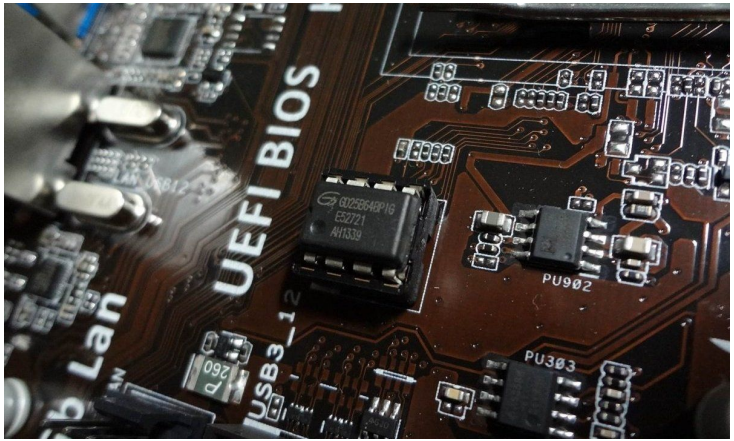
Firmware vulnerabilities/feature/backdoor?

Researching firmware vulnerabilities

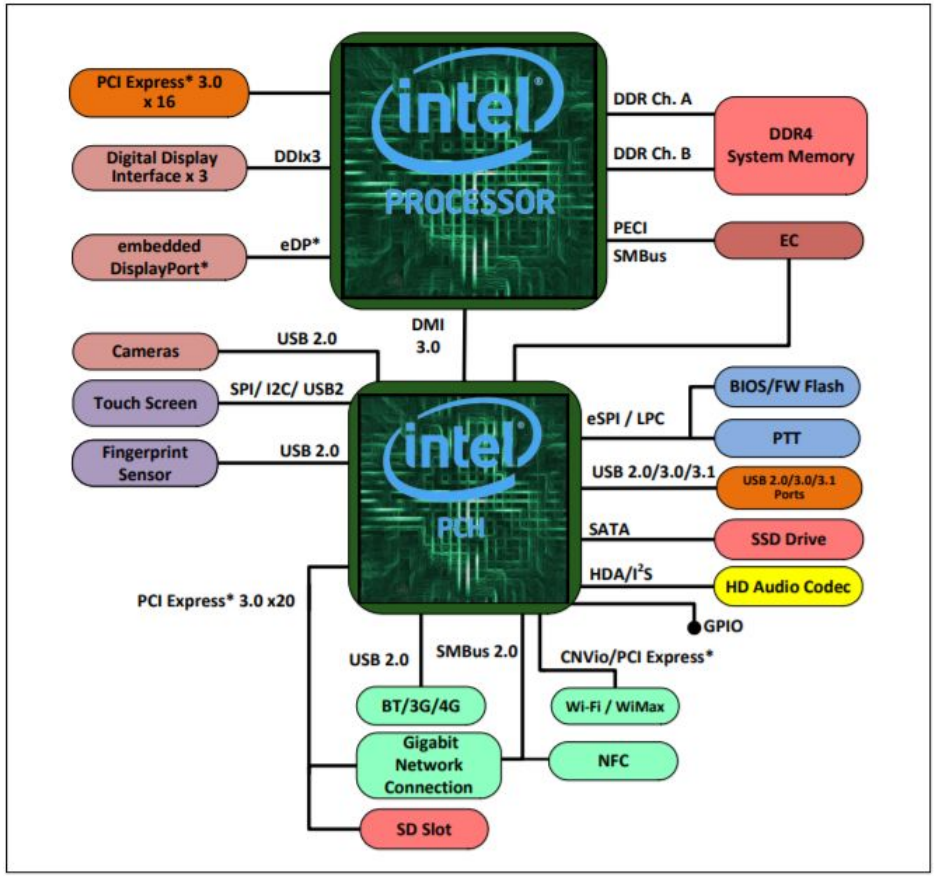
Emin Ghuliev
@drm_gh

Agenda

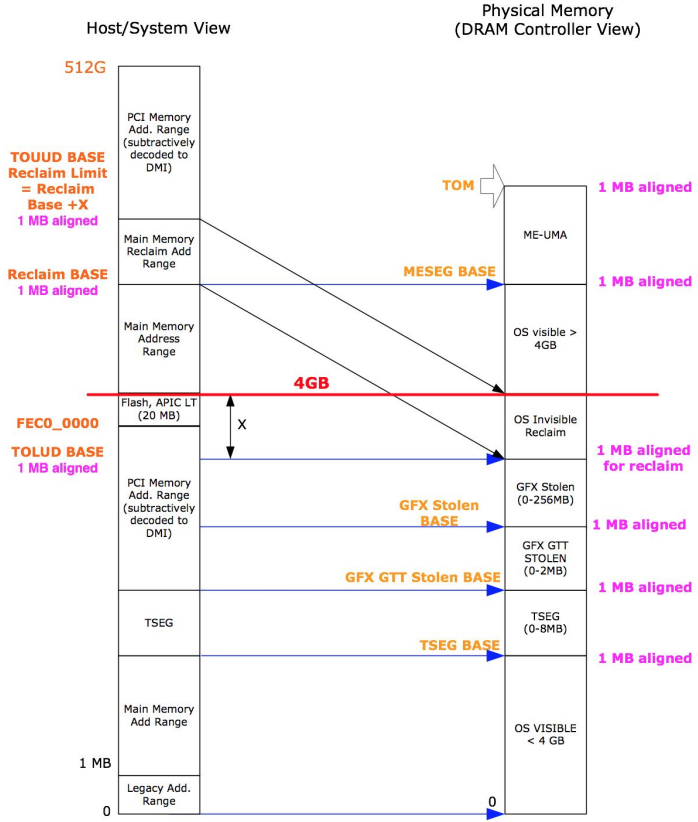
- Intel 8th Gen (S-platform) Processor
- PCI (and PCI Express)
- PCH (Platform Controller Hub)
- About UEFI/BIOS
- SPI Flash (with Protections)
- SMM Mode (SMRAM/SMI handlers)
- Attack vectors on SMM mode
- Persistency (e.g LoJaX or HackingTeam)



PC components

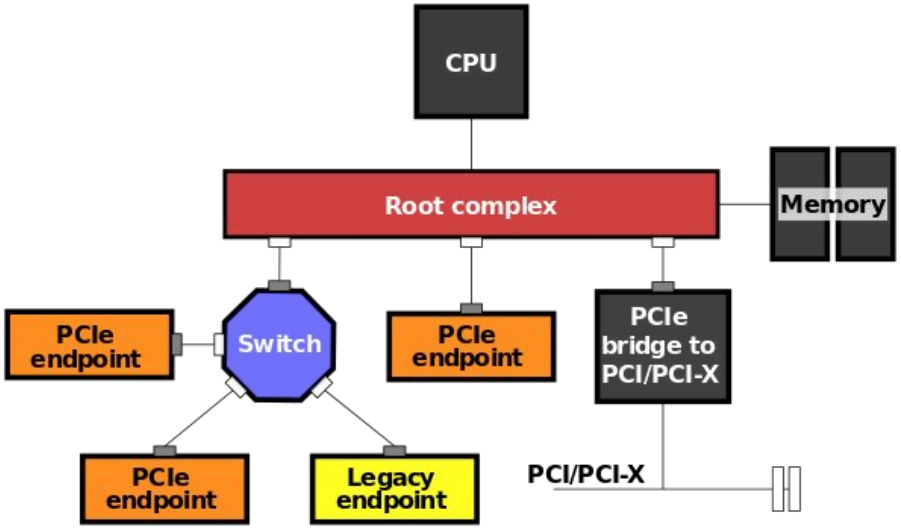


System Memory Map



PCI Express Topology

The Root Complex connects the processor to the system memory and components. Each PCIe/PCI peripheral is identified by a bus number, a device number, and a function number.



