

UEFI & EDK II Training

Platform Configuration Database (PCD)

tianocore.org

LESSON OBJECTIVE

- ★ Define Platform Configuration Database (PCD) and explain the syntax
- ★ Differentiate types of PCDs
- ★ Explain how changing a PCD value affects output
- ★ Evaluate the results of a PCD value modification
- ★ Special PCDs

PCD Overview

EDK II PCD's Purpose and Goals

 Documentation : [MdeModulePkg/Universal/PCD/Dxe/Pcd.inf](https://github.com/tianocore/edk2/blob/master/MdeModulePkg/Universal/PCD/Dxe/Pcd.inf)

Purpose

- Establishes platform common definitions
- Build-time/Run-time aspects
- Binary Editing Capabilities

Goals

- Simplify porting
- Easy to associate with a module or platform

EDK II PCD's Purpose and Goals

 Documentation : [MdeModulePkg/Universal/PCD/Dxe/Pcd.inf](https://github.com/tianocore/edk2/blob/master/MdeModulePkg/Universal/PCD/Dxe/Pcd.inf)

```
////////////////////////////////////  
//                                                                    //  
//          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.

See Link above to view the entire documentation

PCD Types

FixedAtBuild

Dynamic

PatchableInModule

DyanmicEx

DynamicHii

FeatureFlag

DynamicVpd

Syntax Examples

```
[pcdsFeatureFlag.common] [pcdsFixedAtBuild.IA32]  
[PcdsFixedAtBuild, PcdsPatchableInModule, PcdsDynamic,  
PcdsDynamicEx]
```



UEFI Platform Initialization (PI) 1.x Spec & PCDs

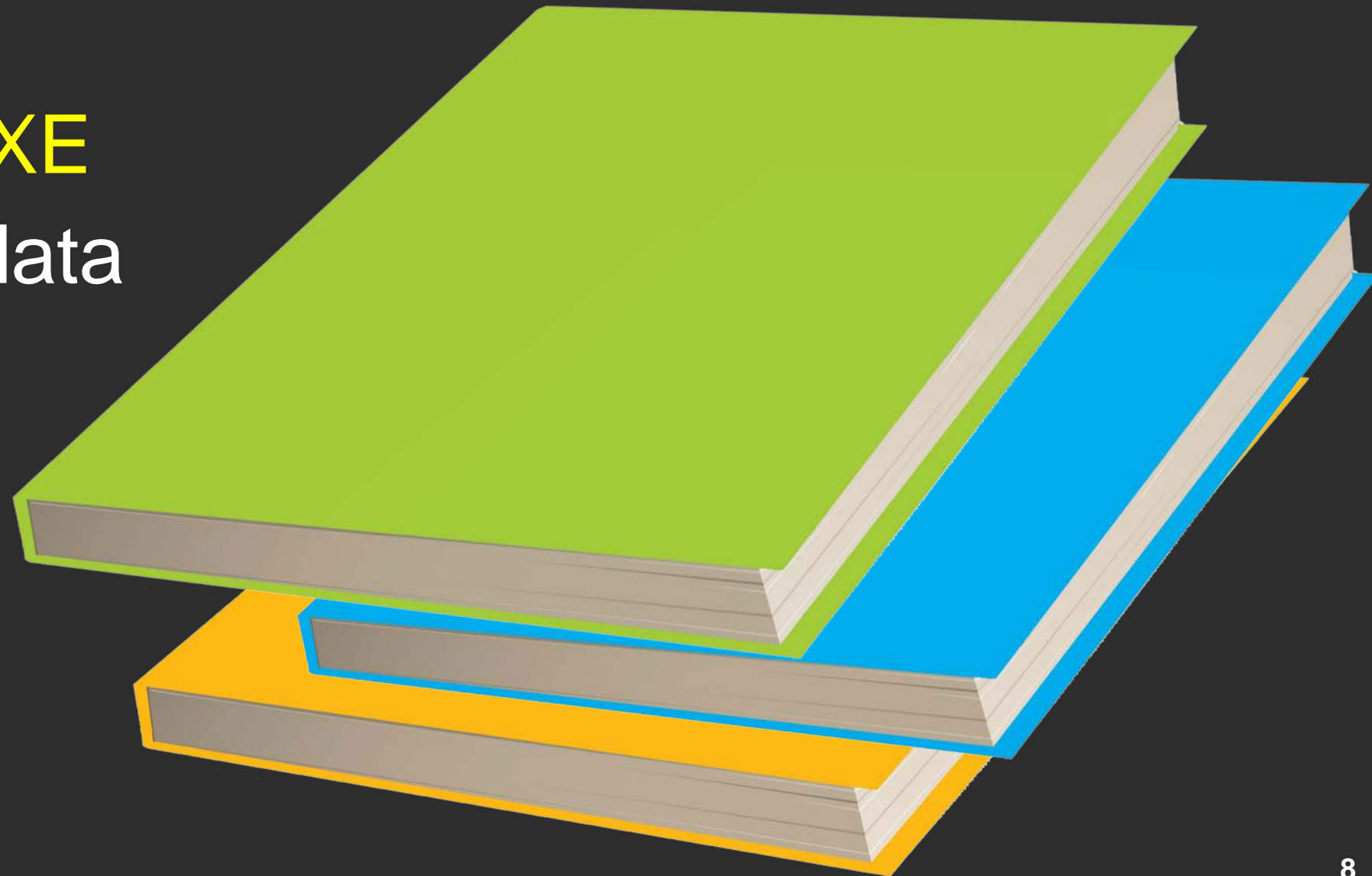
PEI

- PCD PEIM produces PCD database
- Two PCD PPIs: PCD_PPI and EFI_PEI_PCD_PPI

DXE

- DXE Driver Manages PCDs
- Two PCD Protocols: PCD_PROTOCOL and EFI_PCD_PROTOCOL

- Provides interface for PCDs
- PCD PPI - PEI
- PCD Protocol – DXE
- Allows access to data



PCD Library Calls: PCD Protocol and PCD PPI Functions

Example of different Functions:

