AMD 780G Family
BIOS Developer's Guide

For the RS780, RS780C, RS780D, RS780M,
RS780E, RS780MC, and RX781

Technical Reference Manual
Rev. 1.01

P/N: 43734_rs780_bdg_pub_1.01

Trademarks

AMD, the AMD Arrow logo, AMD Athlon, ATI, Mobility, PowerPlay, CrossFire, Radeon, and combinations thereof, are trademarks of Advanced Micro Devices, Inc.

HyperTransport is a licensed trademark of the HyperTransport Technology Consortium.

Microsoft and Windows are registered trademarks of Microsoft Corporation.

Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

Disclaimer

The contents of this document are provided in connection with Advanced Micro Devices, Inc. ("AMD") products. AMD makes no representations or warranties with respect to the accuracy or completeness of the contents of this publication and reserves the right to make changes to specifications and product descriptions at any time without notice. No license, whether express, implied, arising by estoppel, or otherwise, to any intellectual property rights are granted by this publication. Except as set forth in AMD's Standard Terms and Conditions of Sale, AMD assumes no liability whatsoever, and disclaims any express or implied warranty, relating to its products including, but not limited to, the implied warranty of merchantability, fitness for a particular purpose, or infringement of any intellectual property right.

AMD's products are not designed, intended, authorized or warranted for use as components in systems intended for surgical implant into the body, or in other applications intended to support or sustain life, or in any other application in which the failure of AMD's product could create a situation where personal injury, death, or severe property or environmental damage may occur. AMD reserves the right to discontinue or make changes to its products at any time without notice.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 About This Document

This manual contains guidelines for the BIOS and software development of a system that utilizes the AMD 780G family of northbridges. Unless indicated otherwise, the programming information in this document applies to the following 780G variants:

- RS780 (AMD 780G)
- RS780C (AMD 780V)
- RS780D (AMD 790GX)
- RS780E (AMD 780E)
- RS780M (AMD M780G)
- RS780MC (AMD M780V)
- RX781 (AMD M770)

Changes and additions to the previous release of this document are highlighted in red. Refer to *Appendix A* at the end of this document for a detailed revision history.

1.2 Architecture Overview

The RS780 is a high performance system logic chip that supports the simultaneous operation of two external PCI Express graphics cards on an AMD Sempron/Athlon 64/Athlon 64 FX/Rev 10/Rev 11 platform. It supports a four-lane PCI Express interface to the AMD IXP, as well as six ports for external PCI Express devices.

AMD's RS780 provides a bridge between HyperTransport and PCI Express. Both HyperTransport and PCI Express are point-to-point connections. Using point-to-point connections allows much higher bandwidth since the electrical design can be controlled much better than that with a shared bus (such as PCI). From a software perspective both HyperTransport and PCI Express are software compatible with PCI configuration mechanisms.

This page is left blank intentionally.

Chapter 2
Register Space Access

2.1 PCIE Core/Port Indirect Register Space (PCIEIND)

The Core Index Space contains control and status registers that are generic to all PCIE ports in the RS780. This register space is accessed through the following index/data register pair located in the RS780's northbridge PCI configuration registers:

- Index register is bus 0, device 0, register 0xE0.
- Data register is bus 0, device 0, register 0xE4.

Note: Register descriptions are referenced with the name PCIEIND.

The Core Index Space is subdivided into two parts. Since there are a total of 3 PCIE cores in the RS780, hardware has been implemented to provide a mechanism to access these registers either independently or jointly through the programming of bits [18:16] of the index register 0xE0. The encoding is as follows:

- 0xE0[18:16] == 0x1: Reads and writes accesses to 0xE4 will be directed at the per-core index registers in PCIE-GPPSB only.
- 0xE0[18:16] == 0x2: Reads and writes accesses to 0xE4 will be directed at the per-core index registers in PCIE-GPP only.
- 0xE0[18:16] == 0x0: Reads and writes accesses to 0xE4 will be directed at the per-core index registers for PCIE-GFX only.

Note: Register descriptions are referenced with the name PCIEIND or BIFNB.

2.2 PCIE Port Indirect Register Space (PCIEIND_P)

This register space is accessed through the following index/data register pair located in the RS780's northbridge PCI configuration registers:

- Index register is bus 0, device 2/3/4/5/6/7/9/10, register 0xE0[15..0].
- Data register is bus 0, device 2/3/4/5/6/7/9/10, register 0xE4[31..0].

Note: Register descriptions are referenced with the name PCIEIND_P or BIFNBP.

2.3 HTIU Indirect Register Space (HTIUIND)

This register space is accessed through the following index/data register pair located in the RS780's northbridge PCI configuration registers:

- Index register is bus 0, device 0, register 0x94[7..0].
- Data register is bus 0, device 0, register 0x98[31..0]. To write HTIUIND, register 0x94[8] has to be set.

Note: Register descriptions are referenced with the name HTIUIND.

2.4 MISC Indirect Register Space (MISCIND)

This register space is accessed through the following index/data register pair located in the RS780's northbridge PCI configuration registers:

- Index register is bus 0, device 0, register 0x60[7..0].
- Data register is bus 0, device 0, register 0x64[31..0]. To write MISCIND, register 0x60[7] has to be set.

Note: Register descriptions are referenced with the name MISCIND.

2.5 Clock Configuration Register Space (CLKCONFIG)

The RS780 clock control block is located in Device0, Function1. By default, this PCI function does not appear in the PCI configuration space. System firmware can make Device0, Function1 visible by setting NBCONFIG:0x4C[0] to 1. The expected use of the Clock Control function is to only enable it in the PCI configuration space for modification by system firmware during POST. This function does not implement PCI device and vendor ID's. Therefore, system firmware should hide the Clock Control function after making modifications.

Chapter 3
IRQ Mapping

3.1 Integrated GFX

Table 3-1 Integrated GFX Device Bit Setting

Device	Device Bit Setting
5	INTA -> INTC

3.2 PCIE Ports

Table 3-2 PCIE Ports Device Bit Setting

Device	Device Bit Setting
2	INTA -> INTC
3	INTA -> INTD
4	INTA -> INTA
5	INTA -> INTB
6	INTA -> INTC
7	INTA -> INTD
9	INTA -> INTB
10	INTA -> INTC

This page is left blank intentionally.

Chapter 4
UMA Size Consideration

4.1 UMA/System Memory Size

Table 4-1 contains the UMA/System Memory size information for the RS780.

Table 4-1 UMA/System Memory Size

System Memory	UMA Size
Under 256MB	32MB
256MB ~ 512MB	64MB
512MB ~ 1GB	128MB
Above 1GB	256MB

This page is left blank intentionally.

Chapter 5
CIMX RS780 Architecture

5.1 Introduction

The CIMX-RS780 introduces new interface and distribution model to help quickly integrate RS780 Northbridge family support in the customer products.

5.2 Distribution Model

To avoid miscellaneous build issues and simplify integration with different BIOS code bases CIMX-RS780 distributed in form of binary files called **"B1"** and **"B2"**. **B1** contain minimum initialization required for BIOS recovery. **B2** contain full chipset initialization code.

5.3 Integration and Usage Model

B1 image must be integrated in boot block part of the BIOS in uncompressed form to ensure possibility of recovery if the main bios image is corrupted. There are no restrictions on B2 location. **B2** image integrated in main BIOS is in uncompressed form.

