

# Advanced x86: BIOS and System Management Mode Internals *SMI Suppression*

Xeno Kovah && Corey Kallenberg

LegbaCore, LLC



# All materials are licensed under a Creative Commons “Share Alike” license.

<http://creativecommons.org/licenses/by-sa/3.0/>

## You are free:



to **Share** — to copy, distribute and transmit the work



to **Remix** — to adapt the work

## Under the following conditions:



**Attribution** — You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).



**Share Alike** — If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

Attribution condition: You must indicate that derivative work

"Is derived from John Butterworth & Xeno Kovah's 'Advanced Intel x86: BIOS and SMM' class posted at <http://opensecuritytraining.info/IntroBIOS.html>"

# SMI Suppression

- SMM stands as the first line of defense for protecting the BIOS flash from being overwritten
  - We'll cover how in the flash BIOS portion of the course
- What if the attacker simply suppressed SMI from being generated?
- They can, if the system isn't locked down properly:

## **SMI\_EN—SMI Control and Enable Register**

I/O Address:	PMBASE + 30h	Attribute:	R/W, R/WO, WO
Default Value:	00000000h	Size:	32 bit
Lockable:	No	Usage:	ACPI or Legacy
Power Well:	Core		

I/O Controller Hub Family 9

# SMI\_EN: SMI Control and Enable Register

## SMI\_EN—SMI Control and Enable Register

I/O Address:	PMBASE + 30h	Attribute:	R/W, R/WO, WO
Default Value:	00000000h	Size:	32 bit
Lockable:	No	Usage:	ACPI or Legacy
Power Well:	Core		

0	<b>GBL_SMI_EN</b> — R/W. 0 = No SMI# will be generated by ICH9. This bit is reset by a PCI reset event. 1 = Enables the generation of SMI# in the system upon any enabled SMI event. <b>NOTE:</b> When the SMI_LOCK bit is set, this bit cannot be changed.
---	--

- Located in the Power Management IO Registers (memory-mapped at PMBASE defined in LPC D31:F0)
- The SMI\_EN register can enable or disable some very specific instances of SMI# or globally enable/disable all SMI#
- Shown above is the Global Enable/Disable for SMI#

# SMI\_EN

**APMC\_EN** — R/W.

0 = Disable. Writes to the APM\_CNT register will not cause an SMI#.

1 = Enables writes to the APM\_CNT register to cause an SMI#.

**SLP\_SMI\_EN** — R/W.

0 = Disables the generation of SMI# on SLP\_EN. Note that this bit must be 0 before the software attempts to transition the system into a sleep state by writing a 1 to the SLP\_EN bit.

1 = A write of 1 to the SLP\_EN bit (bit 13 in PM1\_CNT register) will generate an SMI#, and the system will not transition to the sleep state based on that write to the SLP\_EN bit.

**LEGACY\_USB\_EN** — R/W.

0 = Disable.

1 = Enables legacy USB circuit to cause SMI#.

**BIOS\_EN** — R/W.

0 = Disable.

1 = Enables the generation of SMI# when ACPI software writes a 1 to the GBL\_RLS bit (D31:F0:PMBase + 04h:bit 2). Note that if the BIOS\_STS bit (D31:F0:PMBase + 34h:bit 2), which gets set when software writes 1 to GBL\_RLS bit, is already a 1 at the time that BIOS\_EN becomes 1, an SMI# will be generated when BIOS\_EN gets set.

