

TCG Itanium Architecture Based Server Specification

Version 1.0
March 6, 2006
For TPM Family: 1.1b or 1.2

Contact: admin@trustedcomputinggroup.org

TCG

TGC Published
Copyright © TCG 2004 - 2006



Disclaimer

THIS SPECIFICATION IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE. Without limitation, TCG disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification and to the implementation of this specification, and TCG disclaims all liability for cost of procurement of substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, direct, indirect, or special damages, whether under contract, tort, warranty or otherwise, arising in any way out of use or reliance upon this specification or any information herein.

No license, express or implied, by estoppel or otherwise, to any TCG or TCG member intellectual property rights is granted herein.

Except that a license is hereby granted by TCG to copy and reproduce this specification for internal use only.

Contact the Trusted Computing Group at techquestions@trustedcomputinggroup.org for information on specification licensing through membership agreements.

Any marks and brands contained herein are the property of their respective owners.

Table of Contents

1 Introduction..... 4

 1.1 Key words 5

 1.2 Statement Type 6

2 Context 7

3 Boot Flow 8

 3.1 The Core Root of Trust for Measurement (CRTM) 8

 3.2 Firmware Structure and Boot Sequence 9

 3.2.1 Requirements for Itanium Architecture Based Servers which Implement PBB 11

 3.2.2 Requirements for Itanium Architecture Based Servers which do not Implement PBB 12

4 Requirements..... 13

5 PCR Usage and Event Logging 14

 5.1 Locality..... 15

 5.2 Pre-Boot PCR Usage..... 16

 5.2.1 PCR[0] – CRTM, System Firmware, and System Extensions 16

 5.2.2 PCR[1] – System Configuration 16

 5.2.3 PCR[2] – Firmware Driver Code 16

 5.2.4 PCR[3] – Firmware Driver Configuration and Data 16

 5.2.5 PCR[4] – Initial Program Load (IPL) Code..... 16

 5.2.6 PCR[5] – IPL Configuration and Data 16

 5.2.7 PCR[6] – State Transition 17

 5.2.8 PCR[7] – Reserved 17

 5.2.9 PCR[8] – PCR[15] Reserved 17

 5.2.10 Unusual Pre-boot Actions 17

 5.2.11 Runtime Actions and PCR Usage..... 17

 5.3 Event Logging..... 18

 5.3.1 Event Types 18

 5.3.2 Platform Specific Tagged Event Log Structure 19

 5.3.3 EV_ACTION Event Types..... 20

6 TPM 21

7 TBB 22

8 Glossary 23

9 Bibliography..... 24

1 **1 Introduction**

2 ***Start of informative comment:***

3 This specification, combined with the *TCG Generic Server Specification*, the *TCG ACPI*
4 *Specification*, the *TCG EFI Protocol Specification*, and the *TCG EFI Platform Specification* define
5 requirements for a TCG-compliant Itanium Architecture based server system.

6 This specification outlines fundamental requirements that will allow Itanium Architecture based
7 servers to achieve the high level of interoperability required of a TCG compliant system.

8 The requirements listed in this document allow platform designers the freedom to choose from a
9 wide variety of implementations.

10 ***End of informative comment.***

1 **1.1 Key words**

2 The key words "MUST," "MUST NOT," "REQUIRED," "SHALL," "SHALL NOT," "SHOULD,"
3 "SHOULD NOT," "RECOMMENDED," "MAY," and "OPTIONAL" in normative statements are to
4 be interpreted as described in [RFC-2119].

5

1 **1.2 Statement Type**

2 There are two distinctive kinds of text in this document: informative comments and normative
3 statements. Because much of the text in this specification is normative, the authors have
4 informally defined it as the default and, as such, have specifically called out informative
5 comments with special formatting and notices.

6 The beginning and end of each informative comment is highlighted in grey. If text is not set
7 against a grey background, it should be considered a normative statement.

8 For example:

9 ***Start of informative comment:***

10 This is the first paragraph of 1–n paragraphs containing text of the kind informative comment ...

11 This is the second paragraph of text of the kind informative comment ...

12 This is the nth paragraph of text of the kind informative comment ...

13 To understand the TPM specification the user must read the specification. (This use of MUST
14 does not require any action).