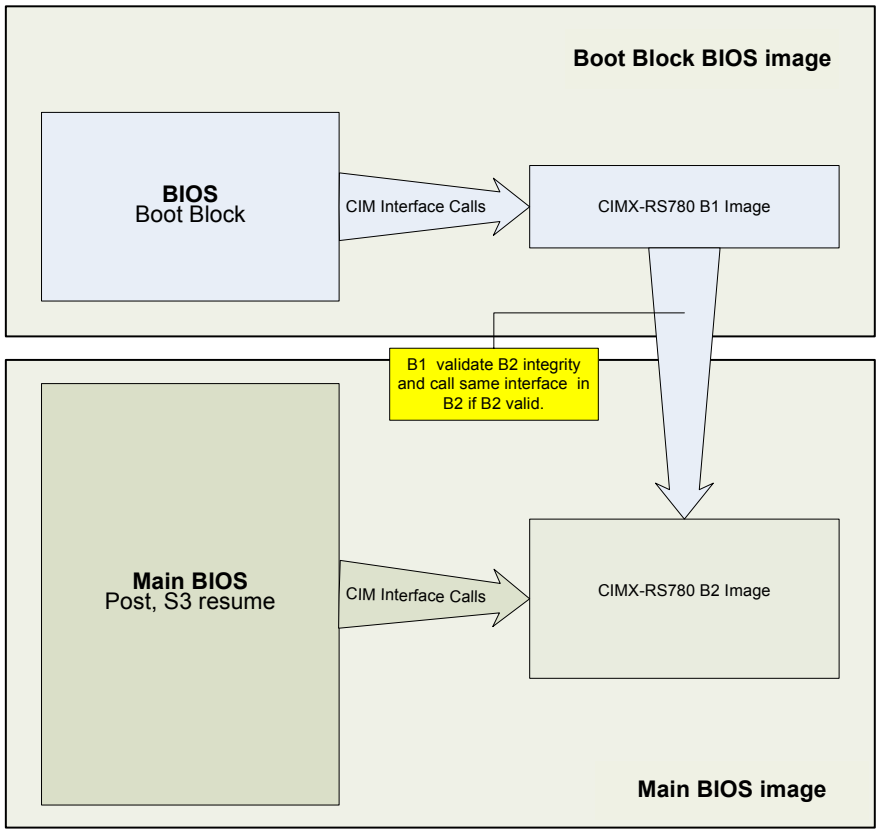

Figure 5-1 Integration And Usage Model

5.4 Flow Chart

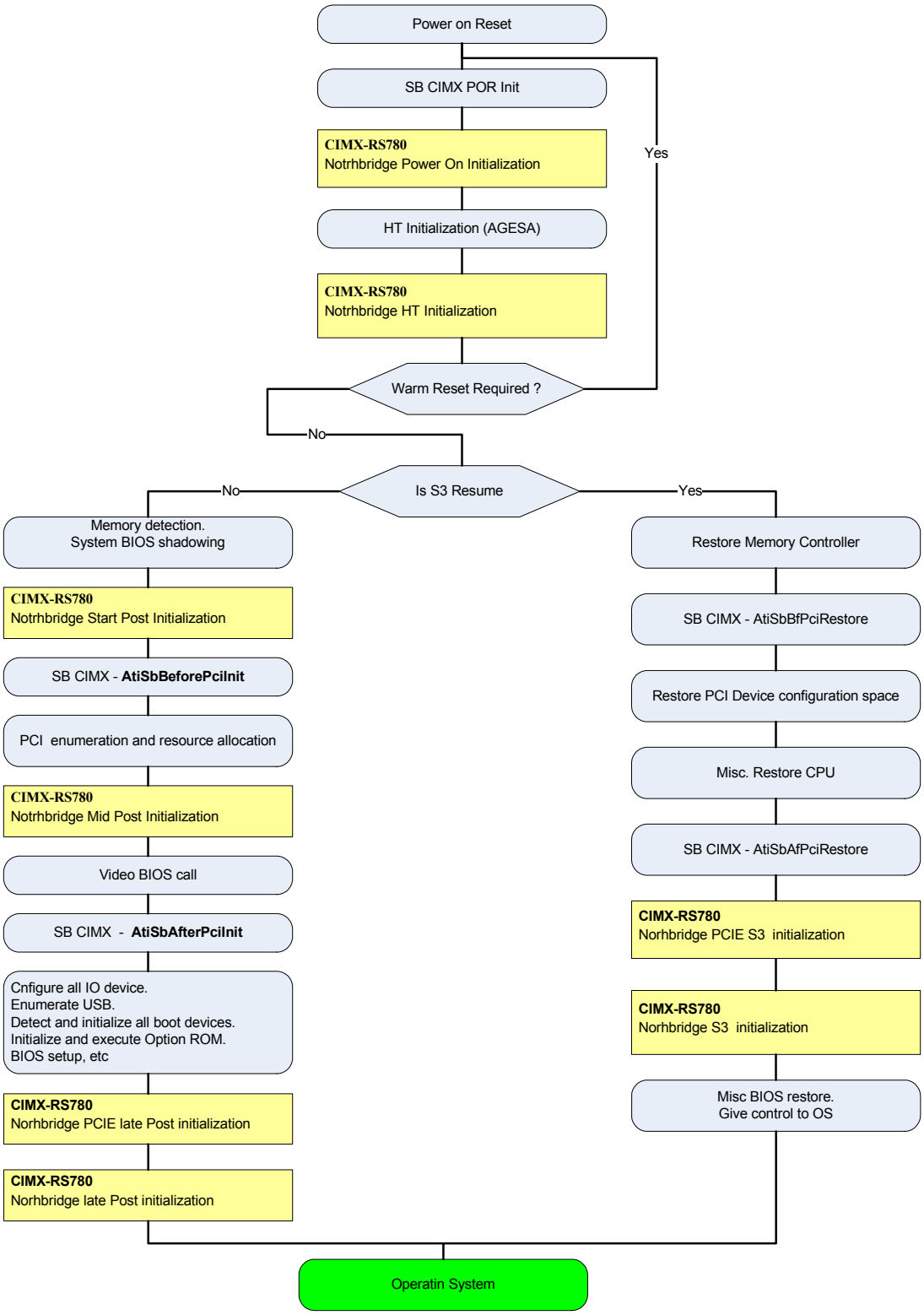

Figure 5-2 RS780 Flow Chart

5.5 Binary File

5.5.1 Overview

The CIMX-RS780 binary file is modified and rebased in 32-bit PE32 executable generated by Microsoft Visual Studio. (For additional information about Microsoft Visual Studio file format refer “Visual Studio, Microsoft Portable Executable and Common Object File Format Specification”). *Figure 5-3* illustrates the difference between Microsoft PE executable format and the CIMX-RS780 binary file format.

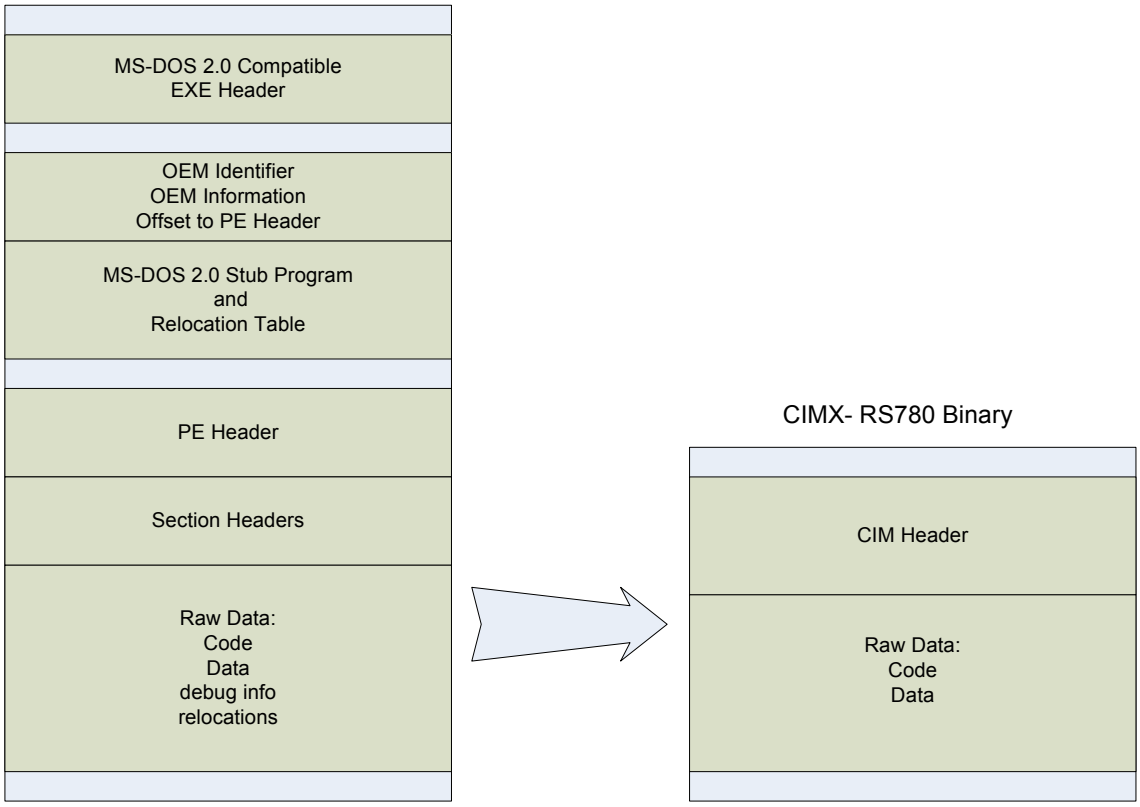

Figure 5-3 CIMX-RS780 Binary File Format

5.5.2 Binary Header

The CIMX-RS780 consists of a file header followed immediately by Raw Code and Data.

Table 5-1 CIM File Header Definition

Offset	Size	Field	Description
0	4	Signature	Signature that identifies this as a CIM module.
4	4	EntryPointAddress	Address of the entry point relevant to the beginning of the binary image.
8	4	ModuleSignature	Signature that identify this particular CIM module.
12	4	ImageSize	Size in number of bytes of complete binary including the header.
16	2	ModuleVersion	Version of the binary module.
18	1	Checksum	Checksum of the binary image.
19	5	Reserved	Space reserved for future use.