```
PcdGetXX()  
PcdSetXX()  
PcdGetExXX()  
PcdSetExXX()  
PcdToken()  
PCDSetSku()  
PcdGetNextToken()  
PcdGetNextTokenSpace()  
CallbackOnSet()  
CancelCallback()
```

Where “XX” =

```
8  
16  
32  
Size  
Ptr  
Boolean
```

PCDs can be located anywhere within the Workspace even though a different package will use those PCDs for a given project

.DEC

**Define
PCD**

Package

.INF

**Reference
PCD**

Module

.DSC

**Modify
PCD**

Platform

PCD Syntax example



DEC

PCD defined in the DEC file from any package

```
[Guids.common]
PcdTokenSpaceGuidName={ 0XXXXXXXXX, 0XXXXX, 0XXXXX, { 0XXX, . . .}}

. . .
[Pcds...]
PcdTokenSpaceGuidName.PcdTokenName|Value[|DatumType[|MaxSize]]|Token
```



INF

PCD usage listed in INF file for module

```
[...Pcd...]
PcdTokenSpaceGuidName.PcdTokenName|[Value]
```



DSC

Value of PCD set in Platform DSC

```
[Pcds...]
PcdTokenSpaceGuidName.PcdTokenName|Value[|DatumType[|MaximumDatumSize]]
```

PCD Variable Example

Defined MdeModulePkg/MdeModulePkg.dec



```
[PcdsFixedAtBuild, PcdsPatchableInModule]
gEfiMdeModulePkgTokenSpaceGuid.PcdMaxVariableSize|0x400|UINT32|0x30000003
```

Referenced