15 ***End of informative comment.***

16 This is the first paragraph of one or more paragraphs (and/or sections) containing normative
17 statements ...

18 To understand the TPM specification the user MUST read the specification. (This use of MUST
19 indicates a keyword usage and requires an action).

1 **2 Context**

2 ***Start of informative comment:***

3 Refer to the *TCG Generic Server Specification* for the context of Trusted Computing in the server
4 space.

5 End of informative comment.

1 **3 Boot Flow**

2 ***Start of informative comment:***

3 This section specifies the required behavior for implementing the static model for establishing a
4 root of trust.

5 ***End of informative comment.***

6 **3.1 The Core Root of Trust for Measurement (CRTM)**

7 ***Start of informative comment:***

8 A platform implementing the static root of trust must begin execution in a known secure state to
9 be trusted. Ensuring that the first instruction fetch is from code known to the manufacturer that
10 cannot be replaced or spoofed by rogue code is one way to accomplish this trusted state.

11
12 Itanium Architecture based servers fetch the first instruction from PAL code. This implementation
13 ensures trusted execution.

14
15 In a Protected Boot Block (PBB) system, a portion of the firmware part is protected from update
16 by hardware. Other sections of the firmware part may be updated on a section-by-section basis.
17 The firmware in the PBB consists of **Generic PAL_A** and **SAL_A**. All other sections lie outside of
18 the PBB.

19
20 All code in the PBB is contained in the CRTM for platforms that use the protected boot block
21 firmware update model, since the code that lies in this region is known to come from the
22 manufacturer and the machine is in a known state when this code executes.

23
24 All other code, including firmware outside of the PBB must be measured and extended to the
25 TPM before it is allowed to execute. The only exception to this process is if the platform
26 manufacturer uses a secure firmware update mechanism that allows only known code to be
27 added to the firmware part. Firmware outside the PBB may also be contained within the CRTM if
28 that firmware is updated only by a secure update method.

29
30 Platforms without a Protected Boot Block system are referred to as non-PBB platforms, and use a
31 different method for firmware update in order to provide a CRTM. Non-PBB platforms update an
32 entire firmware image at once and thus require a secure firmware update method.

33
34 A secure firmware update method allows only known-good firmware images to be written to the
35 firmware part and executed. More detailed requirements for secure firmware update will be
36 described in a following section. In addition, since the entire firmware image is known to come
37 from the manufacturer, the CRTM consists of all of the code in the firmware part.

38 ***End of informative comment.***

39
40 An Itanium-based server design that implements the static root of trust model **MUST** implement
41 one of the following types of CRTM models:

- 42 • Protected Boot Block (PBB model) – the CRTM is protected by hardware and/or PBB
43 firmware from update
- 44 • Non-PBB (non-PBB model) – the CRTM may be updated, but is protected by platform
45 specific mechanisms. Only manufacturer code can update the CRTM.

1

2 3.2 Firmware Structure and Boot Sequence

3 ***Start of informative comment:***

4 This section describes the firmware within the CRTM and external firmware for both PBB and
5 non-PBB CRTM models. This section also explains which firmware must be measured during the
6 boot process based on the Itanium Architecture firmware specifications.

7

8 For a more complete description of the firmware structure and layout see the PAL chapter of the
9 *Intel Itanium Architecture Software Developers Manual, Volume 2, Rev 2.2.*

10

11 For more information about SAL, see the *Itanium Processor Family System Abstraction Layer*
12 *Specification, Ver. 3.2, Dec 2003* including updates for Ver. 3.2 dated June 2004 and August
13 2005.

14

15 The requirements in this document apply only until EFI launch. After EFI launch, an ITANIUM
16 ARCHITECTURE Server must meet the requirements in the *TCG EFI Protocol Specification,*
17 *Version 1.20,* and the *TCG EFI Platform Specification, Version 1.2.*

18

19 The different types of firmware in an Itanium Architecture based system come from various
20 providers.

21

- 22 • The PAL firmware is processor dependent and is provided by the processor manufacturer.
- 23 • The SAL firmware is platform dependent, and is provided by the platform manufacturer.
- 24 • EFI firmware may come from the platform manufacturer or the I/O card manufacturer of any
25 I/O cards included in the platform and is provided by said manufacturer.