- The PCI specification permits a single system to host up to
  - Up to 256 PCIe buses
  - Up to 32 PCIe devices
  - Up to 8 Functions
  - Each Function can implement up to 4 KB of configuration space

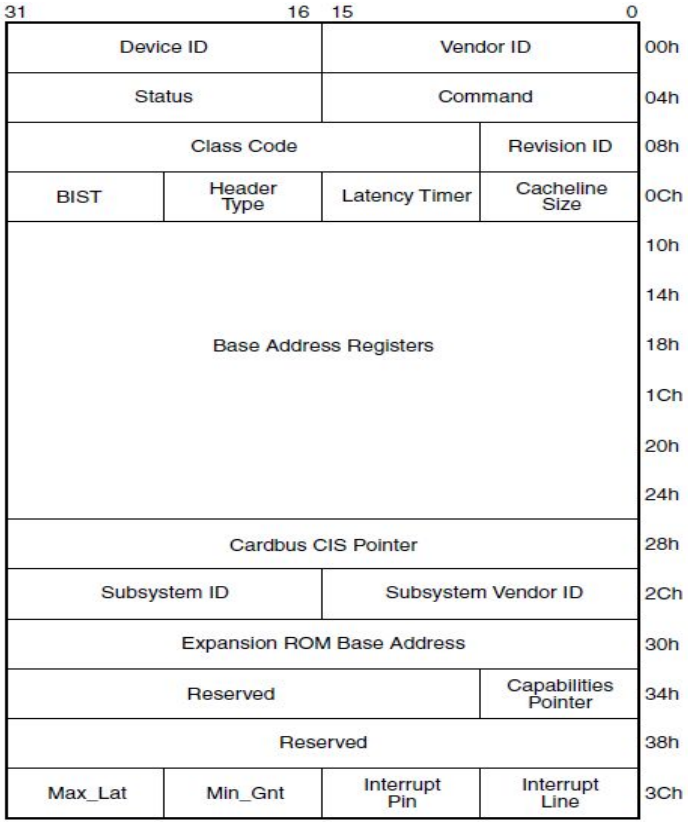
PCI Express (PCIe) Address Spaces

PCIe implements four address spaces:

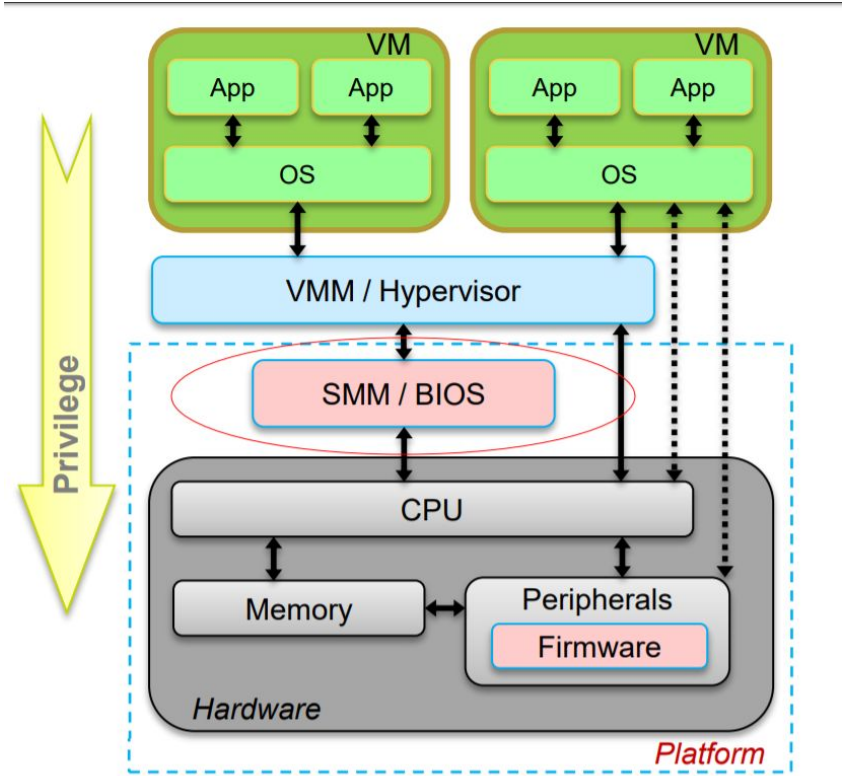
- 1) PCIe Configuration Space (up to 4KBytes)
  - Every PCIe device has its configuration space mapped to memory.
  - Also provides the first 256 bytes of compatible PCI (memory-mapped and via port IO for backwards compatibility).
- 2) PCIe Memory-mapped space
  - Memory-mapped I/O uses the same address space to address both memory and I/O devices. The memory and registers of the I/O devices are mapped to (associated with) address values.
- 3) PCIe I/O-mapped space
  - Port-mapped I/O often uses a special class of CPU instructions designed specifically for performing I/O operations.

Source: wikipedia (where is 4th? We don’t need it)

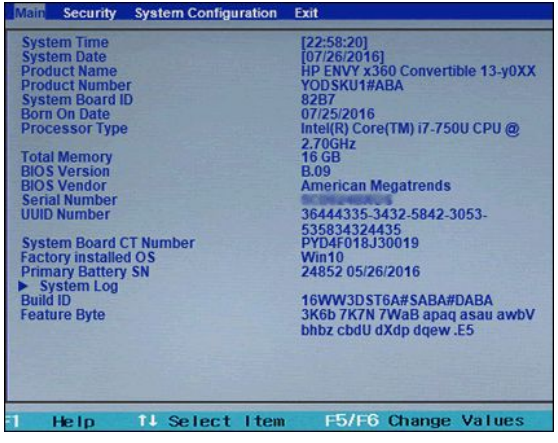
PCIe configuration space

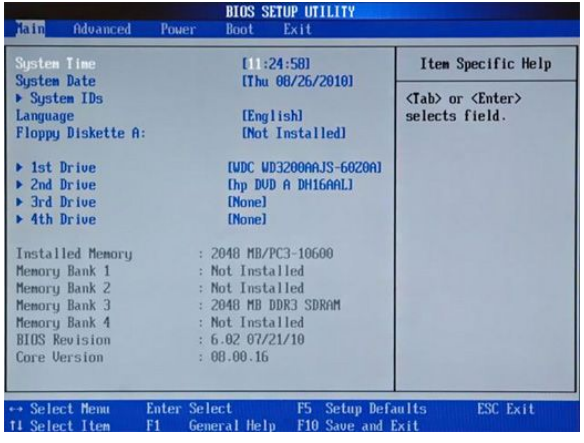


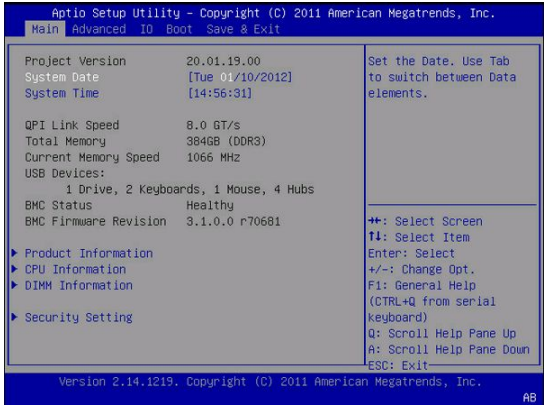
Where is BIOS/UEFI/SMM?