- We can disable the generation of SMI# on writes to IO port 0xB2
- And a slough of others (BIOS\_EN refers to BIOS being able to receive ACPI “messages”, it has nothing to do with enabling/disabling BIOS itself)

# SMI\_EN

27	<b>GPIO_UNLOCK_SMI_EN</b> — R/W/O. Setting this bit will cause the Intel ICH9 to generate an SMI# when the GPIO_UNLOCK_SMI_STS bit is set in the SMI_STS register. Once written to '1', this bit can only be cleared by PLTRST#.
26:19	Reserved
18	<b>INTEL_USB2_EN</b> — R/W. 0 = Disable 1 = Enables Intel-Specific USB2 SMI logic to cause SMI#.
17	<b>LEGACY_USB2_EN</b> — R/W. 0 = Disable 1 = Enables legacy USB2 logic to cause SMI#.
16:15	Reserved
14	<b>PERIODIC_EN</b> — R/W. 0 = Disable. 1 = Enables the ICH9 to generate an SMI# when the PERIODIC_STS bit (PMBASE + 34h, bit 14) is set in the SMI_STS register (PMBASE + 34h).
13	<b>TCO_EN</b> — R/W. 0 = Disables TCO logic generating an SMI#. Note that if the NMI2SMI_EN bit is set, SMIs that are caused by re-routed NMIs will not be gated by the TCO_EN bit. Even if the TCO_EN bit is 0, NMIs will still be routed to cause SMIs. 1 = Enables the TCO logic to generate SMI#.  <b>NOTE:</b> This bit cannot be written once the TCO_LOCK bit is set.
12	Reserved
11	<b>MCSMI_EN</b> Microcontroller SMI Enable ( <b>MCSMI_EN</b> ) — R/W. 0 = Disable. 1 = Enables ICH9 to trap accesses to the microcontroller range (62h or 66h) and generate an SMI#. Note that "trapped" cycles will be claimed by the ICH9 on PCI, but not forwarded to LPC.

- SMI\_EN provides a lot of control over the generation of SMI#
- It can also enable/disable that periodic generation of SMI#
- You get the idea...

# Demo: SMI Suppression

The screenshot shows a PCI configuration utility window titled "PCI" with a toolbar and a dropdown menu showing "Bus 00, Device 1F, Function 00 - Intel Corporation ISA Bridge". The main area is a grid of 16-bit registers (offsets 00 to F0) with their values. The value at offset 0A is 0A. A dialog box titled "PCI00,1F,00 Reg 0DC (220)" is open, showing a bit field editor for bits 7 through 0. Bit 0 is circled in red and has a value of 1. The hex value 0B is shown below the bit field. The text "BIOS\_CNTL register" is written to the right of the dialog box.

Offset	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	86	80	17	29	07	01	10	02	03	00	01	06	00	00	80	00
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00	00	00	00	00	28	10	33	02
30	00	00	00	00	E0	00	00	00	00	00	00	00	00	00	00	00
40	01	10	00	00	80	00	00	00	81	10	00	00	10	00	00	00
50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
60	83	8A	8B	8A	D1	00	00	00	8A	83	8B	80	F8	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
80	00	00	04	3C	01	09	7C	00	00	00	00	00	81	00	00	00
90	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
A0	20	0E	00	00	39	00	80	00	2B	1C	4A	00	00	00	00	00
B0	00	00	F0	00	00	00	00	00	08	00	01	00	00	00	00	00
C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
D0	00	00	00	00	00	00	00	00	80	F0	00	00	0A	00	00	00
E0	09	00	0C	10	00	02	C4	03	04	00	00	00	00	00	00	00
F0	01	80	D1	FE	00	00	00	00	86	0F	03	00	00	00	00	00

- As we know, we (should be) unable to assert bit 0 in the BIOS\_CNTL register located in LPC D31:F0, offset DCh
- Let's "fix" that!

# Demo: SMI Suppression

## PMBASE—ACPI Base Address Register (LPC I/F—D31:F0)

Offset Address:	40h–43h	Attribute:	R/W, RO
Default Value:	00000001h	Size:	32 bit
Lockable:	No	Usage:	ACPI, Legacy
		Power Well:	Core

Sets base address for ACPI I/O registers, GPIO registers and TCO I/O registers. These registers can be mapped anywhere in the 64-K I/O space on 128-byte boundaries.