5.6 CIM Interface Calls Environment

Prior to calling any CIMX interface, it is required that the following steps are performed:

- Step 1: Place CPU into 32-bit protected mode.
- Step 2: Set CS as 32-bit code segment with Base/Limit – 0x00000000/0xffffffff.
- Step 3: Set DS/ES/SS as 32-bit data segment with Base/Limit – 0x00000000/0xffffffff.

See section *5.10* for a programming example.

5.7 Interface Definition

All interface calls to the CIMX-RS780 binary are C-like calls to the Entry Point of the binary image:

void (*ImageEntryPointPtr)(void* Config)

5.7.1 Northbridge Power-On/Reset Initialization

Upon system power-on, or cold reset, the RS780 is supposed to be self-initialized to a working state without any software interference. However, the CIM may need to program the recommended settings for the purpose of enabling special functions and performance improvement during early initialization.

Functions Documentation:

void (*ImageEntryPointPtr)(STDCFG * Config)

Detail Description:

stdHeader: Standard function call
stdHeader.Func = 0x1

```
typedef struct _STDCFG{
    UINT32          pImageBase;
    UINT32          pPcieBase;
    UINT8           Func;
    UINT32          pCallBack;
    UINT32          pB2ImageBase;
}STDCFG;
```

pImageBase: Physical 32bit address of binary image location.
pPcieBase: Address of PCIE Extended configuration space MMIO. It is required that the PCIE MMIO range be programmed in the CPU Memory Mapped IO Base/Limit Registers.

Table 5-2 Northbridge Power-On/Reset Initialization

Bit	Field	Description
[0..7]	PCIE Extended Configuration size	Size of PCIE extended config space in MB. 0 – 256M for backward compatibility.
[31..20]	PCIE Extended Configuration Base	[31..20] of PCIE extended configuration base address.
Example: 0xE0000000 – PCIE extended Cfg. MMIO 0xE0000000 – 0xEFFFFFFF (256MB) 0xE0000040 – PCIE extended Cfg. MMIO 0xE0000000 – 0xE3FFFFFF (64MB)		

Func: Function identifier.
pCallBack: Address of OEM call back function.
pB2ImageBase: Optional parameter. Physical 32bit address of "B2" image location.

5.7.2 Northbridge HT Initialization

The RS780 requires additional initialization for HT3 support Interface call which needs to be executed prior to applying link speed and width, but after HT link is speed/width fully configured on the northbridge and the CPU side.

Functions Documentation:

void (*ImageEntryPointPtr)(HTCFG* Config)

Parameters:

```
typedef       struct _HTCFG{
    STDCFG    stdHeader;
    UINT32    htConfiguration;
    void*     DebugPtr;
    UINT32    Status;
    UINT32    htConfiguration;
    UINT8     ParentNodeLinkID;
    UINT8     ParentNodeSubLinkID;
    UINT32    clmcConfiguration;
    UINT16    HTRefClkMhZ;
    UINT16    Reserved1;
    UINT64    Reserved2;
    UINT64    Reserved3;
}HTCFG;
```

Detail Description:

stdHeader: Standard function call (see STDCFG definition)
stdHeader.**Func = 0x2.**

htConfiguration: HT miscellaneous configuration options.

Table 5-3 htConfiguration

Bit	Field	Description
[0..1]	LSx	RX HT Link Power management mode 0 – Auto. Recommended 1 – L1s 2 – L2s 3 – Disable
[2..3]	LVM	Low voltage mode 0 - Auto. Recommended 1 - Disable 2 - Enable Set to Enable if platform support voltage scaling
[4..5]	HT3ProtocolChecker	HT Protocol checker 0 – Auto. **Recommended** 1 – Disable 2 – Enable Debug option.
[6..7]	LinkTristate	Link tristate 0 – Auto. Recommended 1 – Disable
[8..9]	Reserved	Reserved for future use. Should be set to 0.
[10..11]	Reserved	Reserved for future use. Should be set to 0.
[12..13]	TxLSx	TX HT Link Power management mode 0 – Auto. Recommended 1 – L1s 2 – L2s 3 – Disable

Table 5-3 htConfiguration

Bit	Field	Description
[14..16]	XbarMode	CPU XBAR optimization specific to RS780 [0] – 1/0 Master bit Optimization enable/disable [1] – 1/0 Optimize for Uma/NonUma [2] – 1: CPU MCLK = 400mhz and HT Speed > 1400MhZ 0: All other case. (Applicable only for GR CPU).
[17..31]	Reserved	Reserved for future use. Should be set to 0.

ParentNodeLinkID: Parent node HT Link ID to which RS780 is connected (0,1,2,3)
ParentNodeSubLinkID: Parent node HT Sub Link ID to which RS780 is connected:
(0,1) If the link is unganged.
(0) If the link is ganged.
clmcConfiguration: CLMC miscellaneous configuration options

Table 5-4 clmcConfiguration

Bit	Field	Description
[0]	CDLC	0 - Disable 1 - Enable
[1]	CDLD	0 - Disable 1 - Enable
[2]	CDLW	0 - Disable 1 - Enable
[3]	CDLF	0 - Disable 1 - Enable
[4]	CILR	0 - Disable 1 - Enable
[5]	CDLR	0 - Disable 1 - Enable
[6..7]	CLMC	0 - Disable 1 - Auto* **Recommended** 2 - Enable
[8]	CDLW_SW	0 - Hardware CDLW 1 - Software CDLW
*CIMX will enable the best possible combination of CLMC apability. All clmcConfiguration fields, except CLMC, will be ignored in this mode.		

HTRefClkMhZ: HT reference clock in MhZ. Acceptable range: 200MhZ - 450MhZ. All other values are treated as 200MhZ.
DebugPtr: Pointer to extended HT debug feature. Recommended setting is 0;
Status: Status returned by this interface call (TBD)

5.7.3 Northbridge PCIE Early Initialization

The RS780 requires initialization of the PCIE block (Ports and Device) prior to PCI bus enumeration and resource allocation. The CIM may also need to program the recommended settings for the purpose of enabling special functions and for performance improvement during PCIE early initialization.

Functions Documentation:

void (*ImageEntryPointPtr)(PCICFG* Config)

Parameters:

```
typedef     struct    _PCICFG{
    STDCFG          StdHeader;
```

```
    void*       DebugPtr;
    UINT32      Status;
    UINT32      eConfiguration;
    UINT32      lConfiguration;
    PORTCFG     PortFeatures[11];
    UINT32      TempMMIO;
    UINT8       GPPConfig;
    UINT8       GFXConfig;
    UINT32      BroadcastBase;
    UINT32      BroadcastSize;
    UINT16      DelayTraining;
    UINT32      Reserved1;
    UINT64      Reserved2;
    UINT64      Reserved3;
}PCICFG;
```

Callbacks:

PCIE OEM call back “CIMCB_PCIE_SLOT_RESET_DEASSERT” (section *5.8.1*)
PCIE OEM call back “CIMCB_PCIE_SLOT_RESET_ASSERT” (section *5.8.2*)
PCIE OEM call back “CIMCB_PCIE_SLOT_RESET_SUPPORT” (section *5.8.3*)
PCIE OEM call back “CIMCB_PCIE_TRAINING_COMPLETE” (section *5.8.4*)

Detail Description:

stdHeader: Standard function call (see STDCFG definition).
stdHeader.Func = 0x3.
eConfiguration: PCIE miscellaneous configuration options.