Legacy BIOS


Main Security System Configuration Exit
System Time [22:58:20]
System Date [07/26/2016]
Product Name HP ENVY x360 Convertible 13-y0XX
Product Number YODSKU1#ABA
System Board ID 82B7
Born On Date 07/25/2016
Processor Type Intel(R) Core(TM) i7-750U CPU @ 2.70GHz
Total Memory 16 GB
BIOS Version B.09
BIOS Vendor American Megatrends
Serial Number
UUID Number 36444335-3432-5842-3053-535834324435
System Board CT Number PYD4F018J30019
Factory installed OS Win10
Primary Battery SN 24852 05/26/2016
System Log
Build ID 16WW3DST6A#SABA#DABA
Feature Byte 3K6b 7K7N 7WaB apaq asau awbV bhbz cbdU dXdp dqew .E5
F1 Help ↑↓ Select Item F5/F6 Change Values


BIOS SETUP UTILITY
Main Advanced Power Boot Exit
System Time [11:24:58]
System Date [Thu 08/26/2010]
System IDs
Language [English]
Floppy Diskette A: [Not Installed]
1st Drive [WDC WD3200AAJS-60Z0A]
2nd Drive [hp DVD A DH16AAL]
3rd Drive [None]
4th Drive [None]
Installed Memory : 2048 MB/PC3-10600
Memory Bank 1 : Not Installed
Memory Bank 2 : Not Installed
Memory Bank 3 : 2048 MB DDR3 SDRAM
Memory Bank 4 : Not Installed
BIOS Revision : 6.02 07/21/10
Core Version : 08.00.16
Item Specific Help
<Tab> or <Enter> selects field.
↔ Select Menu Enter Select F5 Setup Defaults ESC Exit
↑↓ Select Item F1 General Help F10 Save and Exit


Aptio Setup Utility – Copyright (C) 2011 American Megatrends, Inc.
Main Advanced IO Boot Save & Exit
Project Version 20.01.19.00
System Date [Tue 01/10/2012]
System Time [14:56:31]
QPI Link Speed 8.0 GT/s
Total Memory 384GB (DDR3)
Current Memory Speed 1066 MHz
USB Devices:
1 Drive, 2 Keyboards, 1 Mouse, 4 Hubs
BMC Status Healthy
BMC Firmware Revision 3.1.0.0 r70681
Product Information
CPU Information
DIMM Information
Security Setting
Set the Date. Use Tab to switch between Data elements.
→←: Select Screen
↑↓: Select Item
Enter: Select
+/-: Change Opt.
F1: General Help
(CTRL+Q from serial keyboard)
Q: Scroll Help Pane Up
A: Scroll Help Pane Down
ESC: Exit
Version 2.14.1219. Copyright (C) 2011 American Megatrends, Inc.
AB

BIOS Stage

- 1. CPU Reset vector in BIOS ’ROM’ (Boot Block) - >

  Boot Block

- 1. Basic CPU, chipset initialization
- 2. Initialize CAR, load and run from cache
- 3. Initialize DRAM memory

  POST (Power-On Self Test)

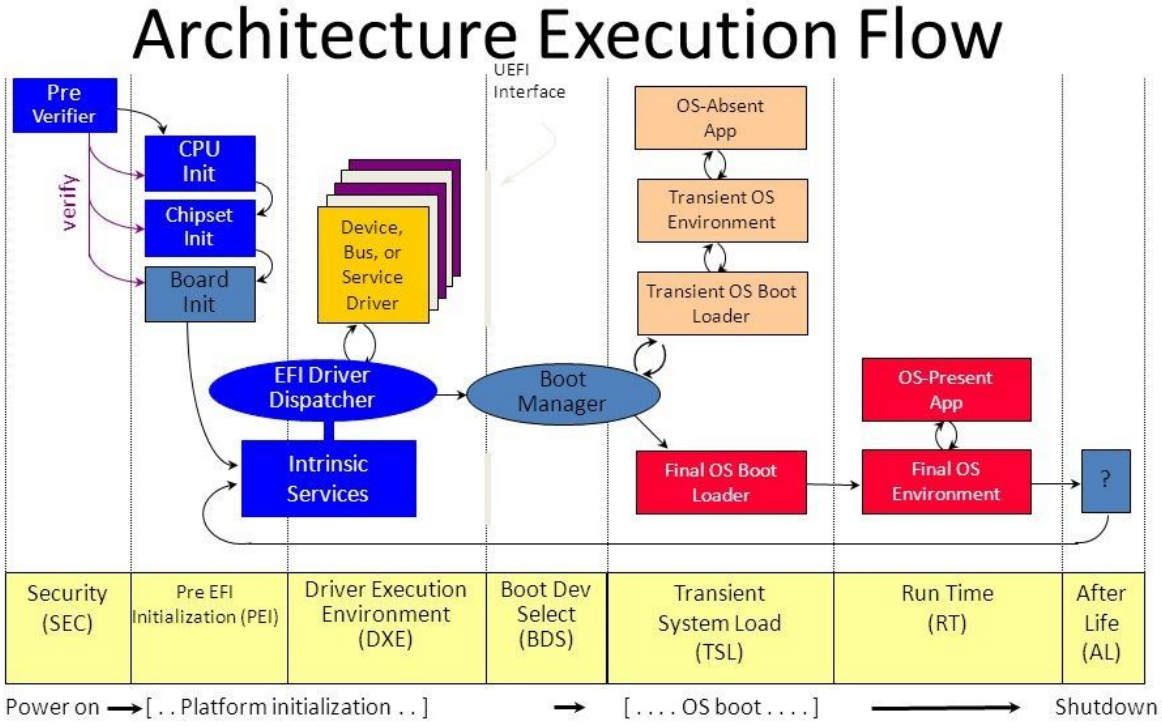
- 1. Decompress and relocate system BIOS in DRAM
- 2. Enumerate add-on devices (ISA, PCI)..
- 3. Execute Option ROMs on expansion cards

  INT 19h

- 1. Locate, load and execute Initial Boot Loader code in MBR (at 0x7C00 PA)
- 2. 2nd Stage Boot Loader OS Loader OS kernel