Bit	Description
31:16	Reserved
15:7	<b>Base Address</b> — R/W. This field provides 128 bytes of I/O space for ACPI, GPIO, and TCO logic. This is placed on a 128-byte boundary.
6:1	Reserved
0	Resource Type Indicator (RTE) — RO. Hardwired to 1 to indicate I/O space.

- Locate the PMBASE address from LPC D31:F0, offset 40h
- This is mapped to the I/O address space, as indicated in the Base Address register description

# Demo: SMI Suppression

The image shows two BIOS/UEFI windows. The left window is titled 'PCI' and shows a table for 'Bus 00, Device 1F, Function 00 - Intel Corporation ISA Bridge'. The right window is titled 'ACPI Power Management IO Space' and shows a table of IO Space Base values. A dialog box titled 'IO Space Base' is open, showing 'IO Base' set to '1000' and the checkbox 'ACPI Power Management Base' checked. A red circle highlights the value '00001001' in the PCI table, and a red arrow points from this circle to the 'IO Base' field in the dialog box.

Offset	Value	Value	Value
64	03020100	07060504	0B0A0908
00	29178086	02100107	06010003
10	00000000	00000000	00000000
20	00000000	00000000	00000000
30	00000000	000000E0	00000000
40	00001001	00000000	00001081
50	00000000	00000000	00000000
60	8A8B8A83	000000D1	808B838A
70	00000000	00000000	00000000

Offset	Value							
0	0100	0302	0504	0706	0908	0B0A	0D0C	0F0E
00	0000	FF0D	FFFF	FF0D	FFFF	FF0D	FFFF	2082
10	BFCF	0000	0000	0000	0000	0000	0000	0000
20	FFFF							
30	FFFF							
40	FFFF							
50	FFFF							
60	FFFF							
70	FFFF							
80	FFFF							
90	FFFF							
A0	FFFF							

- In this case we can see it is mapped to I/O starting at address 0x1000
- Open up an I/O ports window and enter 0x1000
- Be sure to check ACPI Power Management Base
- On some systems not doing this causes lockups or system crashes

# Demo: SMI Suppression

## SMI\_EN—SMI Control and Enable Register

I/O Address:	PMBASE + 30h	Attribute:	R/W, R/WO, WO
Default Value:	00000000h	Size:	32 bit
Lockable:	No	Usage:	ACPI or Legacy
Power Well:	Core		

0	<b>GBL_SMI_EN</b> — R/W. 0 = No SMI# will be generated by ICH9. This bit is reset by a PCI reset event. 1 = Enables the generation of SMI# in the system upon any enabled SMI event. <b>NOTE:</b> When the SMI_LOCK bit is set, this bit cannot be changed.
---	--

ACPI Power Management IO Space

IO Space Base = 1000

48	03020100	07060504	0B0A0908	0F0E0D0C
00	FF0C0000	FF0CFFFF	FF0CFFFF	A0CAFFFF
10	0000FFFF	00000000	00000000	00000000
20	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFF0E0E
30	FFFFFFFF	FFFFFFFF	FFFFFFFF	FF00FFFF
40	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF

- The bit to suppress global SMI# is at bit 0 in the SMI\_EN register located at PMBASE + 30h
- It looks like uninitialized space, but everything is enabled
- Just not locked down

# Demo: SMI Suppression

ACPI Power Management IO Space

IO Space Base = 1000

48	03020100	07060504	0B0A0908	0F0E0D0C
00	FF0C0000	FF0CFFFF	FF0CFFFF	A0CAFFFF
10	0000FFFF	00000000	00000000	00000000
20	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFF0E0E
30	FFFFFFFF	FFFFFFFF	FFFFFFFF	FF00FFFF
40	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF

IO Space 1030 (48)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
FF								FF								FF								FE							