Table 5-5 eConfiguration

Bit	Field	Description
[0]	PCIE_GFX_LANE_REVERSAL	Lanes reversal configuration for GFX slots (DEV 2/3). The value is Platform specific. 0 – Disable 1 – Enable
[1]	Reserved	Must be set to 0.
[2..3]	Reserved	Must be set to 0.
[4..5]	PCIE_REFCLK_MODE	Select reference clock mode for GFX PCIE core. [4] – 1 Use SB REFCLK for GFX REFCLK, 0 – use external REFCLK for GFX. [5] – 1 - Use SB REFCLK for GPP REFCLK, 0 – use external REFCLK for GPP.
[6]	PCIE_GFX_WORKAROUND_DIS	Misc. GFX cards (ATI/NV) workaround. 0 – Enable. **Recommended.** 1 – Disable.
[7]	PCIE_GFX_LANEPOWEROFF_DIS	Unused lanes power off for GFX/GFX2. 0 – Enable. **Recommended.** 1 – Disable.
[8]	PCIE_GPP_LANEPOWEROFF_DIS	Unused lanes power off for GPP. 0 – Enable. **Recommended.** 1 – Disable.
[9..12]	Reserved	Must be set to 0.
[13..14]	Reserved	Must be set to 0.
[15]	Reserved	Must be set to 0.
[16]	PCIE_LPC_DEADLOCK_WRK_DIS	LPC deadlock workaround. 0 – Enable. **Recommended.** 1 – Disable.
[17..18]	PCIE_CPL_BUFF_ALLOC	Slave CPL buffer allocation. 0 – Auto. **Recommended.** 1 – Disable. 2 – Enable
[19]	PCIE_OVERCLOCKING	PCIE REFCLKC is overclocked. 0- Non overlcocked. **Recommended.** 1 – Overclocked.
[20]	PCIE_GPP2_CONFIG	PCIE GPP core configuration. 0 – Dev9 – x1, Dev10 – x1 1 – Dev9 – x2
[21]	PCIE_NB_SB_HALF_SWING	NB-SB Link Half Swing mode. This feature is applicable for mobile designs with trace length 1-cm between the NB and SB.

lConfiguration: PCIE miscellaneous configuration options.

Table 5-6 lConfiguration

Bit	Field	Description
[0..1]	PCIE_GFX_TX_GATING	Control GFX core TX clock gating 0 – Auto. **Recommended.** 1 – Disable. 2 – Enable.
[2..3]	Reserved	Must be set to 0.
[4..5]	PCIE_GFX_PLL_CONTROL	Debug option. Should be set to 0
[6..7]	Reserved	Must be set to 0.
[8..9]	PCIE_GPP_PLL_CONTROL	Debug option. Should be set to 0

Table 5-6 IConfiguration

Bit	Field	Description
[10..11]	PCIE_GFX_LCLK_GATING	Control GFX core LCLK clock gating 0 – Auto. **Recommended.** 1 – Disable. 2 – Enable.
[12..13]	Reserved	Must be set to 0.
[14..15]	PCIE_GPP_TX_GATING	Control GPP Core TX clock gating 0 – Auto. **Recommended.** 1 – Disable. 2 – Enable.
[16..17]	PCIE_SB_TX_GATING	Control SB Core TX clock gating 0 – Auto. **Recommended.** 1 – Disable. 2 – Enable.
[18..19]	PCIE_GPP_LCLK_GATING	Control GPP Core LCLK clock gating 0 – Auto. **Recommended.** 1 – Disable. 2 – Enable.
[20..21]	PCIE_SB_LCLK_GATING	Control SB Core LCLK clock gating 0 – Auto. **Recommended.** 1 – Disable. 2 – Enable.
[22..23]	PCIE_NB_SB_VC	Enable Non-Posted VC1 traffic on NB-SB link. 0 – Auto. **Recommended.** 1 – Disable. 2 – Enable.

PortFeatures[11]: Per PCIE port configuration options. (PortFeatures[2] – DEV 2, PortFeatures[3] – DEV 3,…,)
PortFeatures[10]: DEV 10, PortFeatures[0]/PortFeatures[1] – Reserved for future use).

Table 5-7 PortFeatures[11]

Bit	Field	Description
[0]	PortPresent	Slot presence for PCIE Port. 0 – Not present 1 – Present.
[1]	PortDetect	Endpoint detection. For internal use. Filled by CIM. 1 – Detected. 0 – Not detected. Should be set to 0 on CIM module call.
[2]	Reserved	Reserved for future use should be set to 0.
[3]	PortHotPlug	Port hotplug control. 0 – Port does not support hotplug. 1 – Port supports hotplug.
[4..5]	PortAspm	Enable ASPM. 0x0 – Disable. 0x1 – Enable L0s. 0x2 – Enable L1 0x3 – Enable L0s and L1.
[6..7]	PortGen2	Control link switch to Gen 2 mode* 0 – Auto (RC Advertise Gen2 + Autonomous switch) 1 – Disable 2 – Software Switch to Gen2 3 – RC Advertise Gen2 only

Table 5-7 PortFeatures[11]

Bit	Field	Description
[8..11]	PortLinkWidth	Set link width. 0x0 – Auto. **Recommended.** 0x1 – x1 0x2 – x2 0x3 – x4 0x4 – x8
[13..14]	PortCompliance	Set port to compliance mode. 0 – Disable. **Recommended.** 1 – Gen2 compliance. 2 – Gen1 compliance. Debug option.
[16..23]	Reserved	Must be set to 0.
[24..31]	PortPowerLimit	Slot power limit, W. (Default recommended value: **75** - for GFX/GFX2 slots DEV2/3 **30 -** for GPP slots DEV 4/5/6/7/9/10
*To support Dynamic Voltage control (see section 5.9.2) PortGen2 should be set to 1 to disable advertising and use PCIE gen2 link speed.		

TempMMIO: Temporary MMIO base address (size of MMIO range no less that 4M). It is required that the PCIE MMIO range be programmed in the CPU Memory Mapped IO Base/Limit Registers.
GPPConfig: GPP core configuration

Table 5-8 GPPConfig

Value	Description
0x0	Auto. Configuration selected by pin strap on the board.
0x1	4:0:0:0:0
0x2	4:4:0:0:0 Dev 4 –x4
0x3	4:2:2:0:0 Dev 4 –x2 , Dev 6 – x2,
0x4	4:2:1:1:0 Dev 4 –x2 , Dev 6 – x1, Dev 7 – x1
0x5	4:1:1:1:1 Dev 4 – x1, Dev 5 – x1, Dev 6 – x1, Dev 7 – x1

GFXConfig: GFX slot layout configuration.

Table 5-9 GFXConfig

Value	Description
[0..7]	Configuration for PCIE Lane 0-3.*
[8..15]	Configuration for PCIE Lane 4-7.*
[16..24]	Configuration for PCIE Lane 8-11.*
[8..15]	Configuration for PCIE Lane 12-15.*

Table 5-9 GFXConfig

Value	Description
*	
GFXConfig [x..y] = 0x0 – Auto. (If slot present enable on Port 2 only all lanes belongs to Port A if slot present enable on Port 3 only lanes [8..15] belongs to Port B, if both [0..7 belongs to Port A and [8-15] belongs to Port B). **GFXConfig [x..y] = 0x80** – Lanes unused **GFXConfig [x..y]** [3:0] – Lane Type 0x01 – Lane Type Port A 0x02 – Lane Type Port B 0x04 – Lane Type DDI1 0x08 – Lane Type DDI2 **GFXConfig [x..y]** [5:4] – Connector Type 0x00 – Single Link DVI 0x01 – Dual Link DVI 0x02 – HDMI 0x03 – Display Port **GFXConfig [x..y]** [7:5] – Misc Config (Mobile design with DDI) 0x00 – DDI in Docking Station and Undocking case 0x01 – DDI out of Docking Station 0x02 – DDI in Docking Station	

BroadcastBase: Broadcast MMIO is required for the Crossfire mode. This memory mapped IO has to be reserved and not used for any other purpose. It has to be appropriately reported to OS as reserved.
BroadcastSize: Broadcast MMIO size in bytes.
DelayTraining: Delay training after deassert port reset(in ms). 2ms is recommended delay.

5.7.4 Northbridge Start Post Initialization

The RS780 requires the initialization of top of memory to enable DMA to system memory. The CIM also may need to program the recommended settings for the purpose of enabling special functions and performance improvement during early initialization. It is mandatory to use the same copy of NBCFG between Start post, Mid post, Late post, and S3 resume initialization. This will allow CIM to exchange data between interface calls if necessary.

Functions Documentation:

void (*ImageEntryPointPtr)(NBCFG* Config)

Parameters:

```
typedef       struct _NBCFG{
    STDCFG              stdHeader;
    void*               DebugPtr;
    UINT32              Status;
    UINT32              nbConfiguration;
    UINT32              gfxConfiguration;
    UINT32              spConfiguration;
    UINT16              UMABase;
    UINT16              UMASize;
    UINT16              SysMemTomLo;
    UINT16              SysMemTomHi;
    UINT16              MCLK;
    UINT16              ECLK;
    SPTBLHEADER*        spOemInitTablePtr;
    UINT8               CState;
    UINT32              SSID;
    UINT32              APCSSID;
    UINT32              GFXSSID;
    UINT32              AUDIOSSID;
```

```
    UINT8           ParentNodeLinkID;
    UINT8           ParentNodeSubLinkID;
    UINT16          AllPcieGfxInfo;
    UINT16          AmdPcieGfxInfo;
    UINT64          Reserved1;
    UINT64          Reserved2;
    UINT64          Reserved3;
}NBCFG;
```

Callbacks:

NB OEM call back ”CIMCB_NB_UMA_STATE” (section *5.8.5*)

Detail Description:

stdHeader: Standard function call (see STDCFG definition).
stdHeader.Func = 0x4.
nbConfiguration: NB miscellaneous configuration options.