UEFI

Boot Execution Flow



Also UEFI flow

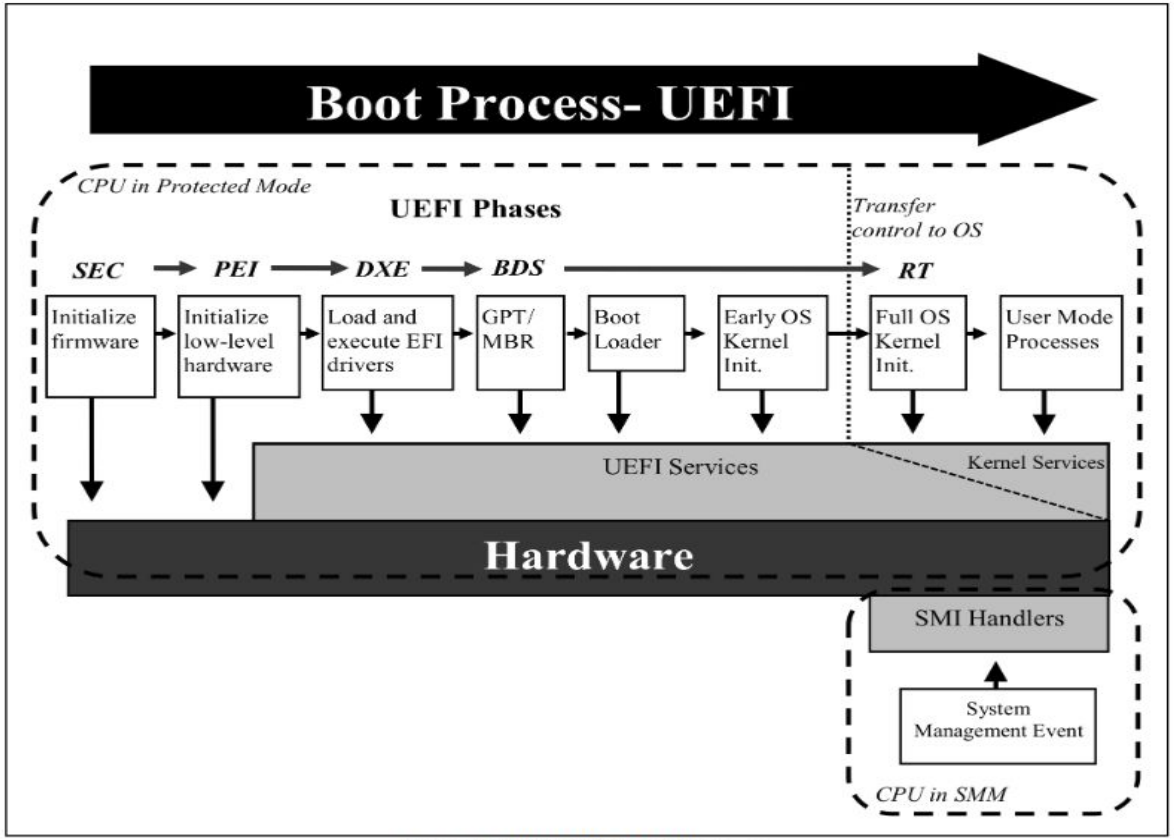
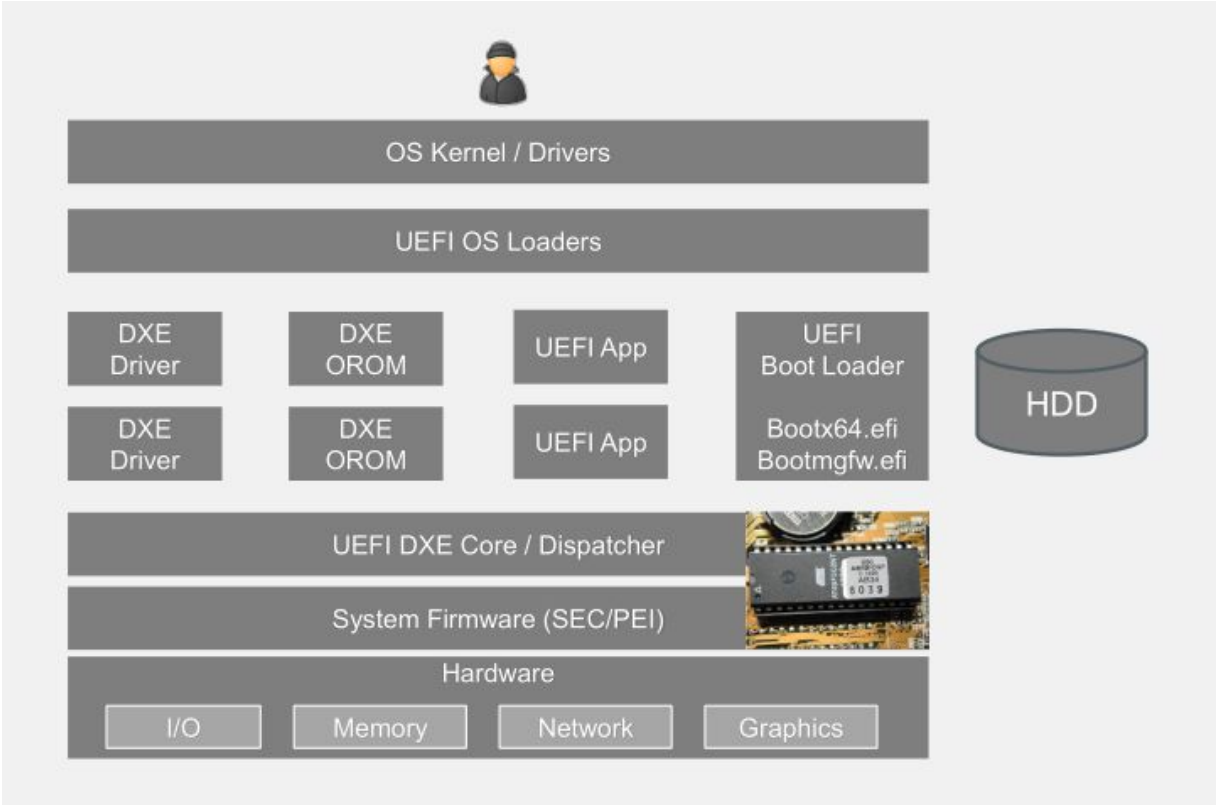


Figure 2: UEFI BIOS Boot Process

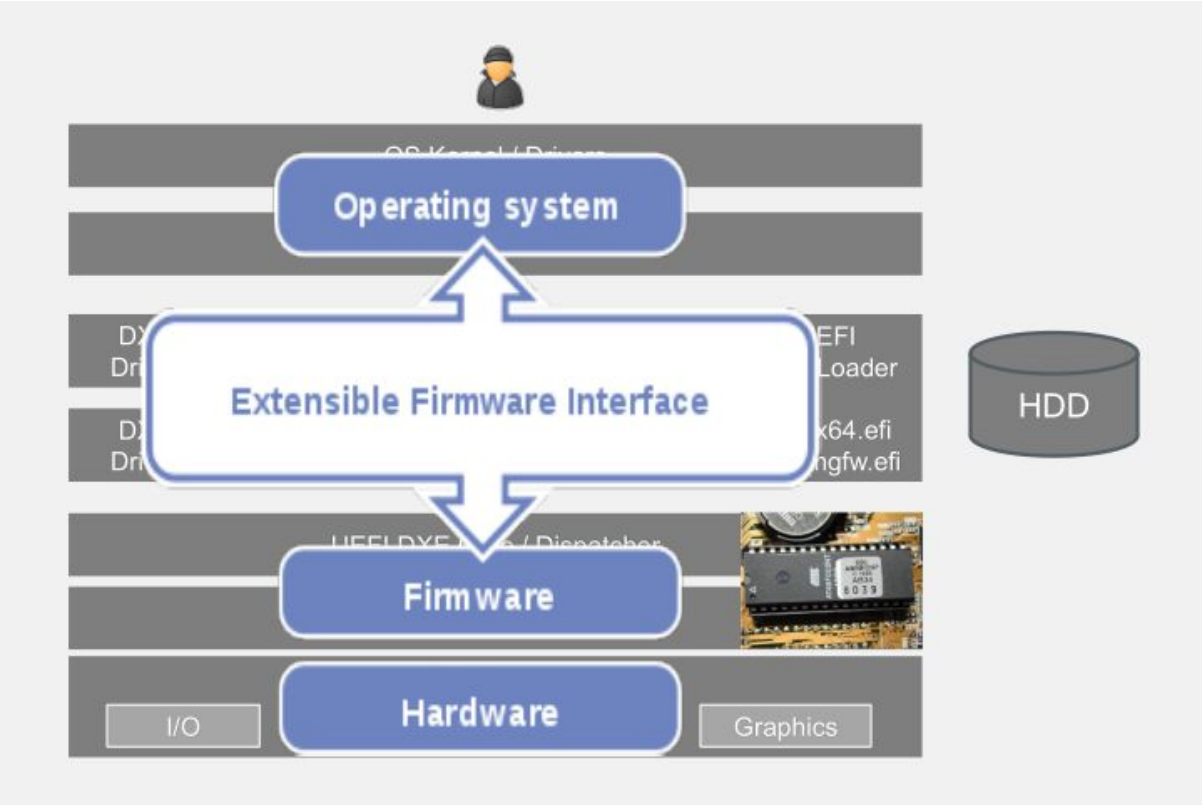
UEFI

- Industry Standard Interface Between Firmware & OS
- Processor Architecture and OS Independent
- initially developed by Intel as BIOS replacement for IA64
- C Development Environment (EDK2/UDK)
- Rich GUI Pre-Boot Application Environment
- Includes Modular Driver Model
- UEFI executables: PE/COFF or TE executable files
- Secure Boot of Microsoft Windows 8 or above requires UEFI

