American Megatrends

Aptio Overview
PEI

AMI CHINA

LEGAL

Disclaimer

- This publication contains proprietary information which is protected by copyright. No part of this publication may be reproduced, transcribed, stored in a retrieval system, translated into any language or computer language, or transmitted in any form whatsoever without the prior written consent of the publisher, American Megatrends, Inc. American Megatrends, Inc. retains the right to update, change, modify this publication at any time, without notice.

For Additional Information

- Call American Megatrends, Inc. at 1-800-828-9264 for additional information.

Limitations of Liability

- In no event shall American Megatrends be held liable for any loss, expenses, or damages of any kind whatsoever, whether direct, indirect, incidental, or consequential, arising from the design or use of this product or the support materials provided with the product.

Limited Warranty

- No warranties are made, either express or implied, with regard to the contents of this work, its merchantability, or fitness for a particular use. American Megatrends assumes no responsibility for errors and omissions or for the uses made of the material contained herein or reader decisions based on such use.

Trademark and Copyright Acknowledgments

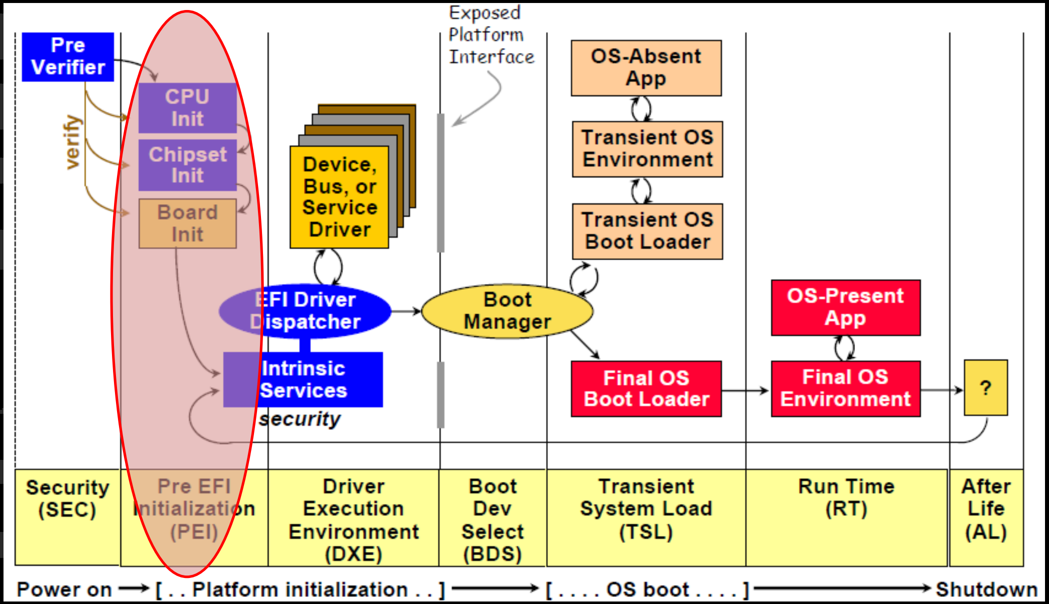
- Copyright ©2013 American Megatrends, Inc. All Rights Reserved.
- American Megatrends, Inc., 5555 Oakbrook Parkway, Suite 200, Norcross, GA 30093
- All product names used in this publication are for identification purposes only and are trademarks of their respective Companies.

Agenda

- PEI phase
  - What is PEI phase
  - PEI phase ---- Do what
  - PEI Elements
  - PEI Flowchart
  - PEI Sample Code
- PEI HOB
  - PEI HOB introduce
  - PEI HOB Sample Code

What is PEI phase

The Pre-EFI Initialization(PEI) phase is invoked quite early in the boot flow.



PEI phase ---- Do what?

The PEI phase will initially operate with the platform in a nascent state, leveraging only onprocessor resources, such as the processor cache as a call stack, to dispatch Pre-EFI Initialization Modules (PEIMs). These PEIMs are responsible for the following:

- Initializing some permanent memory complement
- Describing the memory in Hand-Off Blocks (HOBs)
- Describing the firmware volume locations in HOBs
- Passing control into the Driver Execution Environment (DXE) phase

The PEI phase is also responsible for:

- Crisis recovery
- Resuming from the S3 sleep state.

PEI phase ---- Do what?