Table 5-10 nbConfiguration

Bit	Field	Description
[0..2]	UnitIDClampConfig	UnitID clumping to increase the number of outstanding requests supported by a single device. 0 – Auto. **Recommended.** 1 – Disable. 2 – Unit ID clump for PCIE 3 – Unit ID clump for GFX 4 – Unit ID clump for PCIE & GFX
[3..4]	NBPowerManagment	Misc NB power management features. 0 – Auto. **Recommended.** 1 – Disable. 2 – Enable.
[5]	SetUpCpuMMIO	Set up CPU MMIO 0 – Disable (Platform BIOS sets up MMIO) 1 – Enable. **Recommended**

gfxConfiguration: Miscellaneous GFX configuration.

Table 5-11 gfxConfiguration

Bits	Field	Description
[0]	UMA_ENABLE	UMA mode. 1 – Enable
[1]	SP_ENABLE	SidePort memory support. 1 – Enable See Note.
[2]	AUDIO_ENABLE	HD Audio support. 1 – Enable; **Recommended.**
[3]	SVIEW_ENABLE	Surround View support. 1 – Enable
[4]	FORCE_IGFX_ENABLE	Force enable internal GFX regardless external GFX presence. Debug options. Recommended to be set to 0.
[5..7]	Reserved	
[8]	ADD2_CARD_AUTODETECT	Enables the detection for ADD2 style DDI card.
[9..10]	Reserved	

Table 5-11 gfxConfiguration (Continued)

Bits	Field	Description
[11..12]	SpCalibration	Side Port Memory Calibration Type 00 – Calibration Disabled (Recommended default setting) 01 – Read calibration only 10 – Write calibration only 11 – Read and write calibration
[13..15]	AdjustEighth	Side Port Memory Calibration Parameter 000 – Auto (Recommended default setting) 001 – Reserved 010 – Use 2/8 value for calibration 011 – Use 3/8 value for calibration 100 – Use 4/8 value for calibration 101 – Use 5/8 value for calibration 110 – Use 6/8 value for calibration 111 - Reserved
Note: Only two internal GFX modes support UMA Only and UMA+SP. If SP_ENABLE is set to 1, then UMA_ENABLE must also be set to 1.		

spConfiguration: Miscellaneous sideport memory/configuration.

Table 5-12 spConfiguration

Bits	Field	Description
[0..5]	InterleaveRatio	SP/UMA memory interleave ratio. 0 - Auto. **Recommended**. (Actual ratio will be determined after RS780 performance tuning); 1 – 1:1 2 – 1:3 3 – 1:7 4 – 1:15 5 – 3:5 6 – 3:13 7 – 5:11 8 – 7:9
[6..7]	InterleaveMode	SP/UMA memory interleave mode. 0 - Auto. **Recommended**. 1 – Corse. 2 – Fine.
[8..10]	SPdynamicCtrl	Misc. SP MC power management. 0 - Auto. **Recommended**. 1 – Dynamic CKE 2 – Dynamic CMD 3 – Dynamic CKE 4 – Disabled
[11..12]	ControllerTermination	NB termination. 0 - Auto. **Recommended**. 1 – Disable 2 – Enable
[13..14]	MemoryTermination	Memory termination. 0 - Auto. **Recommended**. 1 – Disable 2 – Enable
[15..16]	SPcmdHold	SP CMD Hold.(Debug feature). 0 - Auto. Recommended. 1 – Disable 2 – Enable
[17..18]	SPdataHold	SP DATA Hold.(Debug feature). 0 - Auto. **Recommended**. 1 – Disable 2 – Enable

Table 5-12 spConfiguration (Continued)

Bits	Field	Description
[19..28]	spInterleaveSize	SP size to be interleaved in MB. 0 – Auto. **Recommended**.

UMABase: Start address of UMA memory in MB(Mast be aligned by 32M). If set to 0x0 and UMA is enabled CIMX will calculate base UMABase = SysMemTomLo – UMASize. If set to 0xffff and UMA is enabled, then CIMX will calculate base UMABase = SysMemTomHi – UMASize.
UMASize: Size of UMA memory in MB. If set to 0x0 CIMX will calculate optimal size to achieve best performance.
SysMemTomLo: Top of memory below 4G in MB.
SysMemTomHi: Top of memory above 4G in MB.
MCLK: MCLK clock index for SideportMemory
0x0: 200MhZ.
0x1: 266MhZ.
0x2: 333MhZ.
0x3: 400MhZ.
0x4: 533MhZ.
0x5: 667MhZ.
0x6 and above: Auto (In the case of SP memory support, maximum clock is supported by SP. In the case of UMA only mode, this parameter is ignored).
ECLK: Internal GFX boot up engine clock in Mhz. If set to 0 CIMX will use optimal engine clock setting.
spOemInitTablePtr: Pointer to Sp configuration table generated by sptblgen.exe (included in CIM package). This is mandatory parameter in case of using SP.
CState: CPU C State support.

Table 5-13 CState

Bit	Field	Description
[2]	C2 supported	1 – C2 supported (should also be set if C1E supported)
[3]	C3 supported	1 – C3 supported (should also be set if C1E supported)

SSID: OEM NB Subsystem/Subvendor ID. (0xffffffff – use same as device/vendor ID, 0x00000000 - use power on default value).
APCSSID: OEM APC Bridge Subsystem/Subvendor ID. (0xffffffff – use same as device/vendor ID, 0x00000000 - - use power on default value).
GFXSSID: OEM Internal GFX device Subsystem/Subvendor ID. (0xffffffff – use same as device/vendor ID, 0x00000000 - use power on default value).
AUDIOSSID: OEM Internal HDMI Audio device Subsystem/Subvendor ID. (0xffffffff – use same as device/vendor ID, 0x00000000 - use power on default value).
ParentNodeLinkID: Parent node HT Link ID to which the RS780 is connected (0,1,2,3)
ParentNodeSubLinkID: Parent node HT Sub Link ID to which the RS780 is connected.
(0,1) If link is unganged
(0) If link is ganged.
AllPcieGfxInfo: Used by CIMX to store GFX card presence info.
AmdPcieGfxInfo: Used by CIMX to store GFX card presence info.

5.7.5 Northbridge Mid-Post Initialization

Based on OEM requirements, CIM will initialize the Integrated GFX device. It is required that this interface call be executed after resources are allocated and decoding enabled for Integrated Graphics device and parent and host bridge. It is mandatory to use the same copy of NBCFG for Start post, Mid post , Late post, and S3 resume initialization. This will allow CIM to exchange data between interface calls if necessary.

Functions Documentation:

void (*ImageEntryPointPtr)(NBCFG* Config)

Parameters:

Northbridge Start Post Initialization (section *5.7.4*)

Callbacks:

NB OEM call back ”CIMCB_NB_GET_VBIOS” (section *5.8.6*)

NB OEM call back ”CIMCB_NB_UPDATE_INTEGRATED_INFO” (section *5.8.7*)

NB OEM call back “CIMCB_NB_DDI_COMMEN” (section *5.8.7*)

Detail Description:

stdHeader – Standard function call (see STDCFG definition).
stdHeader.**Func = 0x5.**

5.7.6 Northbridge PCIE Late Post Initialization

The RS780 late post initialization enables link ASPM, Switch link speed to GEN2 program port capability. It is mandatory to use the same copy of PCIECFG for Late post and S3 resume initialization. This will allow the CIM to exchange data between interface calls if necessary.

Functions Documentation:

void (*ImageEntryPointPtr)(PCICFG* Config)

Parameters:

See Northbridge PCIE Early Initialization (section *5.7.3*)

Detail Description:

stdHeader – Standard function call (see STDCFG definition).
stdHeader.**Func = 0x6.**

Note: Only **stdHeader** needs to be filled out rest of the parameters in PCICFG are not required for this call and should be set to 0.

5.7.7 Northbridge Late Post Initialization

Based on OEM requirements, the CIM will initialize misc. SSID. Also the CIM will do necessary programming to prepare NB to boot to the OS. It is mandatory to use the same copy of NBCFG for Start post, Mid post, Late post, and S3 resume initialization. This will allow CIM to exchange data between interface calls if necessary.

Functions Documentation:

void (*ImageEntryPointPtr)(NBCFG* Config)

Parameters:

See Northbridge Start Post Initialization (section *5.7.4*)

Detail Description:

stdHeader – Standard function call (see STDCFG definition).
stdHeader.**Func = 0x5.**

5.7.8 Northbridge PCIE S3 Resume Initialization

The CIM will do the necessary programming to prepare PCIE to resume from sleep state. Certain northbridge PCIE Port configuration registers need to be restored prior to this call (see Resume from S3 State (section *7.2.2*)). It is mandatory to use the same copy of PCIECFG for Late post and S3 resume initialization. This will allow CIM to exchange data between interface calls if necessary.