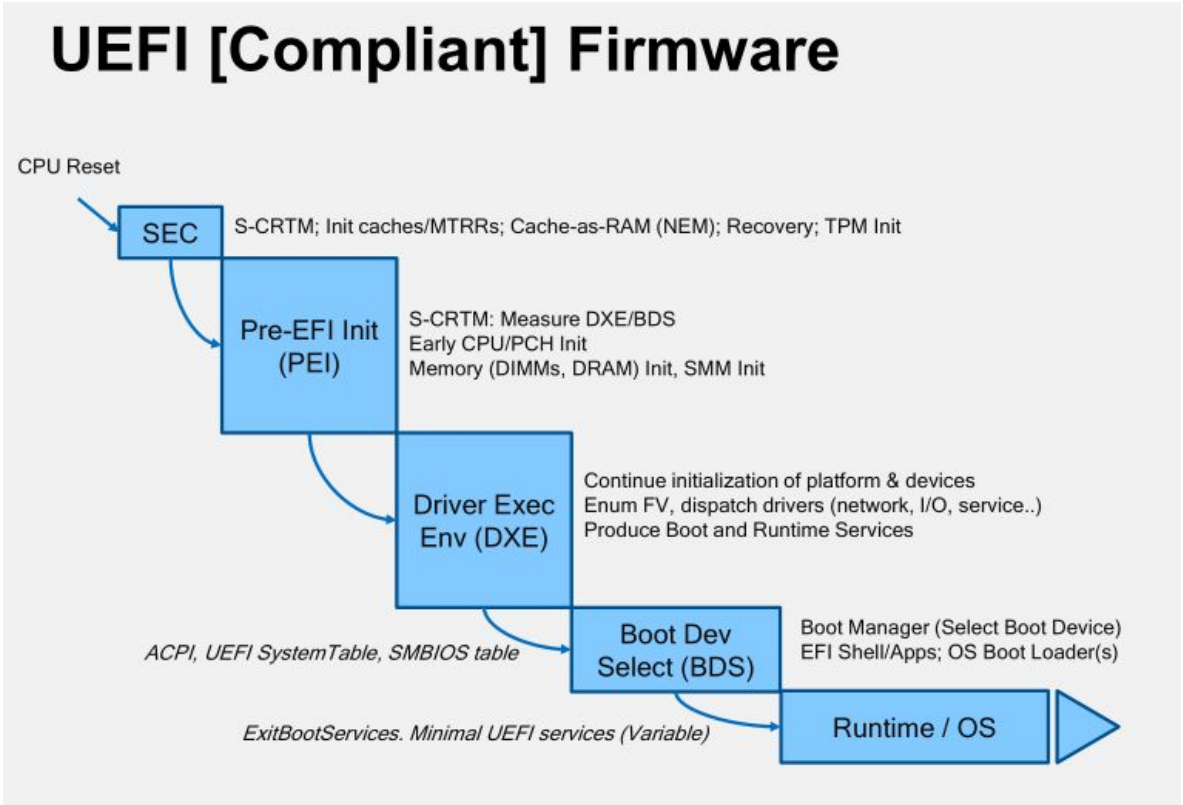
UEFI Firmware



UEFI Firmware



UEFI stages


UEFI [Compliant] Firmware
CPU Reset
SEC
S-CRTM; Init caches/MTRRs; Cache-as-RAM (NEM); Recovery; TPM Init
Pre-EFI Init (PEI)
S-CRTM: Measure DXE/BDS
Early CPU/PCH Init
Memory (DIMMs, DRAM) Init, SMM Init
Driver Exec Env (DXE)
Continue initialization of platform & devices
Enum FV, dispatch drivers (network, I/O, service..)
Produce Boot and Runtime Services
Boot Dev Select (BDS)
Boot Manager (Select Boot Device)
EFI Shell/Apps; OS Boot Loader(s)
ACPI, UEFI SystemTable, SMBIOS table
ExitBootServices. Minimal UEFI services (Variable)
Runtime / OS

DXE stage

- 64 bit code on most machines, exceptions are rare
- binaries stored in PE32+ format
- DXE Core, Dispatcher, Drivers, OROMs and Applications
- DXE Protocols
- SMM Core, Dispatcher, Drivers and SMI Handlers
- DXE dependency expressions
- BootServices and RuntimeServices
- dispatches all available drivers and starts BDS application

Information About MMIO

```
[*] running module: chipsec.modules.tools.vmm.pcie_overlap_fuzz
[x][ ================================================================
[x][ Module: Tool to overlap and fuzz MMIO spaces of available PCIe devices
[x][ ================================================================
[*] Enumerating available PCIe devices..
[*] About to fuzz the following PCIe devices..
BDF     | VID:DID   | Vendor                          | Device
-------------------------------------------------------------------------
00:00.0 | 8086:7192 | Intel Corporation               | 440BX/ZX chipset Host-to-PCI Bridge
00:07.0 | 8086:7110 | Intel Corporation               | Intel 82371AB/EB PCI to ISA bridge (ISA mode)
00:07.1 | 8086:7111 | Intel Corporation               | Intel(R) 82371AB/EB PCI Bus Master IDE Controller
00:07.3 | 8086:7113 | Intel Corporation               | PIIX4/4E/4M Power Management Controller
00:08.0 | 1414:5353 |                                 |
[*] overlapping MMIO bars...
[*] overlapping MMIO bars...
[*] overlapping MMIO bars...
[*] overlapping MMIO bars...
[*] overlapping MMIO bars...
```

Serial Peripheral Interface (SPI)

Overview

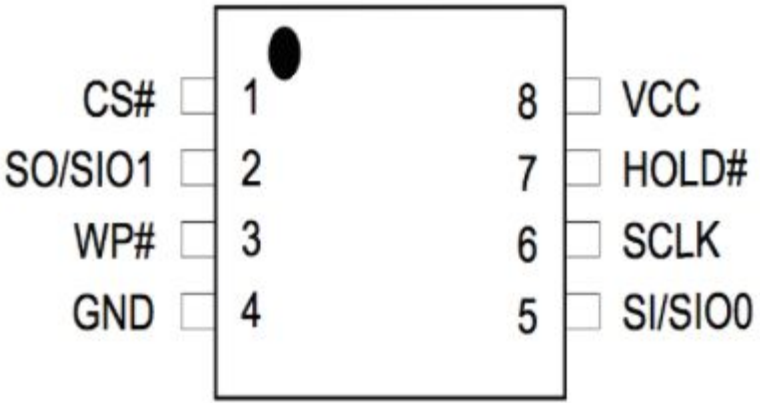
The PCH provides one Serial Peripheral Interface (SPI0) to support the system flash and TPM devices. The interface implements 3 Chip Select signals (CS#), allowing up to two flash devices and one TPM device to be connected to the PCH. The CS0# and CS1# are used for flash devices and CS2# is dedicated to TPM.