Done  
Cancel

- Commence SMI# suppression!
- De-assert bit 0 so that SMI\_EN is FFFF\_FF FEh

# Demo: SMI Suppression

The image shows a screenshot of a PCI configuration utility window. The main window displays a grid of configuration space for 'Bus 00, Device 1F, Function 00 - Intel Corporation ISA Bridge'. The grid has columns for offset (00-0F) and rows for address (00-F0). The value at offset 0A is highlighted with a red circle. An inset dialog box titled 'PCI00,1F,00 Reg 0DC (220)' is open, showing a bit field editor for bits 7 through 0. Bit 0 is circled in red and set to 1. The current register value is shown as 0B. The text 'BIOS\_CNTL register' is written to the right of the dialog box.

Offset	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	86	80	17	29	07	01	10	02	03	00	01	06	00	00	80	00
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00	00	00	00	00	28	10	33	02
30	00	00	00	00	E0	00	00	00	00	00	00	00	00	00	00	00
40	01	10	00	00	80	00	00	00	81	10	00	00	10	00	00	00
50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
60	83	8A	8B	8A	D1	00	00	00	8A	83	8B	80	F8	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
80	00	00	04	3C	01	09	7C	00	00	00	00	00	81	00	00	00
90	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
A0	20	0E	00	00	39	00	80	00	2B	1C	4A	00	00	00	00	00
B0	00	00	F0	00	00	00	00	00	08	00	01	00	00	00	00	00
C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
D0	00	00	00	00	00	00	00	00	80	F0	00	00	00	00	00	00
E0	09	00	0C	10	00	02	C4	03	04	00	00	00	00	00	00	00
F0	01	80	D1	FE	00	00	00	00	86	0F	03	00	00	00	00	00

- Now with SMI# suppressed we can enable writes to the BIOS flash by asserting bit 0 in the BIOS\_CNTL register
- We'll cover this in more detail in the next section