26

27 Additionally, some of the EFI firmware may be in the firmware part, in a ROM device on an I/O
28 card, or on a special firmware disk partition. (The Operating System loader also resides on this
29 disk partition.) The manufacturer is responsible for the creating and loading of the firmware part,
30 and for creating and populating the EFI disk partition.

31

32 Each section of code within the firmware part could have been written by any one of several
33 providers who do not know about or share each other's source code. An independent mechanism
34 must be provided to allow jumps from one code section to another.

35

36 The Firmware Interface Table (FIT), allows these jumps between sections of firmware code, and
37 contains the entry point for each logical firmware code segment. The FIT is outside the PBB,
38 except for a FIT entry for **Processor-Specific PAL_A** (see the descriptions of the various logical
39 firmware pieces below).

40

41 The placement of the FIT outside the PBB can create a vulnerability to a denial of service attack if
42 the firmware is updated with untrusted firmware. Keeping multiple copies of each firmware region
43 in the FIT and allowing updates to only one copy at a time can mitigate this attack. Since the
44 CRTM for the non-PBB firmware update model is the entire firmware part, this vulnerability does
45 not exist for the non-PBB firmware update model.

46

47 The following list of firmware modules is sorted by order of execution during the boot process.
48 Each portion of the firmware will be "walked through" from the first instruction fetch until the OS
49 handoff, with a detailed explanation of the actions required within each portion to maintain a
50 trusted computing environment.

- 1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
- **Generic PAL_A:** This is the initial code that is executed when the processor comes out of reset. For the PBB model, this code is not updatable, except in a Manufacturer Approved Environment (MAE). Since it is in the CRTM for both the PBB firmware update model and the non-PBB firmware update model, this code does not need to be measured.
 - **Processor-Specific PAL_A:** This code is executed after **Generic PAL_A** runs. Since this code can change from one processor generation to another and can be upgraded for processor bug fixes, this code is not in the PBB. This code must be cryptographically protected in the PBB implementation. A digital signature controlled by the processor manufacturer provides this function. The signature provides sufficient protection that this code segment does not need to be measured. **Processor-Specific PAL_A** also authenticates **PAL_B**. Code from **PAL_B** may be executed at this time.
 - **SAL_A:** This code is called from **Processor-Specific PAL_A**. It is provided by the platform manufacturer, and is contained within the PBB. **SAL_A** may be updated in a Manufacturer Approved Environment (MAE) only. **PAL_A** calls **SAL_A** to check for a firmware recovery update request and to perform the update if required. The platform must be reset if **SAL_A** updates the platform firmware. If an update is not required, **SAL_A** returns directly to **PAL_A**. A measurement of this code segment is not required.
 - **PAL_B:** **SAL_A** then returns to **Processor-Specific PAL_A**, and **Processor-Specific PAL_A** then jumps to **PAL_B**, which contains the boot and runtime procedures used by **SAL**, **EFI**, and the OS. **PAL_B** is not in the PBB. **PAL_B** must be cryptographically protected in a PBB implementation. Authenticating in an implementation-dependent manner (based on the processor manufacturer) will cryptographically protect **PAL_B**. This code does not need to be measured.
 - **SAL_B:** After **PAL_B** returns to **SAL_A**, **SAL_A** jumps into **SAL_B** in an implementation-dependent manner. **SAL_B** is not in the PBB, and must either be cryptographically authenticated or measured into **PCR[0]** before it is executed when using the PBB update model. Measurement and authentication are not necessary in the non-PBB update model, as **SAL_B** is in the CRTM. **SAL_B** code configures the platform, then transfers control to **EFI** for firmware console interaction or to load the OS.
 - **EFI:** This code is called from within **SAL_B**. Code measurement requirements for **EFI** and any code loaded within **EFI** are contained in the TCG **EFI Platform Specification**, Ver. 1.2.
- End of informative comment.**

1 **3.2.1 Requirements for Itanium Architecture Based Servers**
2 **which Implement PBB**

3 An Itanium Architecture based server that implements the static root of trust and the PBB models
4 MUST do the following.