```
MdeModulePkg/Universal/Variable/RuntimeDxe/VariableRuntimeDxe.inf
[Pcd]
gEfiMdeModulePkgTokenSpaceGuid.PcdMaxVariableSize ## CONSUMES
```

Modified OvmfPkg/OvmfPkgX64.dsc

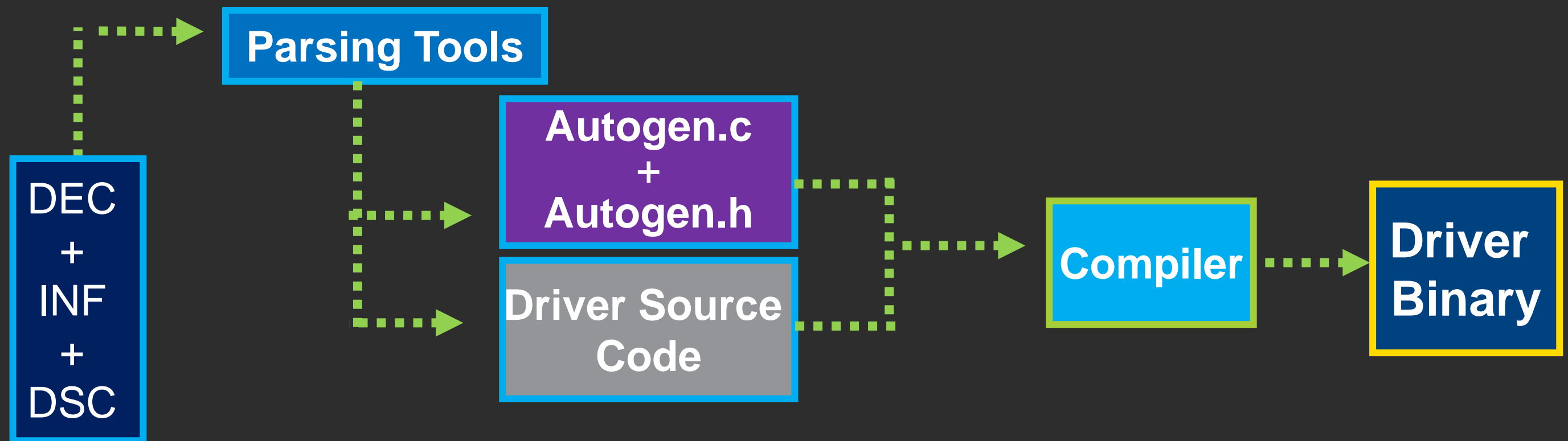


```
[PcdsFixedAtBuild]
gEfiMdeModulePkgTokenSpaceGuid.PcdMaxVariableSize|0x008400
```

Used



```
MdeModulePkg/Universal/Variable/RuntimeDxe/Variable.c // max NV variable size
mVariableModuleGlobal->MaxVariableSize = PcdGet32 (PcdMaxVariableSize);
```



Fixed PCD AutoGen files

Example :  [MdeModulePkg\Universal\Variable\RuntimeDxe\VariableRuntimeDxe](#)

Autogen.h

```
#define _PCD_TOKEN_PcdMaxVariableSize 250U
#define _PCD_SIZE_PcdMaxVariableSize 4
#define _PCD_GET_MODE_SIZE_PcdMaxVariableSize _PCD_SIZE_PcdMaxVariableSize
#define _PCD_VALUE_PcdMaxVariableSize 0x8400U
extern const UINT32 _gPcd_FixedAtBuild_PcdMaxVariableSize;
#define _PCD_GET_MODE_32_PcdMaxVariableSize _gPcd_FixedAtBuild_PcdMaxVariableSize
```

Autogen.c

```
// Definition of PCDs used in this module
...
GLOBAL_REMOVE_IF_UNREFERENCED const UINT32 _gPcd_FixedAtBuild_PcdMaxVariableSize =
    _PCD_VALUE_PcdMaxVariableSize;
```

What about a Dynamic PCDs?

- Only can be Set and changed during Boot time.
- PCD can be set with the library Set: LibPcdSet...
- PCD can be retrieved with the library Get: LibPcdGet...

Example: Use the variable **PcdPlatformBootTimeOut** defined for the platform time in seconds before booting, modified for a value of **03** seconds

Defined

MdeModulePkg/MdeModulePkg.dec



```
[PcdsDynamic]
```

```
gEfiMdePkgTokenSpaceGuid.PcdPlatformBootTimeOut|0xffff|UINT16|0x
```

Modified

OvmfPkg/OvmfPkg.dsc



```
[PcdsDynamicDefault]
```

```
gEfiMdePkgTokenSpaceGuid.PcdPlatformBootTimeOut|03
```

Setting



OvmfPkg/Library/PlatformBootManagerLib/BdsPlatform.c

