Platform Configuration Database –Documentation on the Library

PEI

PCD PEIM produces PCD database to manage all dynamic PCD in PEI phase and install Pcd Ppi service.

There are two PCD PPIs as follows:

1) PCD\_PPI

It is EDKII implementation which support Dynamic/DynamicEx Pcds.

2) EFI\_PEI\_PCD\_PPI

It is defined by PI specification 1.2, Vol 3 which only support dynamicEx

type Pcd.

For dynamicEx type PCD, it is compatible between PCD\_PPI and EFI\_PEI\_PCD\_PPI.

DXE

PCD DXE driver manage database contains all dynamic PCD entries initialized in

PEI phase, DXE phase and produce the implementation of PCD protocol.

There are two PCD Protocols as follows:

1) PCD\_PROTOCOL

It is EDKII implementation which support Dynamic/DynamicEx type Pcds.

2) EFI\_PCD\_PROTOCOL\_PPI

It is defined by PI specification 1.2, Vol 3 which only support dynamicEx

type Pcd.

For dynamicEx type PCD, it is compatible between PCD\_PPI and EFI\_PEI\_PCD\_PPI.

PCD DXE driver will produce above two protocols at same time.

PCD database structure is generated at autogen.h/autogen.c in build time.

Introduction of PCD database

1, Introduction

PCD database hold all dynamic type PCD information. The structure of PEI PCD database is generated by build tools according to dynamic PCD usage for specified platform.

2, Dynamic Type PCD

Dynamic type PCD is used for the configuration/setting which value is determined dynamic. In contrast, the value of static type PCD (FeatureFlag, FixedPcd, PatchablePcd) is fixed in final generated FD image in build time.

2.1 The "dynamic" determination means one of below cases:

a) The PCD setting value is produced by someone driver and consumed by other driver in execution time.

b) The PCD setting value is set/get by user from FrontPage.

c) The PCD setting value is produced by platform OEM vendor in specified area.

2.2 According to module distribution way, dynamic PCD could be classfied as:

a) Dynamic:

If module is released in source code and will be built with platform DSC, the dynamic PCD used by this module can be accessed as:

PcdGetxx(PcdSampleDynamicPcd);

In building platform, build tools will translate PcdSampleDynamicPcd to pair of

{Token Space Guid: Token Number} for this PCD.

b) DynamicEx:

If module is release as binary and will not participate platform building, the dynamic PCD used by this module need be accessed as:

PcdGetxxEx(gEfiMyTokenspaceGuid, PcdSampleDynamicPcd)

Developer needs to explicitly give {Token Space Guid:Token Number} as parameter in writing source code.

2.3 According to PCD value's storage method, dynamic PCD could be classified as:

a) Default Storage:

- The PCD value is stored in PCD database maintained by PCD driver in boot time memory. This type is used for communication between PEIM/DXE driver, DXE/DXE driver. But all set/get value will be lost after boot-time memory is turn off. (e.g. ExitBootServices();)

- [PcdsDynamicDefault] is used as section name for this type PCD in platform DSC file.

- [PcdsDynamicExDefault] is used for dynamicEx type PCD.

b) Variable Storage:

- The PCD value is stored in variable area.

- As default storage type, this type PCD could be used for PEI/DXE driver Communication. But beside it, this type PCD could also be used to store the value associate with a HII setting via variable interface.

- In PEI phase, the PCD value could only be got but cannot be set due to variable area is read only.

- [PcdsDynamicHii] is used as section name for this type PCD in platform DSC file.

- [PcdsDynamicExHii] is for dynamicEx type PCD.

c) OEM specified storage area:

- The PCD value is stored in OEM specified area which base address is specified by a FixedAtBuild PCD setting

- PcdVpdBaseAddress.

- The area is read only for PEI and DXE phase.

- [PcdsDynamicVpd] is used as section name for this type PCD in platform DSC file.

- [PcdsDynamicExVpd] is for dynamicex type PCD.

2.4 When and how to use dynamic PCD

Module developer do not care the used PCD is dynamic or static when writing source code/INF. Dynamic PCD and dynamic type is pointed by platform integrator in platform DSC file. Please ref section 2.3 to get matching between dynamic PCD type and section name in DSC file.