Functions Documentation:

void (*ImageEntryPointPtr)(PCICFG* Config)

Parameters:

Detail Description:

stdHeader – Standard function call (see STDCFG definition).
stdHeader.**Func = 0x7.**

See Northbridge PCIE Late Post Initialization (section *5.7.7*)

5.7.9 Northbridge S3 resume Initialization

CIM will do the necessary programming to prepare NB to resume from sleep state. Certain Northbridge PCI configuration registers needs to be restored prior to this call (see Resume from S3 State (section *7.2.2*). It is highly recommended to use same copy NBCFG for Start post, Late post and S3 resume initialization. This will allow CIM to exchange data between interface calls if necessary.

Functions Documentation:

void (*ImageEntryPointPtr)(NBCFG* Config)

Parameters:

See Northbridge Start Post Initialization (section *5.7.4*)

Detail Description:

stdHeader – Standard function call (see STDCFG definition).
stdHeader.**Func = 0x8.**

5.8 Call Back Interface Definition

5.8.1 PCIE OEM Call Back "CIMCB_PCIE_SLOT_RESET_DEASSERT"

This callback function dedicated to deassert PCIE slot reset signal (For platforms that use GPIO for slot reset).

Functions Documentation:

UINT32 (*pCallBack)(UINTN Param1, UINTN Param2, PCICFG* pConfig)

Parameters:

Param1: Function identifier 0x102.
Param2: Port bit map to deassert reset. BIT2 – Port2, BIT3 – Port3, … etc. For example: Param2 = 0x1c – request deassert reset on Port2/3/4.
pConfig: See Northbridge PCIE Early Initialization (section *5.7.3*)

Return Values:

CIM_SUCCESS (0x00000000): If platform has dedicated GPIO reset for port and support port reset logic for at least one requested port.
CIM_UNSUPPORTED (0x80000001): If platform has does not have GPIO reset for any of requested ports.

5.8.2 PCIE OEM Call Back "CIMCB_PCIE_SLOT_RESET_ASSERT"

This callback function dedicated to deassert PCIE slot reset signal (For platforms that use GPIO for slot reset).

Functions Documentation:

UINT32 (*pCallBack)(UINTN Param1, UINTN Param2, PCICFG* pConfig)

Parameters:

Param1: Function identifier 0x103.
Param2: Port bit map to assert reset. BIT2 – Port2, BIT3 – Port3, … etc. For example Param2 = 0x1c – request assert reset on Port2/3/4.
pConfig: See Northbridge PCIE Early Initialization (section *5.7.3*)

Return Values:

CIM_SUCCESS (0x00000000): If platform has dedicated GPIO reset for port and support port reset logic for at least one requested port.
CIM_UNSUPPORTED (0x80000001): If platform has does not have GPIO reset for any of requested ports.

5.8.3 PCIE OEM Call Back "CIMCB_PCIE_SLOT_RESET_SUPPORT"

This callback function dedicated to deassert PCIE slot reset signal (For platforms that use GPIO for slot reset).

Functions Documentation:

UINT32 (*pCallBack)(UINTN Param1, UINTN Param2, PCICFG* pConfig)

Parameters:

Param1: Function identifier 0x104.
Param2: Port bit map to test if port has GPIO reset capability. BIT2 – Port2, BIT3 – Port3, … etc. For example Param2 = 0x10 – request assert reset on Port4.
pConfig: See Northbridge PCIE Early Initialization (section *5.7.3*)

Return Values:

CIM_SUCCESS (0x00000000): If port has GPIO reset capability.
CIM_UNSUPPORTED (0x80000001): If port does not have GPIO reset capability.

5.8.4 PCIE OEM Call Back "CIMCB_PCIE_TRAINING_COMPLETE"

This callback function called after port training completed and device detected. This can be used to force to disable a device).

Functions Documentation:

UINT32 (*pCallBack)(UINTN Param1, UINTN Param2, PCICFG* pConfig)

Parameters:

Param1: Function identifier 0x101.
Param2: Port number. DEV2 – 0x2, DEV3 – 0x3 ….
pConfig: See Northbridge PCIE Early Initialization (section *5.7.3*)

Return Values:

CIM_SUCCESS (0x00000000): Keep port enable.
CIM_DISABLEPORT (0x80000002): Force port disable/hide/poweroff.

5.8.5 NB OEM Call Back "CIMCB_NB_UMA_STATE"

This callback function is called after the UMA configuration is validated. This call can be used to save UMA state (for example, in NVS). For instance, it can be used on following boot to determine CPU correct setting for CpuBufferOptimization (see Northbridge HT Initialization (section *5.7.2*)).

Functions Documentation:

UINT32 (*pCallBack)(UINT32 Param1, UINT32 Param2, PCICFG* pConfig)

Parameters:

Param1: Function identifier 0x201.
Param2: 0=PCI reset not required. 1=PCI reset required.
pConfig: See Northbridge Start Post Initialization (section *5.7.4*)

Return Values:

CIM_SUCCESS (0x00000000).

5.8.6 NB OEM Call Back ”CIMCB_NB_GET_VBIOS”

This callback function called after to get image of option ROM for integrated GFX for Surround View initialization. It is required to execute the special VBIOS post prior to returning the pointer to the VBIOS image on mobile platforms. The special VBIOS post can be executed by calling offset 64H of the VBIOS with ax = 128h.

Functions Documentation:

UINT32 (*pCallBack)(UINTN Param1, void** Param2, PCICFG* pConfig)

Parameters:

Param1: Function identifier 0x202.
Param2: Pointer on pointer to which needs to be initialized physical address location of uncompressed VBIOS image.
pConfig: See Northbridge Start Post Initialization (section *5.7.4*)

Return Values:

CIM_SUCCESS (0x00000000).

5.8.7 NB OEM Call Back ”CIMCB_NB_UPDATE_INTEGRATED_INFO”

This callback function called after to get image of option ROM for integrated GFX for Surround View initialization.

Functions Documentation:

UINT32 (*pCallBack)(UINTN Param1, ATOM_INTEGRATED_SYSTEM_INFO_V2* Param2, PCICFG* pConfig)

Parameters:

Param1: Function identifier 0x203.
Param2: Pointer to ATOM_INTEGRATED_SYSTEM_INFO_V2 (see CIMRS780.INC for the ATOM_INTEGRATED_SYSTEM_INFO_V2 definition)
pConfig: See Northbridge Start Post Initialization (section *5.7.4*)

Return Values

CIM_SUCCESS (0x00000000).

5.8.8 NB OEM Call Back “CIMCB_NB_DDI_COMMEN”

This callback function called to sert state of COMM_EN signal for ADD2 card detection.

Functions Documentation:

UINT32 (*pCallBack)(UINTN Param1, UINTN Param2, NBCFG* pConfig)

Parameters:

Param1: Function identifier 0x205.
Param2: 1/0 - Assert/Deassert COMM_EN signal
pConfig: See Northbridge Start Post Initialization (section *5.7.4*)

Return Values:

CIM_SUCCESS (0x00000000).

5.9 Integrated Info Table

5.9.1 Overview

The integrated info table is used to communicate platform requirements to the VBIOS/GFX driver. In most instances the CIM will automatically fill the integrated info table. However, some parameters needs to be directly adjusted (see section *5.8.7*) based on platform requirements. The integrated info table defined (ATOM_INTEGRATED_SYSTEM_INFO_V2) in CIMRS780.INC.

