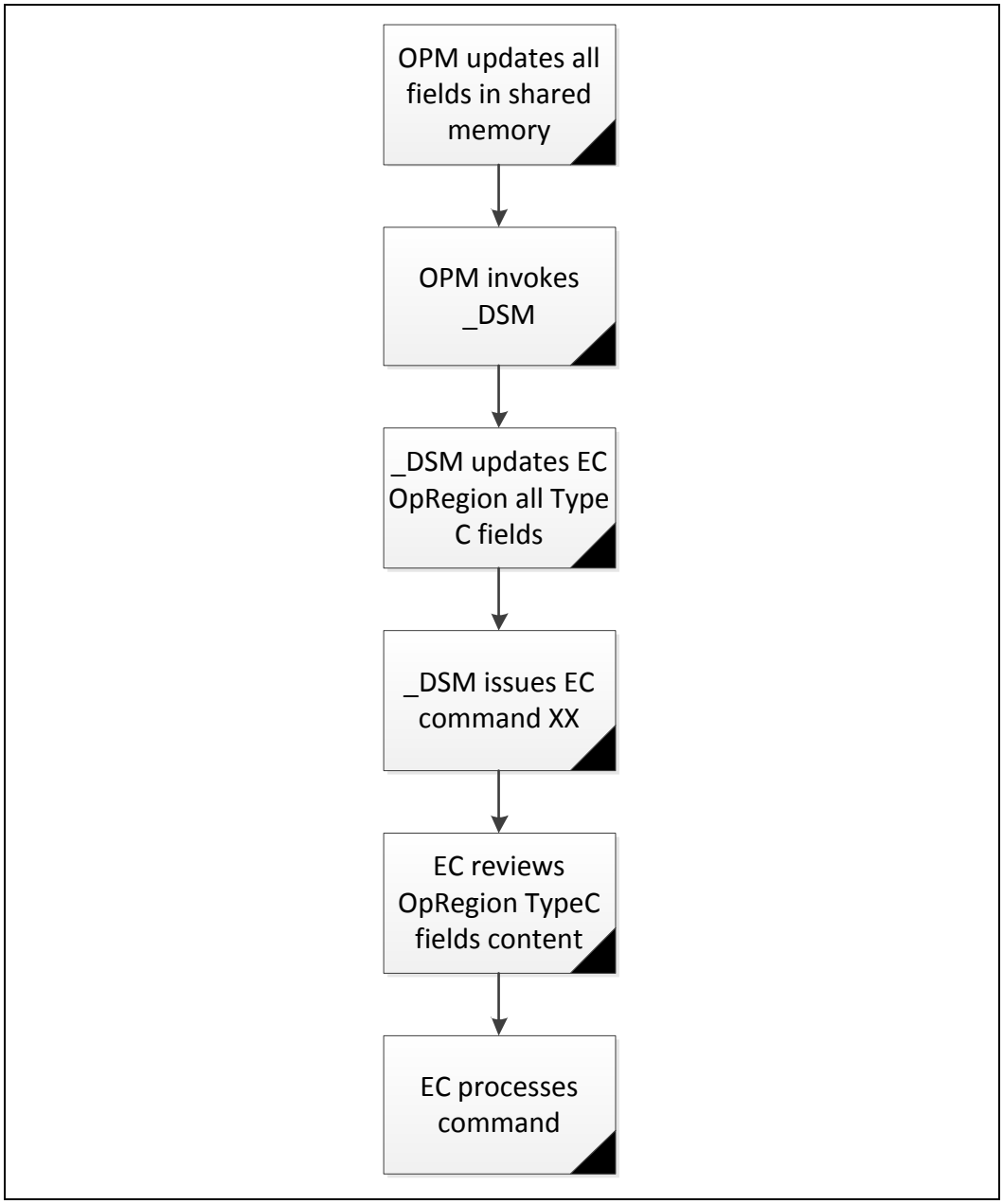
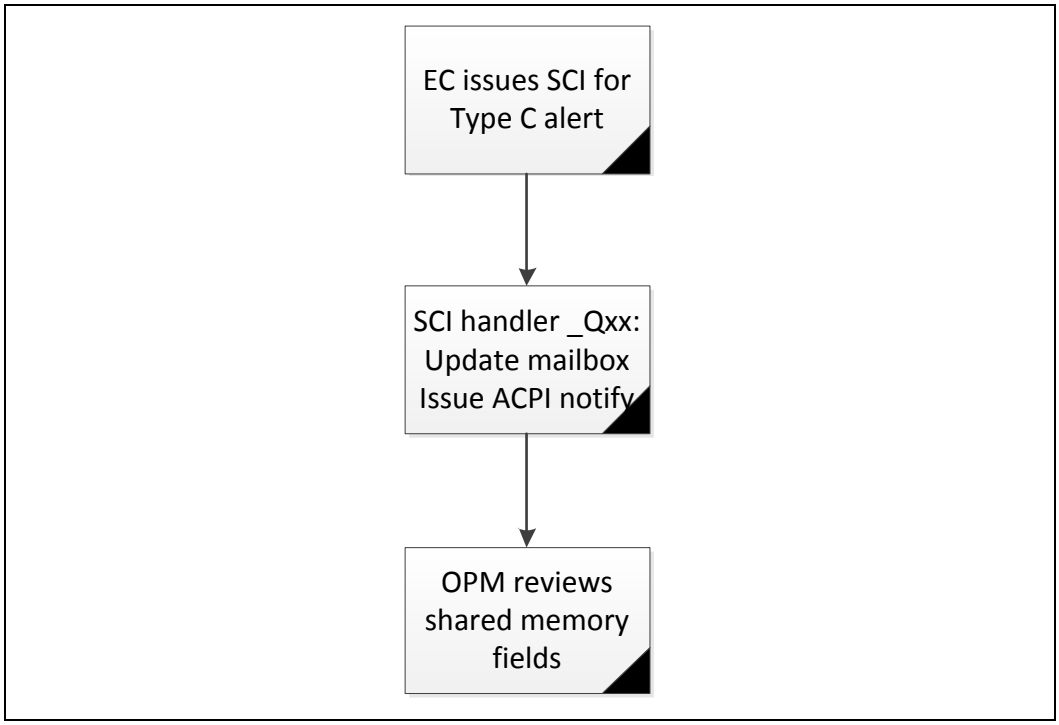


Figure 2-4. [EC->BIOS->OPM] High Level Communication Flow Block Diagram



§ §