```
PcdStatus = PcdSet16S (PcdPlatformBootTimeOut,  
    GetFrontPageTimeoutFromQemu ());
```

Used



OvmfPkg/Library/QemuBootOrderLib/QemuBootOrderLib.c

```
Timeout = PcdGet16 (PcdPlatformBootTimeOut);
```


Dynamic PCD Autogen Files

Example Module: (OvmfPkg/Library/PlatformBootManagerLib)

Autogen.h

• • •

```
#define _PCD_SET_MODE_16_PcdPlatformBootTimeOut(Value) \
    LibPcdSet16(_PCD_TOKEN_PcdPlatformBootTimeOut, ( Value ))
#define _PCD_SET_MODE_16_S_PcdPlatformBootTimeOut(Value) \
    LibPcdSet16S(_PCD_TOKEN_PcdPlatformBootTimeOut, ( Value ))
```

Example Module: (MdeModulePkg/Universal/PCD/Dxe/Pcd)

Autogen.c

```
DXE_PCD_DATABASE_INIT gDXEPcdDbInit = {
    • • •
    /* LocalTokenNumberTable */
    • • •
    offsetof(DXE_PCD_DATABASE, Init.PcdPlatformBootTimeOut_*1) | PCD_TYPE_DATA | PCD_DATUM_TYPE_UINT16,
    • • •
    { 0x3U } /* PcdPlatformBootTimeOut_*1 [1] */,
```

*1 GUID of PCD Variable PcdPlatformBootTimeOut

Multi-Structure PCD

- C data structure and assign the value to each sub-field directly

Multi-Sku PCD

- Multiple configurations generated at build time & set @ run time, (PI Spec Vol 3 chap. 8)

DefaultStores PCD

- Support the default stores concept in UEFI specification, (UEFI, HII Chap. 32)

Multiple “C” Data Structure as PCDs

Example: SMBIOS Type 0 Data Structure PCD Defined

`edk2-platforms/Features/Intel/SystemInformation/ SmbiosFeaturePkg.dec`

```
gSmbiosFeaturePkgTokenSpaceGuid.PcdSmbiosType0BiosInformation| \
{0x0}|SMBIOS_TABLE_TYPE0|0xD0000001 {
  <HeaderFiles>
    IndustryStandard/SmBios.h → MdePkg/Include/IndustryStandard/SmBios.h
  <Packages>
    MdePkg/MdePkg.dec
    SystemInformation/SmbiosFeaturePkg/SmbiosFeaturePkg.dec
}
gSmbiosFeaturePkgTokenSpaceGuid.PcdSmbiosType0BiosInformation.Vendor|0x1
gSmbiosFeaturePkgTokenSpaceGuid.PcdSmbiosType0BiosInformation.BiosVersion|0x2
gSmbiosFeaturePkgTokenSpaceGuid.PcdSmbiosType0BiosInformation.BiosSegment|0xF000
gSmbiosFeaturePkgTokenSpaceGuid.PcdSmbiosType0BiosInformation.BiosReleaseDate|0x3
gSmbiosFeaturePkgTokenSpaceGuid.PcdSmbiosType0BiosInformation.BiosSize|0xFF
gSmbiosFeaturePkgTokenSpaceGuid.PcdSmbiosType0BiosInformation.BiosCharacteristics.\
  PciIsSupported|1
gSmbiosFeaturePkgTokenSpaceGuid.PcdSmbiosType0BiosInformation.BiosCharacteristics.\
  PlugAndPlayIsSupported|1
```

• • •

Multiple “C” Data Structure as PCDs

Example: SMBIOS Type 0 Data Structure “C” Data structure in **SmBios.h**

<https://github.com/tianocore/edk2/.../MdePkg/Include/IndustryStandard/SmBios.h>