- 5 • **Note:** All code that is measured MUST be extended into the relevant PCR before it is
6 executed.
- 7 • Initialize the TPM and extend **PCR[0]** with any of these values:
- 8 o 0
- 9 o The digest of the PBB
- 10 o The revision of the PBB
- 11 • If any code outside the PBB is contained within the CRTM, its revision must be extended
12 also.
- 13 • Extend **PCR[0]** with a digest of any code outside the PBB that can be updated by code
14 not contained within the CRTM. If code must be extended to **PCR[0]**, the FIT table must
15 also be extended.
- 16 • Firmware MAY extend **PCR[1]** with platform configuration data consumed by the
17 firmware. This includes code run before EFI is launched or which is passed to EFI or the
18 OS.
19 Firmware MUST make this measurement if the data is not part of the code in the CRTM
20 or previously measured into **PCR[0]**.
- 21 • Retain the digests and event codes for all measurements made into **PCR[0]** or **PCR[1]**
22 until memory is available. Write a log of the measurements into memory. This area of
23 memory MUST be made available for code that runs after the log is written.
- 24 • All Itanium Architecture based systems implementing the static root of trust model MUST
25 also adhere to the *TCG EFI Protocol Specification, Version 1.2* and *TCG EFI Platform*
26 *Specification, Version 1.2*.

1 **3.2.2 Requirements for Itanium Architecture Based Servers**
2 **which do not Implement PBB**

3 An Itanium Architecture based server which implements the static root of trust and non-PBB
4 models MUST do the following:

- 5 • Initialize the TPM before executing any code outside the CRTM.
- 6 • During CRTM execution, code MAY extend **PCR[0]** with:
 - 7 ○ 0
 - 8 ○ The digest of the CRTM.
 - 9 ○ The revision of the CRTM.
 - 10 ○ The revisions of the modules that make up the CRTM.
- 11 • Firmware MAY extend **PCR[1]** with platform configuration data consumed by the
12 firmware. This includes code run before EFI is launched or which is passed to EFI or the
13 OS.
14 Firmware MUST make this measurement if the data is not part of the code in the CRTM
15 or previously measured into **PCR[0]**.
- 16 • Retain the digests and event codes for all measurements made into **PCR[0]** or **PCR[1]**
17 until memory is available. Write a log of the measurements into memory. This area of
18 memory MUST be made available for code that runs after the log is written.
- 19 • All Itanium Architecture based systems implementing the static root of trust model MUST
20 also adhere to the *TCG EFI Protocol Specification, Version 1.20* and *TCG EFI Platform*
21 *Specification, Version 1.2*.
22

1 **4 Requirements**

2 ***Start of informative comment:***

3 Refer to the *TCG Generic Server Specification* for the requirements of a server that incorporates
4 TCG technology.

5 ***End of informative comment.***

6

1 5 PCR Usage and Event Logging

2 **Start of informative comment:**

3 The *TCG Generic Server Specification* specifies the overall usage of the PCRs for a server that
4 incorporates TCG technology.

5 Detailed logging requirements are explained in this document and the *TCG EFI Platform*
6 *Specification*. These lower level specifications will not contradict the *TCG Generic Server*
7 *Specification*, but will simply add needed details specific to Itanium Architecture based servers.

8 This specification does not enumerate a required number of PCRs. This minimum usage
9 described is compatible with TPM Family: 1.1b; Level: 1; Revision 0 and later.

10 **End of informative comment.**

11 PCR Usage and Event Logging on Itanium Architecture based servers implementing the static
12 root of trust model MUST comply with *the TCG EFI Protocol Specification*, *TCG EFI Platform*
13 *Specification*, the *TCG Generic Server Specification*, and the requirements listed in this section.

1 **5.1 Locality**

2 ***Start of informative comment:***

3 This specification makes no assumptions regarding locality. Statements regarding locality in the
4 *TCG Generic Server Specification* do apply.

5 ***End of informative comment.***

1 **5.2 Pre-Boot PCR Usage**

2 **5.2.1 PCR[0] – CRTM, System Firmware, and System Extensions**

3 ***Start of informative comment:***

4 Itanium Architecture based servers may encounter situations in which there may be multiple
5 version IDs to store into **PCR[0]**.