- Basic initialization for Chipset
- Memory Sizing
- Switch Stack to Memory
- Disable CAR
- Enable Cache
- Create HOBs
- Invoke DxeIpl (DXE Initial Program Loader)
- ✓ BIOS Recovery
- ✓ ACPI S3 Resume

Points of PEI phase

- Memory is not initialized
- Chipset is not initialized
- Code is executed on ROM
- Code is not compressed
- No CPU cache
- PEI Core code is not related to platform

PEI Elements

- PEI Core
  - Provide service for PEIM
  - Load and Execute PEIM
- PEI Service
  - PEI Core service routine, all PEIMs can use them
- PEIM
  - PEI Module
- PPI
  - PEI to PEI Interface, PEIM provides interface to others

PEI Flowchart

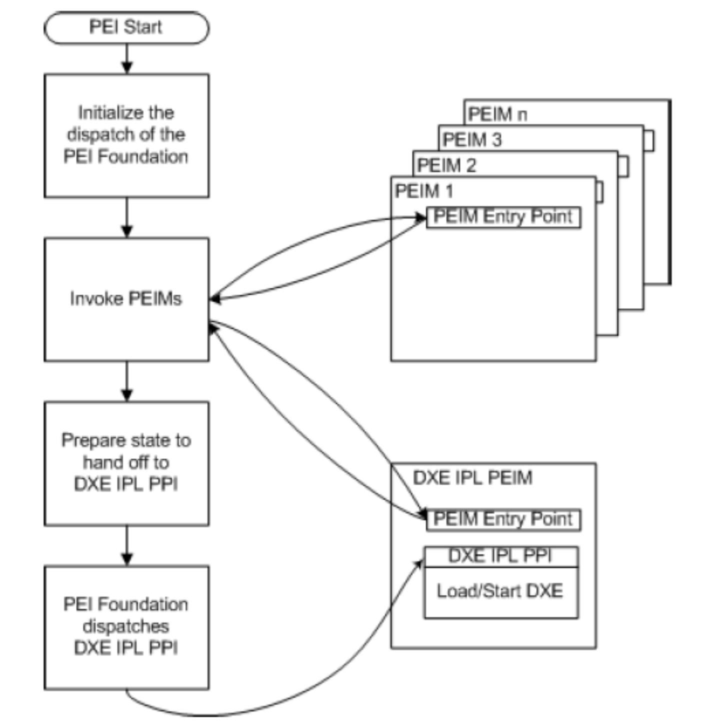
1.Initialize PEI Core

2.Execute PEIM based on dependency status

3.Different Boot Mode are likely using

different boot path

4.Execute DXE IPL


PEI Start
Initialize the dispatch of the PEI Foundation
Invoke PEIMs
Prepare state to hand off to DXE IPL PPI
PEI Foundation dispatches DXE IPL PPI
PEIM n
PEIM 3
PEIM 2
PEIM 1
PEIM Entry Point
DXE IPL PEIM
PEIM Entry Point
DXE IPL PPI
Load/Start DXE

PEI Flowchart

PEIM - Dispatcher

- Dispatcher PEIMs Order
- Dependency for PEIM priority
- Without Dependency, the order is in ROM image(Elink under FV_BB)

	Round 1	Round 2	Round 3	Round 4
IDE (PCICfg) (CPUIO)	Standby	Standby	Running	Ready
PCICfg (CPUIO)	Standby	Running	Ready	Ready
CPUIO	Running	Ready	Ready	Ready

PEI Flowchart

- PEI dependency sample code

```
SBPEI.inf
      [Depex]
       gEfiPeiCpuIoPpiInstalledGuid AND
       gEfiPciCfg2PpiGuid AND
       gEfiPeiReadOnlyVariable2PpiGuid
```

Supported opcode such as AND / OR / NOT / TURE etc…, please refer below PEI CIS spec charpter 5.7 Dependency Expression.

Platform Initialization Specification Pre-EFI Initialization Core Interface

PEI Flowchart

- PEI a *priori* file

the *a priori* file complements the dependency expression mechanism of PEI by stipulating a series of modules which need be dispatched in a prescribed order.

- Optional
- At most one a priori file per FV.