3, PCD database:

Although dynamic PCD could be in different storage type as above description, but the basic information and default value for all dynamic PCD is hold by PCD database maintained by PEI/DXE driver.

As the whole EFI BIOS boot path is divided into PEI/DXE phase, the PCD database also is divided into Pei/Dxe database maintained by PcdPeim/PcdDxe driver separately. To make PcdPeim's driver image smaller, PEI PCD database only hold all dynamic PCD information used in PEI phase or use in both PEI/DXE phase. And DXE PCD database contains all PCDs used in PEI/DXE phase in memory.

Build tool will generate PCD database into some C structure and variable for PEI/DXE PCD driver according to dynamic PCD section in platform DSC file.

3.1 PcdPeim and PcdDxe

PEI PCD database is maintained by PcdPeim driver run from flash. PcdPeim driver build guid hob in temporary memory and copy auto-generated C structur to temporary memory for PEI PCD database. DXE PCD database is maintained by PcdDxe driver.At entry point of PcdDxe driver, new PCD database is allocated in boot-time memory which including all PEI PCD and DXE PCD entry.

Pcd driver should run as early as possible before any other driver access dynamic PCD's value. PEI/DXE "Apriori File" mechanism make it possible by making PcdPeim/PcdDxe as first dispatching driver in PEI/DXE phase.

3.2 Token space Guid/Token number, Platform token, Local token number

3.2.1 Pair of Token space guid + Token number

Any type PCD is identified by pair of "TokenSpaceGuid + TokeNumber". But it is not easy maintained by PCD driver, and hashed token number will make searching slowly.

3.2.2 Platform Token Number

"Platform token number" concept is introduced for mapping to a pair of "TokenSpaceGuid + TokenNumber". The platform token number is generated by build tool in autogen.h and all of them are continual in a platform scope started from 1.(0 meaning invalid internal token number) With auto-generated "platform token number", PcdGet(PcdSampleDynamicPcd) in source code is translated to LibPcdGet(\_PCD\_TOKEN\_PcdSampleDynamicPcd) in autogen.h.

Notes: The mapping between pair of "tokenspace guid + token number" and "internal token number" need build tool establish, so "platform token number" mechanism is not suitable for binary module which use DynamicEx type PCD. To access a dynamicEx type PCD, pair of "token space guid/token number" all need to be specificed for PcdSet/PcdGet accessing macro.

Platform Token Number is started from 1, and inceased continuous. From whole platform scope, there are two zones: PEI Zone and DXE Zone

Platform Token Number

----------|----------------------------------------------------------------

PEI Zone: | 1 ~ PEI\_LOCAL\_TOKEN\_NUMBER

DXE Zone: | (PEI\_LOCAL\_TOKEN\_NUMBER + 1) ~ (PEI\_LOCAL\_TOKEN\_NUMBER + DXE\_LOCAL\_TOKEN\_NUMBER)

3.2.3 Local Token Number

To fast searching a PCD entry in PCD database, PCD driver translate platform token number to local token number via a mapping table.

For binary DynamicEx type PCD, there is a another mapping table to translate token space guid + token number" to local token number directly.Local token number is identifier for all internal interface in PCD PEI/DXE driver.

A local token number is a 32-bit value in following meaning:

32 ------------- 28 ---------- 24 -------- 0

| PCD type mask | Datum Type | Offset |

+-----------------------------------------+

where:

PCd type mask: indicate Pcd type from following macro:

PCD\_TYPE\_DATA

PCD\_TYPE\_HII

PCD\_TYPE\_VPD

PCD\_TYPE\_SKU\_ENABLED

PCD\_TYPE\_STRING

Datum Type : indicate PCD vaue type from following macro:

PCD\_DATUM\_TYPE\_POINTER

PCD\_DATUM\_TYPE\_UINT8

PCD\_DATUM\_TYPE\_UINT16

PCD\_DATUM\_TYPE\_UINT32

PCD\_DATUM\_TYPE\_UINT64

Offset : indicate the related offset of PCD value in PCD database array.