6 PAL, SAL, and other Itanium Architecture firmware modules can be measured and logged
7 independently, for example.

8 ***End of informative comment.***

9 Any of the following Itanium Architecture firmware modules that are not part of the CRTM MUST
10 be measured and extended into **PCR[0]**:

- 11 • PMI code
- 12 • ACPI code
- 13 • SAL Code
- 14 • PAL Code
- 15 • EFI Code (Refer to the *TCG EFI Platform Specification* for more details.)

16 **5.2.2 PCR[1] – System Configuration**

17 The following system firmware data structures may optionally be measured and extended into
18 **PCR[1]** on an Itanium Architecture based server:

- 19 • ACPI tables
- 20 • SMBIOS tables
- 21 • SAL System Table

22 **5.2.3 PCR[2] – Firmware Driver Code**

23 Please refer to the *EFI TCG Platform Specification* for more details on this topic.

24 **5.2.4 PCR[3] – Firmware Driver Configuration and Data**

25 Please refer to the *EFI TCG Platform Specification* for more details on this topic.

26 **5.2.5 PCR[4] – Initial Program Load (IPL) Code**

27 Please refer to the *EFI TCG Platform Specification* for more details on this topic.

28 **5.2.6 PCR[5] – IPL Configuration and Data**

29 Please refer to the *EFI TCG Platform Specification* for more details on this topic.

1 **5.2.7 PCR[6] – State Transition**

2 ***Start of informative comment:***

3 Itanium Architecture based servers generally do not perform state transitions (such as sleep
4 states) before boot time.

5 ***End of informative comment.***

6

7 If an Itanium server performs a pre-boot state transition affecting the security state of the server,
8 the server MUST be reset.

9 **5.2.8 PCR[7] – Reserved**

10 PCR[7] is reserved. Refer to the *TCG Generic Server Specification* for more details on this topic.

11 **5.2.9 PCR[8] – PCR[15] Reserved**

12 PCR 8-15 are reserved for the original equipment manufacturer's (OEM) use in cases where the
13 TPM provides 24 PCRs or more. Refer to the *TCG Generic Server Specification* for more details
14 on this topic.

15 **5.2.10 Unusual Pre-boot Actions**

16 OEMs MUST log platform specific pre-boot actions that affect the security state of the platform in
17 a PCR within the OEM range of PCRs.

18 **5.2.11 Runtime Actions and PCR Usage**

19 Refer to the *TCG Generic Server Specification* for more information on this topic.

1 **5.3 Event Logging**

2 **5.3.1 Event Types**

3 The following events are defined for the field **TCG_ServerPCREventStruct.eventType**.

4

| Label | Value | Description |
|--------------------------|-------|---|
| EV_PREBOOT_CERT | 00h | The event field contains certificates such as the Validation Certificates. |
| EV_PROGRESS_CODE | 01h | The digest field contains the SHA-1 hash of the system firmware progress code. The event field SHOULD NOT contain the actual progress code but MAY contain information about the progress code. |
| EV_UNUSED | 02h | Reserved. |
| EV_NO_ACTION | 03h | The event field contains informative data that was not extended into any PCR. The fields: pcrIndex and digest MUST contain the value 0. |
| EV_SEPARATOR | 04h | Delimits actions taken during the pre-OS and OS environments. |
| EV_ACTION | 05h | A specific action measured as a string defined in the TCG EFI Platform Specification. |
| EV_EVENT_TAG | 06h | The event field contains the structure defined in Section 5.3.2. |
| EV_S_CRTM_CONTENTS | 07h | This digest field contains is the SHA-1 hash of the S-CRTM. The event field SHOULD NOT contain the actual S-CRTM code but MAY contain informative information about the S-CRTM code. |
| EV_S_CRTM_VERSION | 08h | The event field contains the version string of the S-CRTM |
| EV_UNUSED_2 | 09h | Reserved. |
| EV_PLATFORM_CONFIG_FLAGS | 0Ah | The format and contents to be defined by the platform manufacturer. Examples of information contained in this event type is the capabilities of the platform's measurements, whether the owner has disabled measurements, etc. |
| EV_TABLE_OF_DEVICES | 0Bh | The event field contains the Platform manufacturer-provided Table of Devices or other Platform manufacturer-defined information. The Platform Manufacturer defines the content and format of the Table of Devices. The Platform Credential may provide a reference to the meaning of these structures and data. This structure is measured into PCR[1] . |