PEI a *priori* file

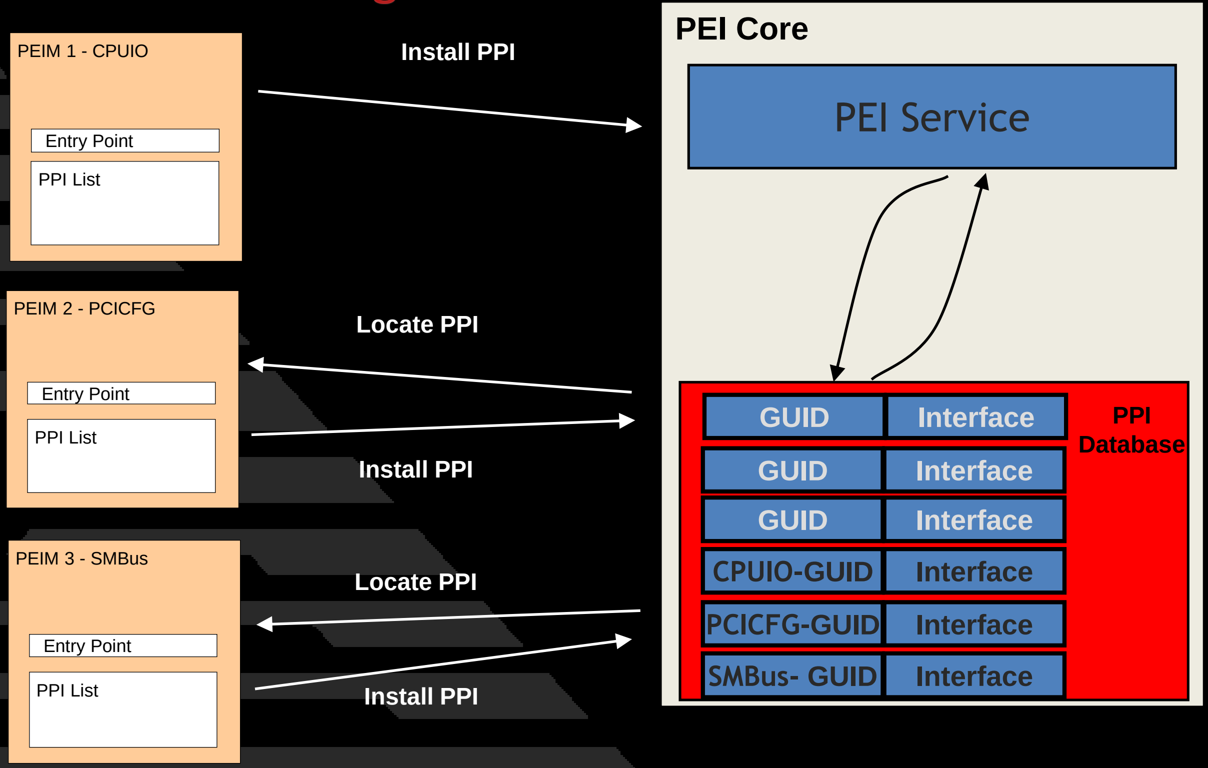
GUID

```
#define PEI_APRIORI_FILE_NAME_GUID \
{0x1b45cc0a, 0x156a, 0x428a, 0xaf62, 0x49, 0x86, \
0x4d, 0xa0, 0xe6, 0xe6}

typedef struct {
EFI_GUID FileNamesWithinVolume[NumberOfModulesInVolume];
// Optional list of file-names
} PEI_APRIORI_FILE_CONTENTS;
```

PEI Flowchart

PEIM & PPI Working Model



GUID	Interface
GUID	Interface
GUID	Interface
CPUIO-GUID	Interface
PCICFG-GUID	Interface
SMBus- GUID	Interface

PEI Sample Code

- Install a PPI

```
// GUID Definition(s)
EFI_GUID gAmiPeiClkGenGuid = AMI_PEI_CLKGEN_PPI_GUID;

// PPI Definition(s)
static AMI_PEI_CLKGEN_PPI mPeiClkGenPpi = {
    GetClkGenData,
    SetClkGenData
};

// PPI that are installed
static EFI_PEI_PPI_DESCRIPTOR mClkGenPpi[] = {
    { EFI_PEI_PPI_DESCRIPTOR_PPI | EFI_PEI_PPI_DESCRIPTOR_TERMINATE_LIST, \
      &gAmiPeiClkGenGuid, &mPeiClkGenPpi }
};

// Function Definition(s)
EFI_STATUS EFIAPI ClkGenPei_Init (
    IN EFI_FFS_FILE_HEADER      *FfsHeader,
    IN EFI_PEI_SERVICES         **PeiServices )
{
    EFI_STATUS                  Status;

    // Install the clock generator PPI
    Status = (*PeiServices)->InstallPpi( PeiServices, mClkGenPpi );
    ASSERT_PEI_ERROR( PeiServices, Status );

    return EFI_SUCCESS;
}
//GetClkGenData & SetClkGenData are the function or data you should implement.
```

