InsydeH2O® Dynamic PCD Extension for SMM and Runtime

March 13, 2014
Revision 1.0
Confidential

Disclaimer

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

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

The following trademarks are used in this document:

Insyde is a registered trademark and Peony is a trademark of Insyde Software Corp.
All other trademarks or trade names are property of their respective holders.

Contents

Table of Content

Revision history

Revision Number	Description	Revision Date
1	Revision 1.0	March 13, 2014

1. Introduction

This document is a part of the InsydeH2O® 5.0 Technical Reference, which describes the architectural features of the product. These architectural features serve as an agreement between Insyde and its partners and customers about the types of APIs, data structures, processes and file formats that the InsydeH2O 5.0 product will use and produce.

This document focuses on the InsydeH2O 5.0 PCD extensions for SMM and UEFI run-time phase.

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

- *Unified Extensible Firmware Interface Specification*, Version 2.3.1c, Unified EFI, Inc, 2012, http://www.uefi.org.
- *EDK II Build Specification*, Revision 1.22 Errata B, June 2012, Copyright © 2007 – 2011 Intel Corporation.
- *EDK II Package Declaration (DEC) File Format Specification*, Revision 1.22 Errata B, June 2012, Copyright © 2007 – 2011 Intel Corporation.
- *EDK II Platform Description (DSC) File Specification,* Revision 1.22 Errata B, June 2012, Copyright © 2007 – 2011 Intel Corporation.
- *EDK II Flash Description (FDF) File Specification*, Revision 1.22 Errata B, June 2011, Copyright © 2007 – 2011 Intel Corporation.
- *EDK II Module Information (INF) File Specification,* Revision 1.22 Errata B, June 2012, Copyright © 2007 – 2011 Intel Corporation.
- *VFR Programming Language*, Revision 1.7, May 2012, Copyright © 2007 – 2012 Intel Corporation.
- *InsydeH2O 5.0 Coding Standard*, Version 1.01, Copyright © 2012 Insyde Software Corp.

1 Overview

The PCD (Platform Configuration Database) support introduced by EDKII provides UEFI firmware developers a universal and convenient way to setup system configuration through a central controlled configuration database. PCDs can also be set to dynamic type PCDs which can be modified during run-time, these PCDs are called dynamic PCDs.

However, dynamic PCDs are stored in BootServices type memory by the PCD DXE/PEIM modules, which means we cannot access dynamic PCDs at UEFI Runtime or in SMM at OS run-time, otherwise it might cause system hang-up due to invalid memory access because BootServices memory has been freed at OS run-time.

The dynamic PCD extension for SMM and Runtime allows that the dynamic PCDs (either with dynamic or dynamicEx type) can be accessed even if the system is at OS run-time, and the PCD usage is almost the same as the normal PCD usage during POST at boot time with some exceptions that some PCD Protocol/Library functions are unsupported at Runtime or in SMM mode. The potential risks of system break caused by accidentally accessing dynamic/dynamicEx PCDs at OS Runtime or in SMM can be eliminated.

2 Related Modules

2.1 PcdSmmDxe Module

Module Path

InsydeModulePkg/Universal/PcdSmmDxe/PcdSmmDxe.inf

Description

The PcdSmmDxe module is a COMBINED_SMM_DXE module which can be dispatched both by DXE dispatcher and SMM dispatcher. The module instance dispatched by DXE dispatcher will be relocated to Runtime memory space as a DXE_RUNTIME_DRIVER module to allow it to be persisted after ExitBootServices event is triggered.

This module is to replace the standard EDKII's DXE PCD module located at

MdeModulePkg/Universal/PCD/DXE/Pcd.inf

which can only accessing dynamic/dynamicEx type's PCDs during POST process.

2.2 SmmDxePcdLib Module

Module Path

InsydeModulePkg/Library/SmmDxePcdLib/SmmDxeLib.inf

Description

The SmmDxePcdLib module is the extended Library instance for PcdLib Library Class for drivers with DXE_RUNTIME_DRIVER, DXE_SMM_DRIVER and COMBINED_SMM_DXE module types. For drivers with other module types, standard PcdLib Library instances are used.