```
/// BIOS Information (Type 0).
///
typedef struct {
    SMBIOS_STRUCTURE           Hdr;
    SMBIOS_TABLE_STRING       Vendor;
    SMBIOS_TABLE_STRING       BiosVersion;
    UINT16                    BiosSegment;
    SMBIOS_TABLE_STRING       BiosReleaseDate;
    UINT8                     BiosSize;
    MISC_BIOS_CHARACTERISTICS BiosCharacteristics;
    UINT8                     BIOSCharacteristicsExtensionBytes[2];
    UINT8                     SystemBiosMajorRelease;
    UINT8                     SystemBiosMinorRelease;
    UINT8                     EmbeddedControllerFirmwareMajorRelease;
    UINT8                     EmbeddedControllerFirmwareMinorRelease;
    EXTENDED_BIOS_ROM_SIZE    ExtendedBiosSize;
} SMBIOS_TABLE_TYPE0;
```

Names in the “C” data structure match the names in the PCDs

DSC File – SKU Set at BUILD time

```
• • •
SKUID_IDENTIFIER = ?

[SkuIds]
0|DEFAULT
4|BoardX
0x42|BoardY

[PcdsDynamicDefault.common.BoardX]
gBoardModuleTokenSpaceGuid.PcdGpioPin|0x8
gBoardModuleTokenSpaceGuid.PcdGpioInitValue|\
    {0x00, 0x04, 0x02, 0x04, ...}

[PcdsDynamicDefault.common.BoardY]
gBoardModuleTokenSpaceGuid.PcdGpioPin|0x4
gBoardModuleTokenSpaceGuid.PcdGpioInitValue|\
    {0x00, 0x02, 0x01, 0x02, ...}
```

SKU PCD Set Dynamically

```
BoardXBoardDetect( VOID)
{
    • • •
    if (LibPcdGetSku () != 0) {
        return EFI_SUCCESS;
    }
    if (IsBoardX ()) {
        LibPcdSetSku (BoardIdIsBoardX);
        ASSERT (LibPcdGetSku() ==
                BoardIdIsBoardX);
    }
    return EFI_SUCCESS;
}
```

DSC File –

```
•  
•  
•  
VPD_TOOL_GUID = 8C3D856A-9 ...
```

```
[DefaultStores]  
0|STANDARD  
1|MANUFACTURING  
2|SAFE
```

```
[PcdsDynamicExVpd.common.DEFAULT]  
  gEfiMdeModulePkgTokenSpaceGuid.PcdNvStoreDefaultValueBuffer|*  
[PcdsDynamicEx.common.DEFAULT.STANDARD]  
  gOemSkuTokenSpaceGuid.PcdSetupData.CloudProfile|0x0  
  gOemSkuTokenSpaceGuid.PcdSetupData.Use1GPageTable|0x1  
[PcdsDynamicEx.common.DEFAULT.MANUFACTURING]  
  gOemSkuTokenSpaceGuid.PcdSetupData.CloudProfile|0x1  
  gOemSkuTokenSpaceGuid.PcdSetupData.Use1GPageTable|0x0
```

- Special PCD to support the default stores concept in UEFI specification
- Can be Dynamically set

Summary

- Define Platform Configuration Database (PCD) and explain the syntax
- Differentiate types of PCDs
- Explain how changing a PCD value affects output
- Evaluate the results of a PCD value modification
- Special PCDs

Questions?



Return to Main Training Page



Return to Training Table of contents for next presentation [link](#)



ACKNOWLEDGEMENTS

Redistribution and use in source (original document form) and 'compiled' forms (converted to PDF, epub, HTML and other formats) with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code (original document form) must retain the above copyright notice, this list of conditions and the following disclaimer as the first lines of this file unmodified.

Redistributions in compiled form (transformed to other DTDs, converted to PDF, epub, HTML and other formats) must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS DOCUMENTATION IS PROVIDED BY TIANOCORE PROJECT "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL TIANOCORE PROJECT BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS DOCUMENTATION, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 2021, Intel Corporation. All rights reserved.

BACKUP

PCD Dynamic and DynamicEx

PCD DynamicEx (follows PI 1.x Spec)

- Referenced using Token Number and GUID
- Required for modules that are distributed as binaries
- The size is slightly larger compare with Dynamic

PCD Dynamic

- Referenced only by a Token Number without a GUID
- Useful for modules that are build from sources
- Reduce the size overhead of using PCDs

Dynamic PCD is size optimized compared to DynamicEX when modules are build from source