PEI Sample Code

- Invoke PPI

```
 Status = (*PeiServices)->LocatePpi( PeiServices, \
                                    &gAmiPeiClkGenGuid, \
                                    0, \
                                    NULL,\
                                    &ClkGen );
if (Status != EFI_SUCCESS) return Status;

Status = ClkGen->GetClkGenData( PeiServices, \
                                ClkGen,
                                CLK_GEN_SMBUS_ADDR, \
                                &Length, \
                                ClockBuffer );
ASSERT_PEI_ERROR (PeiServices, Status);
```

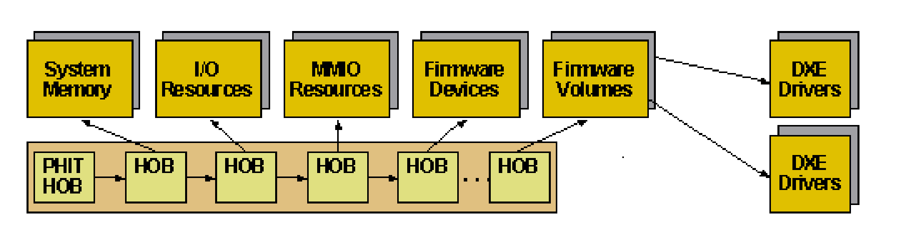
PEI HOB Introduce

- HOB: Hand off Block
- Why?

  PEI Phase pass some information to DXE Phase

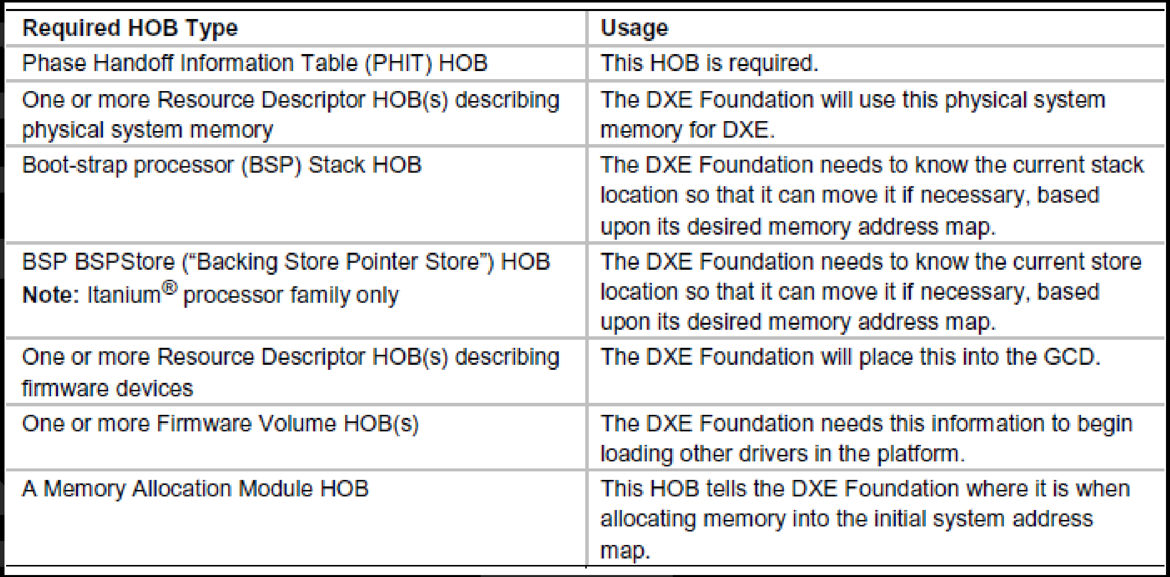
  THE DXE IPL PPI passes the HOB list from PEI to the DXE foundation when it invokes the DXE foundation.