5.9.2 Voltage Control

To enable PWM voltage control usNumberOfCyclesInPeriod/usMaxNBVoltage/usMinNBVoltage/usBootUpNBVoltage parameters needs to be filled up. Note that the value for this parameters depend on platform implementation PWM voltage control logic. **Also note that the CIMX will request the driver to stay at max voltage at all times if PCIE Gen2 capability is enabled for any port.**

The following is an example for typical dual voltage level control (1.0V – 1.1V) implementation:

- usNumberOfCyclesInPeriod = 0x8019
- usMaxNBVoltage =0x1A
- usMinNBVoltage = 0x00
- usBootUpNBVoltage = 0x1A

5.10 Example (Northbridge HT Initialization)

This is an example of the call Northbridge HT Initialization interface (section *5.7.2*). To help understand the **B1** and **B2** operation it is assumed this call is executed from boot block.

```
STDHEADER       STRUC
        pImageBase              DD      0
        pPcieBase               DD      0
        Func                    DB      0
        pCallBack               DD      0
        pB2ImageBase            DD      0
STDHEADER       ENDS

HTCONFIGINFO    STRUC
        stdhdr                  DB      (sizeof STDHEADER) DUP (0)
        htConfiguration         DD      0;
        DebugPtr                DD      0;
        Status                  DD      0;
        ParentNodeLinkID        DB      0;
        ParentNodeSubLinkID     DB      0;
HTCONFIGINFO    ENDS

GDT_PTR         LABEL   FWORD
        DW      (offset  GDT_End - offset GDT_Start)
        DD      0f0000h + offset GDT_Start
GDT_Start                       LABELDWORD
        DD      000000000h,00000000h
        DD      00000ffffh,00CF9300h    ; Data/Stack 0x00000000 - 0xffffffff
        DD      00000ffffh,00CF9B00h    ; Code 0x00000000 - 0xffffffff
        DD      00000ffffh,0000930ch    ; RM Stack 0xc0000 - 0xcffff
        DD      00000ffffh,00009b0fh    ; RM Code  0xf0000 - 0xfffff
GDT_End         LABEL   DWORD

OEM_RD780_HTInit PROC   NEAR
```

```
        Local Config: HTCONFIGINFO
        xor     esi, esi
        xor     eax, eax
        mov     eax, ss
        shl     eax, 4
        lea     si, HTConfig
; Setup pointer to Config: HTCONFIGINFO
        add     esi,  eax
        mov     (STDHEADER ptr HTConfig.stdhdr).Func, 02h
        mov     ax,  cs
        movzx   eax, ax
        add     eax, offset  OEM_RS780_HTInit_CallBack
        mov     (STDHEADER ptr HTConfig.stdhdr).pCallBack, eax
        mov     (STDHEADER ptr HTConfig.stdhdr).pImageBase,\ _CIMRS780_B1_Image_Address ;
                Porting required
        mov     (STDHEADER ptr HTConfig.stdhdr).pB2ImageBase,\ _CIMRS780_B2_Image_Address ;
                Porting required
        mov     (STDHEADER ptr HTConfig.stdhdr).pPcieBase,\ PCIE_BASE_ADDRESS ; Porting required
        xor     eax, eax
        mov     HTConfig.htConfiguration, eax
        mov     HTConfig.ParentNodeLinkID, al
        mov     HTConfig.ParentNodeSubLinkID, al
        call    CIM_CallBB
        ret
OEM_RS780_HTInit        ENDP
; ---------------------------------------------------------------------------
; ESI - Config
; ---------------------------------------------------------------------------
CIM_CallBB      PROC  NEAR
        pushad
        push    ds
        push    es

; Assume ds is flat mode segment 0-4G and A20 is enabled
        xor     eax, eax
        xor     edi, edi
        mov     ds, ax
        xor     eax, eax
        xor     edi, edi
        mov     ax, ss
; Save real mode SS stack segment
        push    ax
        shl     eax, 4
; Save real mode stack pointer
        mov     di, sp
        add     di, 2
        push    di                  ; Save RM stack pointer
; Save pointer to interface
        push    esi                 ; Pass ptr to config structure to binary
        movzx   edi, sp
        add     edi, eax
        mov     esp, edi
; Load GDT
        lgdt    FWORD PTR GDT_PTR
; Go to protected mode
        mov     eax, CR0
```

```
        or      al, BIT0
        mov     CR0, eax
        jmp     $+2
; Set 32 bit data/stack selectors
        mov     ax, 8
        mov     ds, ax
        mov     es, ax
        mov     ss, ax

; Retrive  image entry point address
        mov     eax, [ESI].STDHEADER.pImageBase
        add     eax, [eax+4]        ; eax - Image entry Point

; Set 32 bit code selectors
        db      066h, 0EAh
        dd      0f0000h + offset PM32Entry
        dw      010h
PM32Entry:
        jmp $+2
; Call  CIMX-RS780 image entry point
        call    ax                  ; It is "call eax"   at this moment
        add     sp, 4               ; It is "add esp, 4" at this moment
        pop     edi                 ; It is "pop di" at this moment
        pop     ecx                 ; It is "pop cx" at this moment
; Set real mode selector for CS
        db      066h, 0EAh
        dw      offset PM16Entry
        dw      020h
PM16Entry:
; Setup real mode selector for SS
        mov     bx, 18h
        mov     ss, bx              ; Fix up ss limit

; Return to the real mode
        mov     eax, CR0
        and     al, NOT BIT0
        mov     CR0, eax
        jmp     $+2
; We are in real mode
        db      0EAh
        dw      offset RMEntry
        dw      0f000h
RMEntry:
; Restore realmode SS and SP
        mov     ss, cx
        mov     sp, di

SkipImageCall:
        pop     es
        pop     ds
        popad
        ret
CIM_CallBB      ENDP
```

Chapter 6
SurroundView Feature

6.1 SurroundView Feature (Dual Display)

SurroundView (Dual Display) is technology that provides support of multiple traditional CRT monitors and flat panel displays by turning on both internal graphics and external PCIE graphics devices under OS. When the SurroundView feature is enabled, the SBIOS sets the internal graphics as the secondary display device, while keeping the external PCIE graphics device as the primary display device. Note that the primary graphics device should be and AMD PCIE graphics card.

Note: To enable CIMX Surroundview support see Northbridge Start Post Initialization (section *5.7.4*), and NB OEM call back ”CIMCB_NB_GET_VBIOS” (section *5.8.6*).

This page is left blank intentionally.

Chapter 7
Power Management

7.1 Cx State Power Management

TBA

7.2 Sx State Power Management

During certain Sx states (such as S1) the system maintains power to the RS780 and all register settings are retained. Therefore, no action is required when the system resumes. However, under other S states (such as S3) typical designs will shut down power to the RS780 and all of the register settings will be lost.

The SBIOS is responsible for reinitializing all of the RS780 registers. Reinitializing the RS780 registers after S3 can be done by either restoring the tables of register settings which were saved to non-volatile storage prior to entering S3, or by initializing all registers as if from cold.

For most configurations restoration by table is not practical. The recommendation is to re-execute the initialization code.

The remaining NB registers are covered in section *7.2.1*.

7.2.1 Register Restoration

The registers listed in this section are restored after the system memory is made available by bringing the system RAM out of self refresh mode.

On RS780 designs the main system memory is controlled by the memory controller in the AMD CPU. (For more information on saving and restoring the memory controller settings, along with entering and exiting self refresh mode, refer to the “AMD Kernel BIOS writers guide”).The reference BIOS saves the registers listed in this section prior to entering S3. It then restores them after system memory is available on resume from S3.

Note: The list of registers in the following sections is subject to change. If an OEM customer changes any other registers in the RS780 during POST, then they must also be restored on an S3 resume.

- Integrated Graphics Device (1:5:0)
  - 0x10 - 0x13
  - 0x14 - 0x17
  - 0x18 - 0x1B
  - 0x24 - 0x27
  - 0x04 - 0x06
- APC Bridge (0:1:0)
  - 0x18 - 0x1B
  - 0x20 - 0x23
  - 0x24 - 0x27
  - 0x2C - 0x2f
  - 0x04 - 0x06
- Northbridge (0:0:0)
  - 0x04 - 0x06
  - 0x4C - 0x4f
  - 0x8C - 0x8f

- PCIE Bridges (0:2/3/4/5/7/9/10:0)
  - 0x18 - 0x1B
  - 0x20 - 0x23
  - 0x24 - 0x27
  - 0x2C - 0x2f
  - 0x04 - 0x06

7.2.2 Resume from S3 State

When the system wakes up from S3 the BIOS is initially executed from ROM, once it is determined this is an S3 resume. The SBIOS is responsible for restoring the system memory controller's configuration as outlined in the "AMD Kernel BIOS Writer's Guide".

Once the system memory is available the BIOS should then restore the RS780 registers that were saved prior to entering S3.

7.3 S3 PCIe Graphics BAR Save/Restore Issue Workaround

Below is the full detail and requirements for the workaround documented in the product advisory PA_RSx80 ("RV6xx/M8x and RS780/RS880 Black Screen Hang During S3 Testing").

7.3.1 BAR Restoration Process and Architecture

1 The BAR registers will only be guaranteed to be restored after the System BIOS CIMx code has completed with the PCIe link training sequences. In addition, the command registers cannot be restored to the correct value because the OS will set those registers to value 0 prior to S3.

2 Since the operating system has the ability to re-allocate the BAR windows of PCIe bridges, and because any general purpose port (GPP) or graphics core root complex bridge may support one or more graphics devices including hot-plug support, the complete BIOS solution must scan for and be aware of the graphics devices that are available behind any of the available PCIe bridges (not just bus 0, device 2, function 0). Code must save/restore BARs for any PCIe bridge device that could possibly support graphics and detect any graphics device (by scanning for and testing the class code of that device) behind the bridge.