# Demo: SMI Suppression

PCI

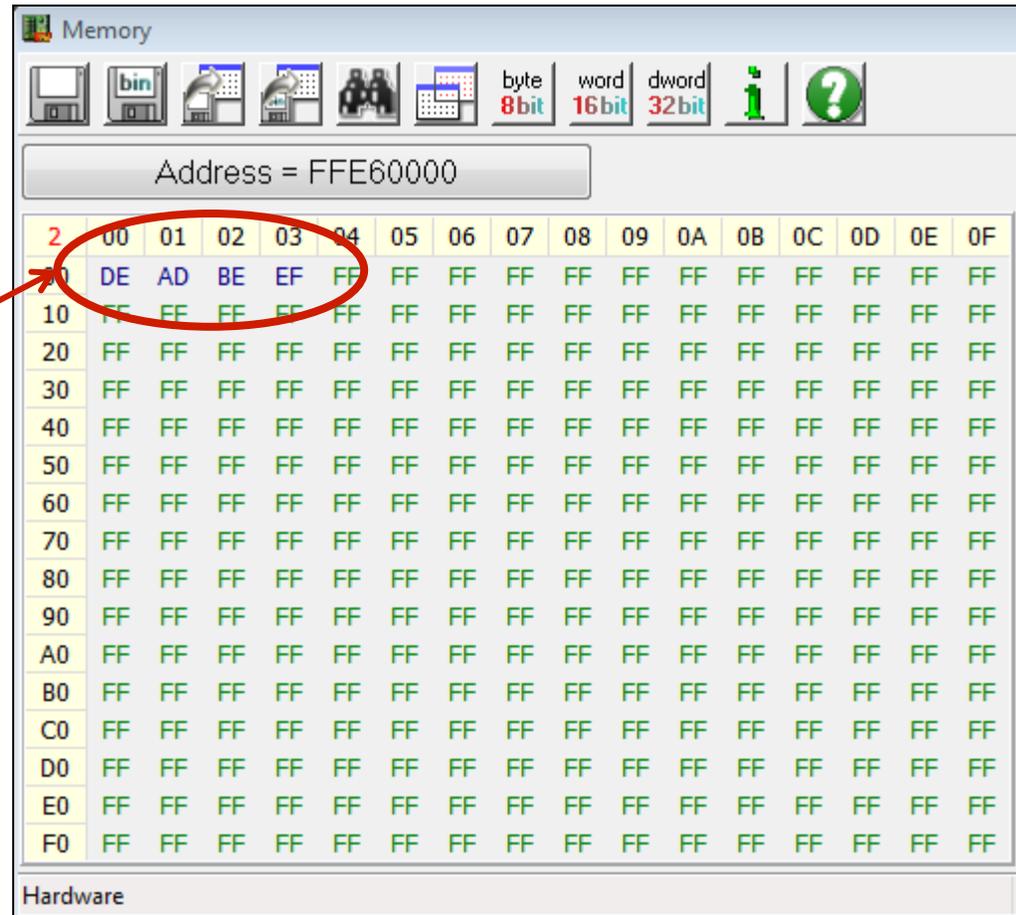
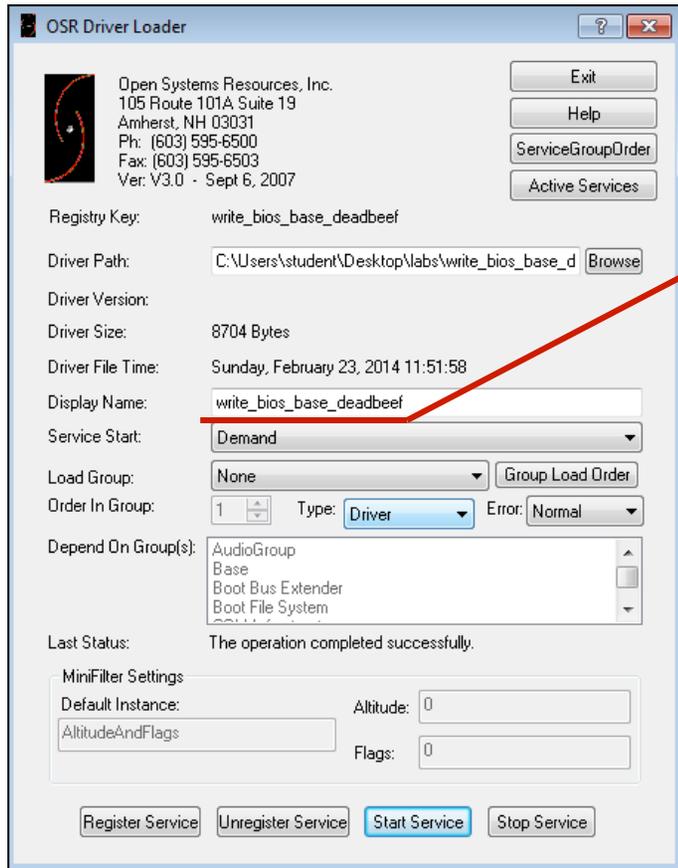
Bus 00, Device 1F, Function 00 - Intel Corporation ISA Bridge

220	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	86	80	17	29	07	01	10	02	03	00	01	06	00	00	80	00
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00	00	00	00	00	28	10	33	02
30	00	00	00	00	E0	00	00	00	00	00	00	00	00	00	00	00
40	01	10	00	00	80	00	00	00	81	10	00	00	10	00	00	00
50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
60	83	8A	8B	8A	D1	00	00	00	8A	83	8B	80	F8	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
80	00	00	04	3C	01	09	7C	00	00	00	00	00	81	0C	3C	00
90	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
A0	20	0E	00	00	39	00	80	00	2B	1C	4A	00	00	03	00	40
B0	00	00	F0	00	00	00	00	00	08	00	01	00	00	00	00	00
C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
D0	00	00	00	00	00	00	00	00	80	F0	00	00	0B	00	00	00
E0	09	00	0C	10	00	02	C4	03	04	00	00	00	00	00	00	00
F0	01	80	D1	FE	00	00	00	00	86	0F	03	00	00	00	00	00