The SPI interfaces support either 1.8V or 3.3V.

The SPI interface covered in this chapter is for flash and TPM support only. This interface is distinct from other SPI described in this document such as the Generic SPI (GSPI).

SPI Flash

- Intel’s ICH/PCH implements a SPI interface for the BIOS flash device.
- Flash memory is an electronic non-volatile computer storage medium that can be electrically erased and reprogrammed.
- SPI is required in order to support the Management Engine (ME), Gigabit Ethernet (GbE), and others.



SPI Flash Regions

Region	Content
0	Flash Descriptor
1	BIOS
2	Intel Management Engine
3	Gigabit Ethernet
4	Platform Data
8	EC

SPI controller security

- SMM based BIOS Write Protection: write-protects entire BIOS region from software other than SMI handler firmware executing in SMM
- SPI Protected Range registers (PR0-PR4): read/write protection of SPI flash regions based on FLA for program register access
- Flash Descriptor based access control: defines read/write access to each flash region by each master

Global BIOS write protection

13.1.33 BIOS_CNTL—BIOS Control Register (LPC I/F—D31:F0)

Offset Address:	DCh	Attribute:	R/WLO, R/W, RO
Default Value:	20h	Size:	8 bit
Lockable:	No	Power Well:	Core

Bit	Description
7:6	Reserved
5	**SMM BIOS Write Protect Disable (SMM_BWP)**— R/WLO. This bit set defines when the BIOS region can be written by the host. 0 = BIOS region SMM protection is disabled. The BIOS Region is writable regardless if processors are in SMM or not. (Set this field to 0 for legacy behavior) 1 = BIOS region SMM protection is enabled. The BIOS Region is not writable unless all processors are in SMM.
4	**Top Swap Status (TSS)** — RO. This bit provides a read-only path to view the state of the Top Swap bit that is at offset 3414h, bit 0.
3:2	**SPI Read Configuration (SRC)** — R/W. This 2-bit field controls two policies related to BIOS reads on the SPI interface: Bit 3 – Prefetch Enable Bit 2 – Cache Disable Settings are summarized below: **Bits 3:2** — **Description** 00b — **No prefetching, but caching enabled.** 64B demand reads load the read buffer cache with "valid" data, allowing repeated code fetches to the same line to complete quickly 01b — **No prefetching and no caching.** One-to-one correspondence of host BIOS reads to SPI cycles. This value can be used to invalidate the cache. 10b — **Prefetching and Caching enabled.** This mode is used for long sequences of short reads to consecutive addresses (that is, shadowing). 11b — **Reserved. This is an invalid configuration**, caching must be enabled when prefetching is enabled.
1	**BIOS Lock Enable (BLE)** — R/WLO. 0 = Setting the BIOSWE will not cause SMIs. 1 = Enables setting the BIOSWE bit to cause SMIs. Once set, this bit can only be cleared by a PLTRST#
0	**BIOS Write Enable (BIOSWE)** — R/W. 0 = Only read cycles result in Firmware Hub I/F cycles. 1 = Access to the BIOS space is enabled for both read and write cycles. When this bit is written from a 0 to a 1 and BIOS Lock Enable (BLE) is also set, an SMI# is generated. This ensures that only SMI code can update BIOS.

In my arch BIOS_CNTL is located in the LPC device (PCI configuration space).

1	**BIOS Lock Enable (BLE)** — R/WLO. 0 = Setting the BIOSWE will not cause SMIs. 1 = Enables setting the BIOSWE bit to cause SMIs. Once set, this bit can only be cleared by a PLTRST#
0	**BIOS Write Enable (BIOSWE)** — R/W. 0 = Only read cycles result in Firmware Hub I/F cycles. 1 = Access to the BIOS space is enabled for both read and write cycles. When this bit is written from a 0 to a 1 and BIOS Lock Enable (BLE) is also set, an SMI# is generated. This ensures that only SMI code can update BIOS.

Insecure state: BIOS_CNTL.BIOSWE (bit 0) enables write access to the flash chip

Additional protection PRx

- New version of Flash Write Protection mechanism
- The fact that PRx works independently from SMM code
- There are 5 Protected Range registers (0-4)
- Setting these will prevent reads and/or writes until the system is reset.

PRx Register contents

21.1.13 PR0—Protected Range 0 Register (SPI Memory Mapped Configuration Registers)

Memory Address: SPIBAR + 74h
Default Value: 00000000h
Attribute: R/W
Size: 32 bits

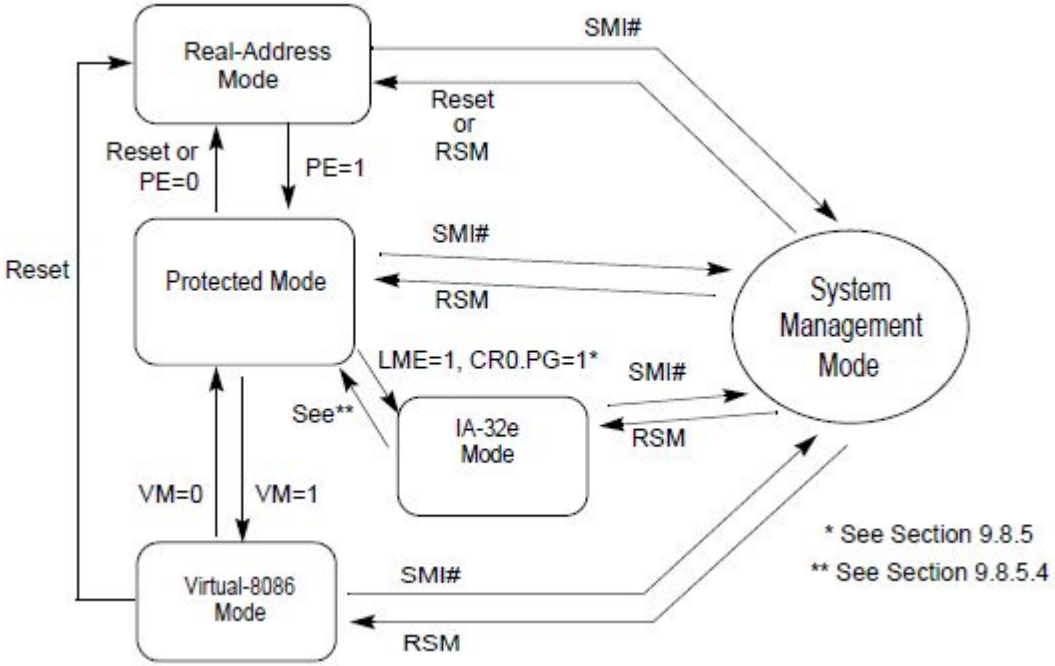
Note: This register can not be written when the FLOCKDN bit is set to 1.

Bit	Description
31	**Write Protection Enable**—R/W. When set, this bit indicates that the Base and Limit fields in this register are valid and that writes and erases directed to addresses between them (inclusive) must be blocked by hardware. The base and limit fields are ignored when this bit is cleared.
30:29	Reserved
28:16	**Protected Range Limit**—R/W. This field corresponds to FLA address bits 24:12 and specifies the upper limit of the protected range. Address bits 11:0 are assumed to be FFFh for the limit comparison. Any address greater than the value programmed in this field is unaffected by this protected range.
15	**Read Protection Enable**—R/W. When set, this bit indicates that the Base and Limit fields in this register are valid and that read directed to addresses between them (inclusive) must be blocked by hardware. The base and limit fields are ignored when this bit is cleared.
14:13	Reserved
12:0	**Protected Range Base**—R/W. This field corresponds to FLA address bits 24:12 and specifies the lower base of the protected range. Address bits 11:0 are assumed to be 000h for the base comparison. Any address less than the value programmed in this field is unaffected by this protected range.