3 For designs in which different configurations can be productized or in which hybrid graphics can be disabled, the bus/device/function address of the end point device may change due to the hiding of the other resources or unused devices. Some designs may also implement hardware (e.g., PCIe switches) which would alter the location of the end point device. Therefore, the SBIOS must scan for the location of the graphics device behind any bridge rather than assuming it will always be at the same location.

7.3.2 Post Time

- BIOS kernel programs BAR registers. Nothing to save/restore related to S4 or reboot.
- RAM and device registers are cleared on every reboot or S4 and the BIOS is re-posted.
- Use the RS780 host device NBMisc register 0x0C to get the bit map of hidden PCIe ports to avoid saving registers for those bridges or devices. Save this map.
- In late post, after the CIMx wrapper code functions were already called, the SBIOS must scan the subordinate bus numbers for the graphics devices, check that the link status register 0xA5 value is greater than or equal to 0x10, and verify that the class code matches that for a graphics device as defined by the PCI specification. The BIOS should not assume that these devices are at specific addresses (bus/device/function) since PCIe endpoint addresses will move around depending on hardware and feature configuration settings.
- Save a corresponding map in System Management mode (SMM) memory of the bus/dev/func of each graphics devices behind each bridge (port) that is found during post time. Do not include hot plug ports in this scan or map since, in the hot plug case, the correct BIOS implementation will result in the operating system calling a different code path (Vista multi-level resource rebalancing triggered by ACPI BIOS _DSM method) to restore endpoint registers on S3 resume that will always consistently work as long as the bridge BAR registers are configured as

expected. For a detailed explanation on how Microsoft Windows Operating Systems handle the PCI and PCIe BAR resources, reference the Microsoft white-paper at http://www.microsoft.com/whdc/system/bus/pci/resources.mspx.

- When the operating system is initializing ACPI in late post, it makes a call to the BIOS and the BIOS is required to save the pcicfg Command Register for all un-hidden PCIe root complex bridges.
- When the operating system is initializing ACPI, it makes a call to the BIOS and the BIOS is required to save the pcicfg Command Register for all non-hot plug PCIe endpoint graphics devices.
- The BIOS is required to configure an I/O-based System Management Interrupt-trap of the SLP_TYP register as explained in the SB700 Register Programming Requirements documentation. The Sleep trap will trigger on any Sleep state transition. The SMI handler must disable the trap once it is triggered and BIOS ASL code must only set the SLP2SMI enable bit for that trap when the PTS method for the desired Sleep State is called by the operating system.

7.3.3 When Entering S3

- The BIOS is required to enable the I/O trap of the SLP_TYP register as explained in the SB700 Register Programming requirements documentation by writing to the SLP2SMI enable register in the _PTS ACPI method for S3 sleep state.
- When the operating system writes the SLP_TYP register, it will be enabled to trigger an SMI. The SBIOS must have implemented a handler that calls a specific function for the S3 sleep case. The main sleep2smi handler (already implemented in Phoenix BIOS MISER code) should direct the calls for the S0-S3 transition to the associated correct sleep type handler routines. Normally, this functionality is already supplied by the BIOS vendor code for AMD Chipsets and the ODM must only implement a subordinate helper routine for the S0-S3 state. For example, in Phoenix BIOS, these routines are called REGISTER_ACPI_FN SLEEP_TRAP32 functions.
- The SBIOS saves the 32-bit command register offset 0x04 (in the same place that was done previously during POST) for the bridge but ignores the lower byte of the register read back at this time (the operating system will have sometimes set it to 0) and only saves the upper 3 bytes in the DWORD location in memory. The data for the lower byte of that register that was previously saved during post time is not altered.
- The SBIOS uses the previously saved maps to scan for and save the PCIe Bridge BARs (registers as defined for PCICFG Type 1 header) in a table in memory. It also checks behind the bridge for any graphics device as per the previously determined PFAs (bus/dev/func) expected in that slot (port).
- If there is a graphics device behind the bridge, then the PCICFG Type 0 BAR register locations need to be saved in their own unique table (one for each possible graphics device behind each possible bridge).
- If any registers read back 0xFFFF then the expected device was powered down and the registers should NOT be saved.

7.3.4 When Resuming from S3

PCIe Root Complex bridges are restored by CIMx code such that any PCIe links with PCIe endpoint devices behind the bridges must be “re-trained” prior to any access to either the bridge BAR or endpoint BAR registers. The sequence of suggested code calls in the S3 resume path must occur as follows:

1 InitPcieAtEarly

- **a** Call CIMx function NB_INIT_PCIE_EARLY (Medium priority)

2 RestoreRS780S3Config (Low priority, this must happen only after the CIMx PCIE_EARLY call)

- **a** RestoreNBConfig (restore registers for RS780 host (bus0, dev0, func0))
- **b** RestoreAGPConfig (restore registers for IGP bridge (if powered))
- **c** RestorePCIEBridgeConfig (restore registers for S3 Pcie BAR workaround here)
- **d** RestoreOtherPCIConfig (restore registers for any special on-board fixed PCI (not PCIe) devices)
- **e** oemCustomRestorePcieDevices (restore registers for any special on-board Pcie devices)

3 InitNbS3ResumeFar (Lowest priority, this must happen only after the RestoreRS780S3Config)

- **a** Call CIMx function NB_INIT_PCIE_S3

b Call CIMx function NB_INIT_NB_S3

The S3 resume time PCIe BAR workaround register restore is implemented in item 2c) which corresponds to the functionality of RestoreRS780S3Config → RestorePCIEBridgeConfig as follows:

- Get the hidden bridge bits from NBMisc REG0x0C. The bits that are clear indicate active PCIe bridges for which the BARs must be restored with corresponding data that was previously saved during the S3 sleep trap time.
- Test BIT2. If BIT2 is set (port is disabled/hidden) then skip to test the next port (BIT3).
- A generic common register restore routine is suggested. This routine should not attempt to restore any registers for a specified PCIe device or bridge address if the Vendor/Device ID register (offset 0x00 in the PCI Configuration space) reads back 0xFFFFFFFF) which indicates that the device or bridge is not present.
- If BIT2 = 0, then restore the 32-bit registers for PCI address 80001000h (Bus0, Dev2, Func 0) in this order: offset 0x18, 0x1c, 0x20, 0x24, 0x2Ch, 0x30, 0x3Ch, 0x04. These registers correspond to the PCI specification definition for the Type 1 PCI Configuration space header (used for PCI-to-PCI Bridges corresponding to each PCIe root complex port).
- After restoring the PCIe root complex bridge registers for a particular port, check the previously stored (from post time) bit map and table of the end-point device PCI addresses (bus/device/function number) to determine if there is expected to be a graphics device behind this bridge.
- If there was a corresponding post time graphics device PFA previously saved for the expected device behind this port, then proceed to restore the previously saved 32-bit BAR and Command registers for that PCI device address. The registers must be restored in this order: offset 0x10, 0x14, 0x18, 0x1C, 0x20, 0x24, 0x04. These registers correspond to the PCI specification definition from the Type 0 PCI Configuration space header (used for PCIe endpoint devices).
- The corresponding PCI address for each of the RS780 root complex bridge ports is as follows:

  nbmisccfg 0x0C BIT2 = 0, the PCI address is 8001000h (Bus 0 Dev 2 Fn 0).

  nbmisccfg 0x0C BIT3 = 0, the PCI address is 8001800h (Bus 0 Dev 3 Fn 0).

  nbmisccfg 0x0C BIT4 = 0, the PCI address is 8002000h (Bus 0 Dev 4 Fn 0).

  nbmisccfg 0x0C BIT5 = 0, the PCI address is 8002800h (Bus 0 Dev 5 Fn 0).

  nbmisccfg 0x0C BIT6 = 0, the PCI address is 8003000h (Bus 0 Dev 6 Fn 0).

  nbmisccfg 0x0C BIT7 = 0, the PCI address is 8003800h (Bus 0 Dev 7 Fn 0).

  nbmisccfg 0x0C BIT9 = 0, the PCI address is 8004800h (Bus 0 Dev 9 Fn 0).

  nbmisccfg 0x0C BIT10 = 0, the PCI address is 8005000h (Bus 0 Dev 0x0A Fn 0).
- Repeat the steps for all of the remaining root complex bridge ports BITn (if set then go to Test next port BITn+1) where n = 3, 4, 5, 6, 7, 9, 10. Note that port 8 is reserved (used for Northbridge-Southbridge communication) and should not have any associated code to save or restore registers.

Appendix A
Revision History

A.1 Rev 1.01 (August 2009)

- Modified cover title.
- Added marketing names to the variants in section 1.1.

A.2 Rev 1.00 (July 2009)

- First public release based on OEM release 1.07.

This page intentionally left blank.