Based on local token number, PCD driver could fast determine PCD type, value type and get PCD entry from PCD database.

3.3 PCD Database C structure.

PCD Database C structure is generated by build tools in PCD driver's autogen.h/autogen.c file. In generated C structure, following information is stored:

- ExMapTable: This table is used translate a binary dynamicex type PCD's "tokenguid + token" to local token number.

- LocalTokenNumberTable: This table stores all local token number in array, use "Internal token number" as array index to get PCD entry's offset fastly.

- SizeTable: This table stores the size information for all PCD entry.

- GuidTable: This table stores guid value for DynamicEx's token space, HII type PCD's variable.

- SkuIdTable: TBD

- SystemSkuId: TBD

- PCD value structure: Every PCD has a value record in PCD database. For different datum type PCD has different record structure which will be introduced in 3.3.1

In a PCD database structure, there are two major area: Init and UnInit.

Init area is use stored above PCD internal structure such as ExMapTable,

LocalTokenNumberTable etc and the (default) value of PCD which has default

value specified in platform DSC file.

Unint area is used stored the value of PCD which has no default value in

platform DSC file, the value of NULL, 0 specified in platform DSC file can

be seemed as "no default value".

3.3.1 Simple Sample PCD Database C Structure

A general sample of PCD database structue is as follows:

typedef struct \_PCD\_DATABASE {

typedef struct \_PCD\_DATABASE\_INIT {

//===== Following is PCD database internal maintain structures

DYNAMICEX\_MAPPING ExMapTable[PEI\_EXMAPPING\_TABLE\_SIZE];

UINT32 LocalTokenNumberTable[PEI\_LOCAL\_TOKEN\_NUMBER\_TABLE\_SIZE];

GUID GuidTable[PEI\_GUID\_TABLE\_SIZE];

SIZE\_INFO SizeTable[PEI\_SIZE\_TABLE\_SIZE];

UINT8 SkuIdTable[PEI\_SKUID\_TABLE\_SIZE];

SKU\_ID SystemSkuId;

//===== Following is value structure for PCD with default

//value

....

....

....

} Init;

typedef struct \_PCD\_DATABSE\_UNINIT {

//==== Following is value structure for PCD without default

//value

....

....

} UnInit;

}

3.3.2 PCD value structure in PCD database C structure

The value's structure is generated by build tool in PCD database C structure.

The PCDs in different datum type has different value structure.

3.3.2.1 UINT8/UINT16/UINT32/UINT64 datum type PCD

The C structure for these datum type PCD is just a UINT8/UINT16/UINT32/UINT64

data member in PCD database, For example:

UINT16 PcdHardwareErrorRecordLevel\_d3705011\_bc19\_4af7\_be16\_f68030378c15\_VariableDefault\_0;

Above structure is generated by build tool, the member name is "PcdCName\_Guidvalue"

Member type is UINT16 according to PcdHardwareErrorRecordLevel declaration

in DEC file.

3.3.2.2 VOID\* datum type PCD

The value of VOID\* datum type PCD is a UINT8/UINT16 array in PCD database.

3.3.2.2.1 VOID\* - string type

If the default value for VOID\* datum type PCD like L"xxx", the PCD is

used for unicode string, and C structure of this datum type PCD is

UINT16 string array in PCD database, for example:

UINT16 StringTable[29];

The number of 29 in above sample is max size of a unicode string.

If the default value for VOID\* datum type PCD like "xxx", the PCD is

used for ascii string, and C structure of this datum type PCD is

UINT8 string array in PCD database, for example:

UINT8 StringTable[20];

The number of 20 in above sample is max size of a ascii string.

3.3.2.2.2 VOID\* - byte array

If the default value of VOID\* datum type PCD like {'0x29', '0x01', '0xf2'}

the PCD is used for byte array. The generated structrue is same as

above ascii string table,

UINT8 StringTable[13];

The number of 13 in above sample is max size of byte array.

3.3.3 Some utility structures in PCD Database

3.3.3.1 GuidTable

GuidTable array is used to store all related GUID value in PCD database:

- Variable GUID for HII type PCD

- Token space GUID for dynamicex type PCD