3 PCD Protocol/Library Function Usages

The following sections describes the support matrices of the PCD Protocol/LibraryClass functions for Runtime and SMM modes with the PCD extension feature added, in general, PCD set functions are not supported after booting to the operating system, either at Runtime or in SMM mode.

3.1 EFI_PCD_PROTOCOL Functions

The following table is the support matrix for EFI_PCD_PROTOCOL functions at Runtime and SMM phases

Function Name	Supported at Runtime (after ExitBootServices)	Supported In SMM mode before SmmReadyToBoot Event	Supported In SMM mode after SmmReadyToBoot Event
SetSku()	N	**Y**	N
GetXxxx()	**Y**	**Y**	**Y**
GetSize()	**Y**	**Y**	**Y**
CallbackOnSet()	N	N	N
CancelCallback()	N	N	N
GetNextToken()	**Y**	**Y**	**Y**
GetNextTokenSpace()	**Y**	**Y**	**Y**

3.2 PCD_PROTOCOL Functions

The following table is the support matrix for PCD_PROTOCOL functions at Runtime and SMM phases

Function Name	Supported at Runtime (after ExitBootServices)	Supported In SMM mode before SmmReadyToBoot Event	Supported In SMM mode after SmmReadyToBoot Event
SetSku()	N	**Y**	N
GetXxxx()	**Y**	**Y**	**Y**
GetSize()	**Y**	**Y**	**Y**
GetXxxxEx()	**Y**	**Y**	**Y**
GetSizeEx()	**Y**	**Y**	**Y**
SetXxxxEx()	N	**Y**	N
CallbackOnSet()	N	N	N
CancelCallback()	N	N	N
GetNextToken()	**Y**	**Y**	**Y**
GetNextTokenSpace()	**Y**	**Y**	**Y**

3.3 PcdLib Functions

The following table is the support matrix for PcdLib LibraryClass functions at Runtime and SMM phases

Function Name	Supported at Runtime (after ExitBootServices)	Supported In SMM mode before SmmReadyToBoot Event	Supported In SMM mode after SmmReadyToBoot Event
PcdToken()	**Y**	**Y**	**Y**
FeaturePcdGet()	**Y**	**Y**	**Y**
FixedPcdGetXxxx()	**Y**	**Y**	**Y**
PatchPcdGetXxxx()	**Y**	**Y**	**Y**
PcdGetXxxx()	**Y**	**Y**	**Y**

PcdSetXxxx()	N	**Y**	N
PcdGetExXxxx()	**Y**	**Y**	**Y**
PcdSetExXxxx()	N	**Y**	N
LibPcdSetSku()	N	**Y**	N
LibPcdGetXxxx()	**Y**	**Y**	**Y**
LibPcdGetExXxxx()	**Y**	**Y**	**Y**
LibPcdSetXxxx()	N	**Y**	N
LibPcdSetExXxxx()	N	**Y**	N
LibPcdCallbackOnSet()	N	N	N
LibCancelCallback()	N	N	N
LibGetNextToken()	**Y**	**Y**	**Y**
LibPcdGetNextTokenSpace()	**Y**	**Y**	**Y**
LibPatchPcdSetPtr()	**Y**	**Y**	**Y**
LibPcdGetInfo()	N	**Y**	N
LibPcdGetInfoEx()	N	**Y**	N
LibPcdGetSku()	**Y**	**Y**	**Y**

4 Design Considerations

The following items are the list of limitations and recommendations for the usage of PCD related functions

1. PCD set functions such as PcdSetXxxx(), PcdSetExXxxx() (Xxxx=8/16/32/64/Ptr/Bool/Sku), LibPcdCallbackOnSet(), etc. are not supported during OS run-time. Developers should avoid this usage.

2. The PCD database will be duplicated to SMRAM after SmmReadyToLock event triggered, PcdSetXxxx() called by DXE drivers after SmmReadyToLock event will not be synchronized to SMRAM, care must be taken if PCD set functions are called after SmmReadyToLock event.

3. If possible, avoid using PCD Protocol functions or LibPcdGetXxxx() and LibPcdSetPcdXxx() functions directly, always try to use PCD macro functions such as PcdGetXxxx() or PcdSetXxxx(), build tool can smartly translate these macro functions to get/set the correct PCD values from the auto-generated AutoGen.h at the module build output folder.