SMM BIOS Write Protect Disable in BIOS_CNTL

Bit	Description
7:6	Reserved
5	**SMM BIOS Write Protect Disable (SMM_BWP)**—R/WL. This bit set defines when the BIOS region can be written by the host. 0 = BIOS region SMM protection is disabled. The BIOS Region is writable regardless if processors are in SMM or not. (Set this field to 0 for legacy behavior). 1 = BIOS region SMM protection is enabled. The BIOS Region is not writable unless all processors are in SMM and BIOS Write Enable (BIOSWE) is set to '1'.

x86 System Management Mode (SMM)



System Management Mode (SMM) is an operating mode on x86 and x86-64 processors, intended for use by Firmware/BIOS to perform low-level system management operations while an OS is running.

System Management Mode

- When the processor enters SMM, all other running tasks are suspended
- SMM can be invoked only by a System Management Interrupt (SMI) and exited only by the RSM (resume) instruction
- SMM can only be invoked by signaling a System Management Interrupt (SMI)
- SMI handler firmware is executing in SMM
- CPU (OS) state is saved in SMRAM upon entry to SMM and restored upon exit from SMM
- SMRAM is a range of DRAM reserved by BIOS for runtime part - SMI handlers
- CPU exits SMM to the interrupted OS when SMI handler executes RSM instruction (“Resume from SMM”)

SMRAM

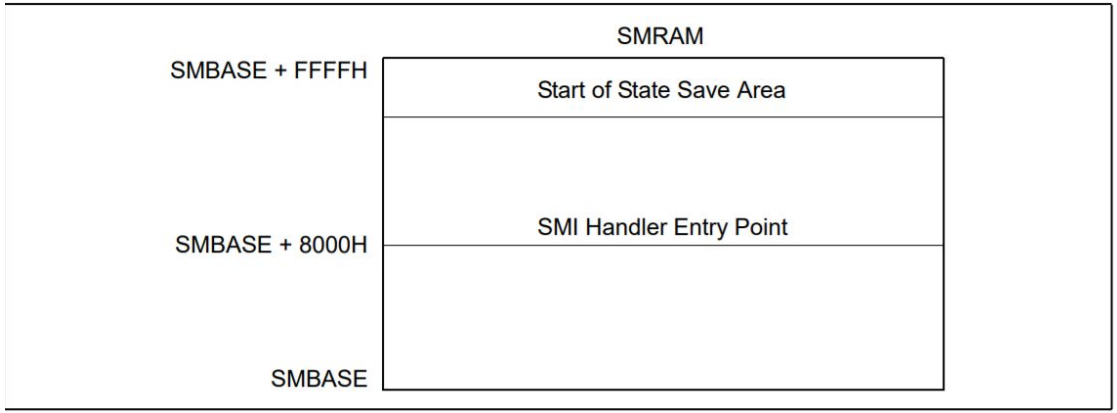


Figure 34-1. SMRAM Usage

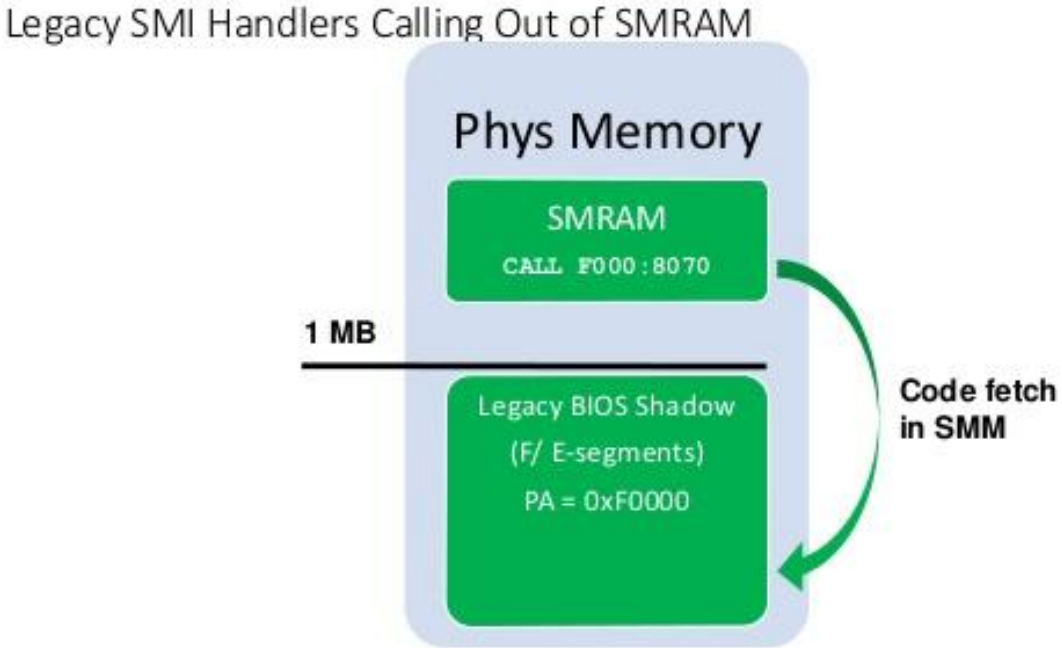
UEFITool NE alpha 52 - tsk0000.dta

File Action View Help

Structure

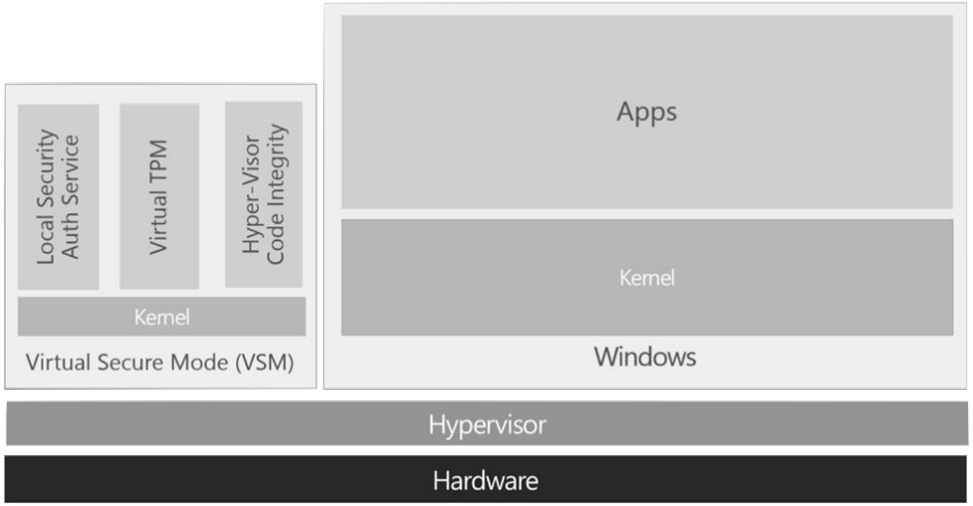
Name	Ac	Type	Subtype	Text
9A8F82D5-39B1-48DA-92DC-A22DA8834DF6		File	Freeform	MeSsdt
7D279373-EECC-4D4F-AE2F-CEC4B706B06A		File	Freeform	Tpm2AcpiTables
6FD18837-36E6-46EC-8F61-6730E3E23D50		File	DXE driver	JhiDxe
B8FE3D49-DCF3-4CBB-8070-47B4F5A34559		File	DXE driver	GopDebugDxe
PciBusDxe		File	DXE driver	PciBusDxe
SmbiosDxe		File	DXE driver	SmbiosDxe
EnglishDxe		File	DXE driver	EnglishDxe
ScsiBus		File	DXE driver	ScsiBus
ScsiDisk		File	DXE driver	ScsiDisk
S3SaveStateDxe		File	DXE driver	S3SaveStateDxe
SmmS3SaveState		File	SMM module	SmmS3SaveState
9EF82BFE-707F-468F-A944-A61626E47DC9		File	DXE driver	AcpiGlobalVariable
AcpiS3SaveDxe		File	DXE driver	AcpiS3SaveDxe
BootScriptExecutorDxe		File	DXE driver	BootScriptExecutorDxe