1 5.3.2 Platform Specific Tagged Event Log Structure

2
3 The *TCG Generic Server Specification* defines the following structure, which is listed for
4 reference below.
5

```
6     typedef struct TCG_ServerPlatformSpecificEventLogStruct{  
7         UINT32 EventID; // Tag from arch. Specific Spec.  
8         UINT32 EventDataSize; // Size of EventData  
9         UINT8 EventData[]; // EventData  
10    } TCG_ServerPlatformSpecificEventLogStruct;
```

11 Example 1. Example Event Log Structure

12

13 5.3.2.1 Server Platform Specific Event Tags

14 The following sections list the Itanium Architecture based server platform event tags used in the
15 Tagged Event Log Structure.

- 16
- 17 • References to **EventID** are to **TCG_ServerPlatformSpecificEventLogStruct.EventID**.
 - 18 • References to **EventData** are to
19 **TCG_ServerPlatformSpecificEventLogStruct.EventData**.

20 These Itanium Architecture-specific event tags will be in the range of 0x70000000 0x7FFFFFFF.
21 EFI events are described in the *TCG EFI Platform Specification*.

22 5.3.2.1.1 SMBIOS structure

23 Each event MAY consist of one or more complete SMBIOS records. This event may appear
24 multiple times in the event log. The SMBIOS structure SHALL be logged using the following:

25 EventID = 0x70000000
26 EventData[] = One or more raw complete SMBIOS records.

27 5.3.2.1.2 BIS Certificate

28 The BIS Certificate MUST be logged using the following:

29 EventID = 0x70000001
30 EventData[] = Raw BIS Certificate

31 5.3.2.1.3 System Firmware ROM Strings

32 The System Firmware ROM Forward Progress Strings SHALL be logged using the following:

33 EventID = 0x70000002
34 EventData[] = Hash of System Firmware ROM Strings

1 **5.3.2.1.4 NVRAM**

2 The NVRAM SHALL be logged using the following:

3 EventID = 0x7000003

4 EventData[] = Raw NVRAM contents

5 **5.3.3 EV_ACTION Event Types**

6 The Itanium Architecture based Server specification does not define a set of strings that must be
7 used for event logging.

8 Refer to the *TCG EFI Platform Specification* for events that are specific to EFI implementations.



1 **6 TPM**

- 2 An Itanium Architecture based server MUST incorporate a TPM compatible with TPM Family:
- 3 1.1B; Level: 1; Revision: 0 or later.



1 **7 TBB**

- 2 Refer to the *TCG Generic Server Specification* for more information about this topic.

1 **8 Glossary**

2

| | |
|------------|---|
| EFI | Extensible Firmware Interface. An interface between the operating system (OS) and the platform firmware. The interface is in the form of data tables that contain platform-related information, and boot and runtime service calls that are available to the OS and its loader. Together, these provide a standard environment for booting an OS. |
| FIT | Firmware Interface Table. Data structure that allows separate firmware modules to communicate entry points. |
| PAL | Processor Abstraction Layer. Processor specific code that abstracts Itanium processor features that are implementation dependent from the operating system and system firmware. |
| PBB | Protected Boot Block. Critical components of system firmware are kept in part of a FLASH chip that is not writable during normal firmware update operations. |
| PMI | Processor Management Interrupt. A special mode of processor execution that allows system firmware to take control of the processor when certain hardware events occur. |
| SAL | System Abstraction Layer. The firmware layer which abstracts system features that are implementation dependent. |

3

1 **9 Bibliography**

2 When used within this document the following document names have the specific meaning shown
3 below:

- 4 • TCG Generic Server Specification: TCG Generic Server Specification, Version 1.0
- 5 • TCG ACPI Specification: TCG ACPI Specification, Version 1.0
- 6 • TCG EFI Protocol Specification: TCG EFI Protocol Specification, Version 1.2
- 7 • TCG EFI Platform Specification: TCG EFI Platform Specification, Version 1.2