- Data Structure
  - Every Block has its own GUID & Structure



PEI HOB Introduce

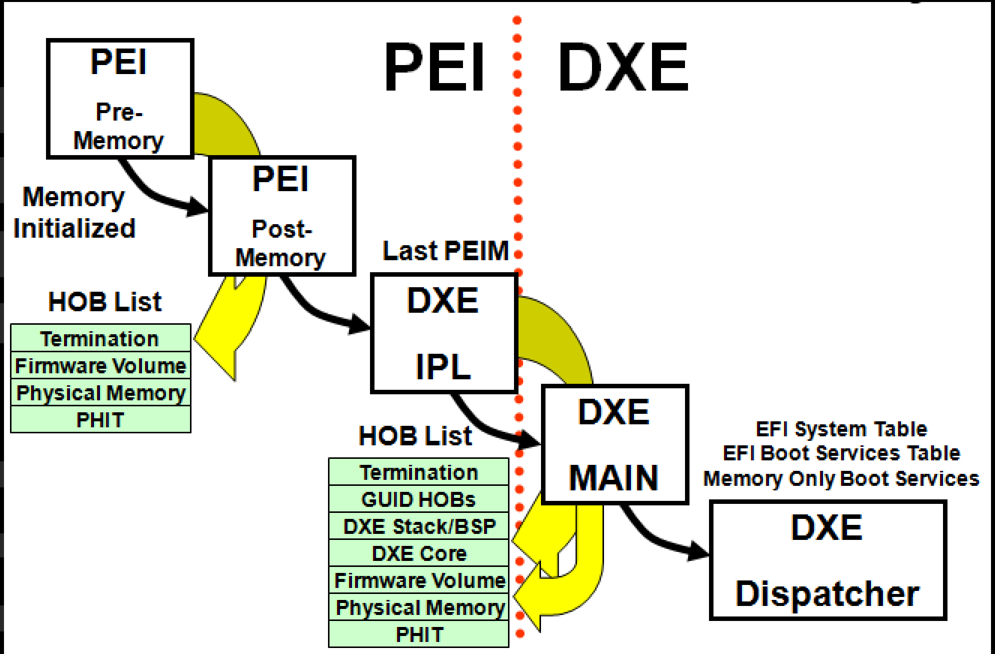
- The HOB list must contain at least the HOBs listed in the following table.



Required HOB Type	Usage
Phase Handoff Information Table (PHIT) HOB	This HOB is required.
One or more Resource Descriptor HOB(s) describing physical system memory	The DXE Foundation will use this physical system memory for DXE.
Boot-strap processor (BSP) Stack HOB	The DXE Foundation needs to know the current stack location so that it can move it if necessary, based upon its desired memory address map.
BSP BSPStore ("Backing Store Pointer Store") HOB **Note:** Itanium® processor family only	The DXE Foundation needs to know the current store location so that it can move it if necessary, based upon its desired memory address map.
One or more Resource Descriptor HOB(s) describing firmware devices	The DXE Foundation will place this into the GCD.
One or more Firmware Volume HOB(s)	The DXE Foundation needs this information to begin loading other drivers in the platform.
A Memory Allocation Module HOB	This HOB tells the DXE Foundation where it is when allocating memory into the initial system address map.

PEI HOB Introduce

- PEI HOB Transfer



HOB List
Termination
Firmware Volume
Physical Memory
PHIT

HOB List
Termination
GUID HOBs
DXE Stack/BSP
DXE Core
Firmware Volume
Physical Memory
PHIT

PEI HOB Sample Code

- Create a HOB

```
typedef struct {
         EFI_HOB_GUID_TYPE     EfiHobGuidType;
         OEM_DEFINE_TYPE                xxx;
} Demo_HOB;

Status = (*PeiServices)->CreateHob(
              PeiServices,
              EFI_HOB_TYPE_GUID_EXTENSION,
              sizeof(Demo_HOB),
              &DemoHob);
if (!EFI_ERROR(Status))
{
      DemoHob->EfiHobGuidType.Name = gDemoGuid;
      DemoHob->xxx = YYY;
}
```

HOB Types: (Each type has its own structure, especially, each Hob with type of EFI_HOB_TYPE_GUID_EXTENSION should define its own structure.)

```
#define EFI_HOB_TYPE_HANDOFF                    0x0001
#define EFI_HOB_TYPE_MEMORY_ALLOCATION          0x0002
#define EFI_HOB_TYPE_RESOURCE_DESCRIPTOR        0x0003
#define EFI_HOB_TYPE_GUID_EXTENSION             0x0004
#define EFI_HOB_TYPE_FV                         0x0005
#define EFI_HOB_TYPE_CPU                        0x0006
#define EFI_HOB_TYPE_MEMORY_POOL                0x0007
#define EFI_HOB_TYPE_CV                         0x0008
```

End