SMM attacking mode

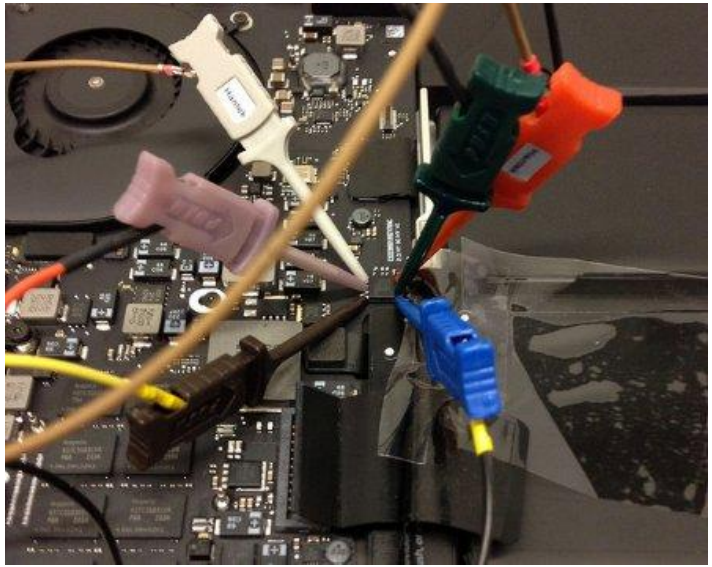


Breaking VSM

VSM is a protected virtual machine that runs on Hyper-V hypervisor separately from host Windows 10 system and its kernel. VSM has it's own isolated kernel mode and user mode, on Credential Guard enabled systems part of Local Security Subsystem Service (LSASS) that is responsible for keeping credentials in memory is running as isolated user mode process inside VSM.



My Firmware Researching (Bypass PRx registers)



PRx (offset)	Value	Base	Limit	WP?	RP?
PR0 (84)	831F031D	0031D000	0031FFFF	1	0
PR1 (88)	86CF0320	00320000	006CFFFF	1	0
PR2 (8C)	86DF06D0	006D0000	006DFFFF	1	0
PR3 (90)	87FF06E0	006E0000	007FFFFF	1	0
PR4 (94)	00000000	00000000	00000000	0	0

All protection bits had been cleared after S3 mode

PRx (offset)	Value	Base	Limit	WP?	RP?
PR0 (84)	00000000	00000000	00000000	0	0
PR1 (88)	00000000	00000000	00000000	0	0
PR2 (8C)	00000000	00000000	00000000	0	0
PR3 (90)	00000000	00000000	00000000	0	0
PR4 (94)	00000000	00000000	00000000	0	0

SMI handlers code quality is very poor

LoJaX infection stage for Persistency

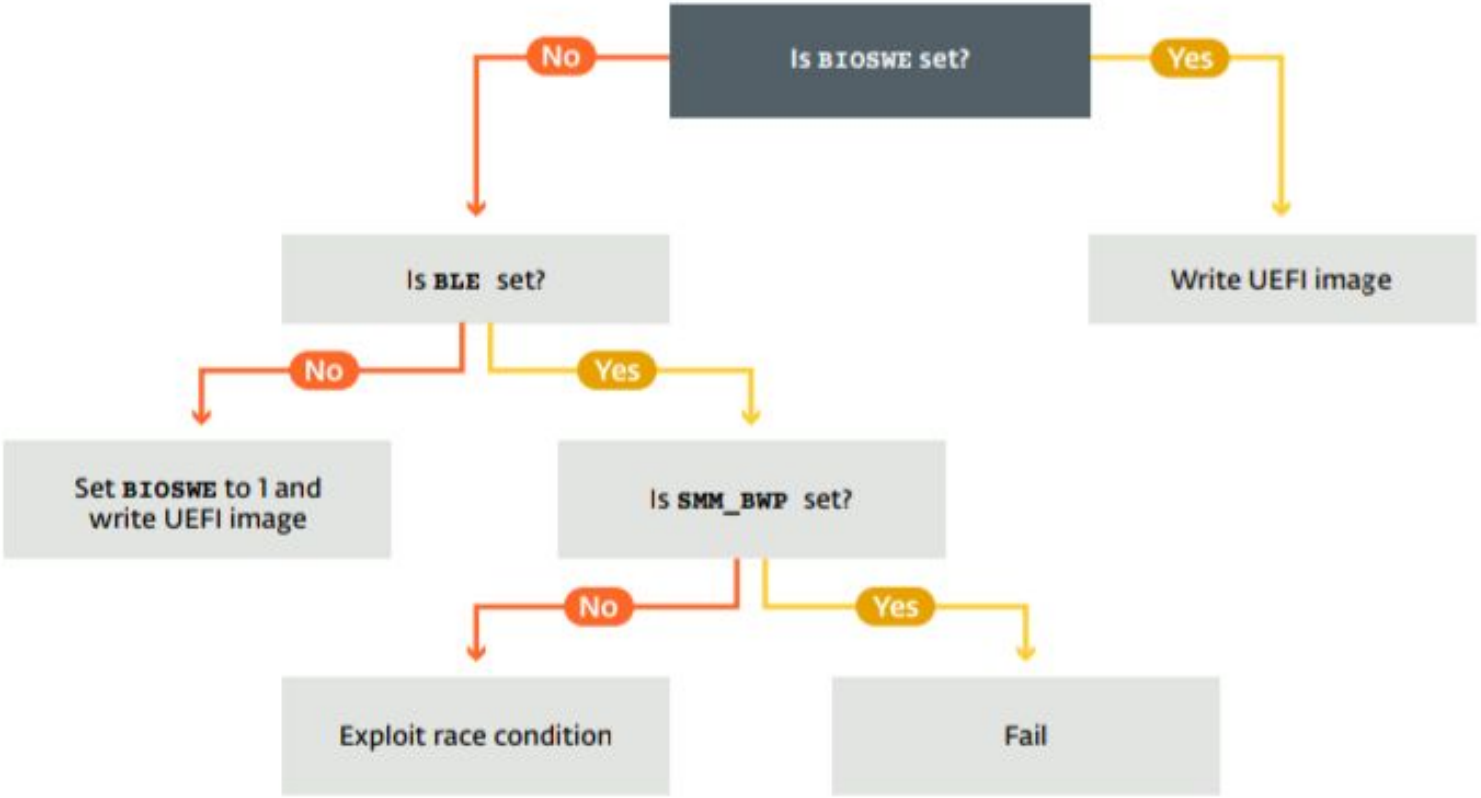
LoJax uses RWEverything driver in order to read and write SPI Flash memory where the UEFI modules are stored.

IOCTL for Kernel Driver and Malware

Table 1 **RwDrv.sys supported IOCTLs**

IOCTL code	Description
0x22280c	Writes to memory mapped I/O space
0x222808	Reads from memory mapped I/O space
0x222840	Reads a dword from given PCI Configuration Register
0x222834	Writes a byte to given PCI Configuration Register

Bypass SMM protection



Q&A

Thank you!