Hardware

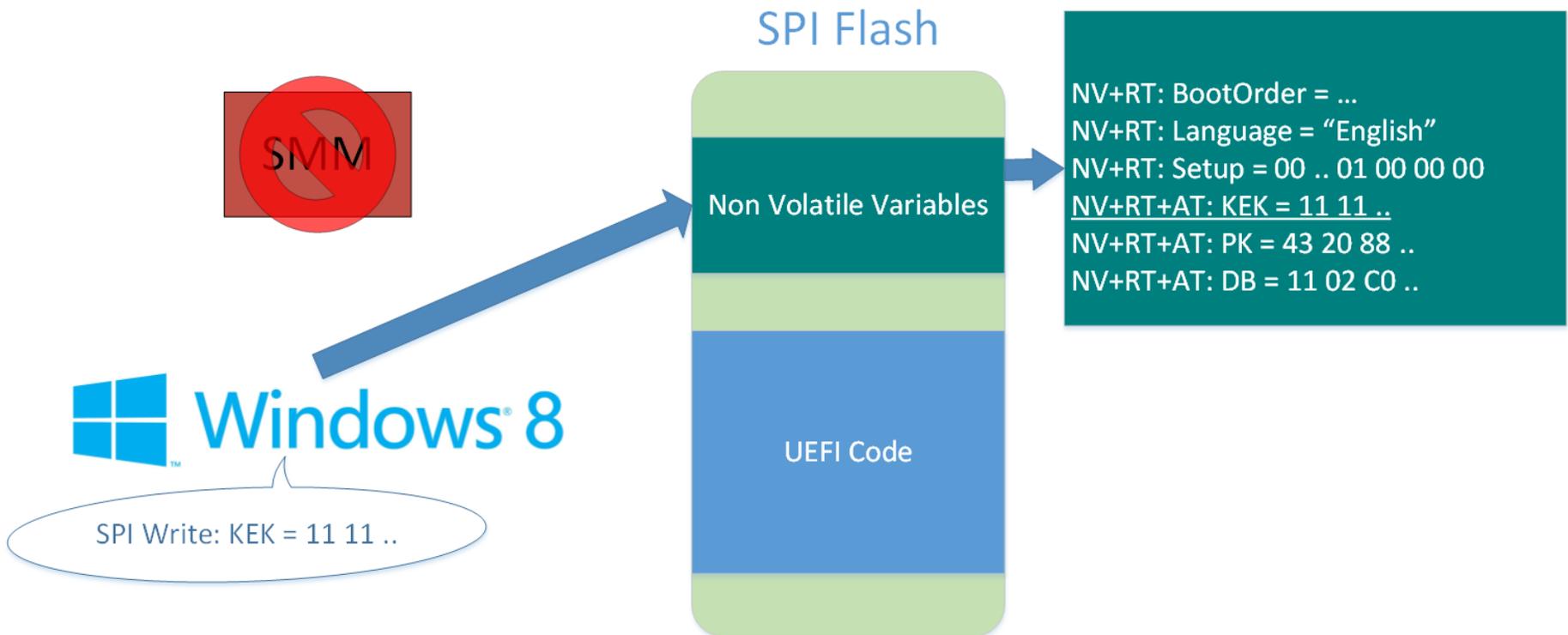
- Notice that bit 0 remains asserted now whereas before disabling SMI# it would have been reset to 0
- Now we can write to the BIOS. This is very bad.

# Demo: SMI Suppression



- Running the write\_bios\_base\_deadbeef.sys writes to the BIOS base to prove this point

# Demo Video: Charizard



- Ring0 can modify authenticated EFI Variables, which allows trivial bypassing of Secure Boot
  - We'll cover this in the UEFI secure boot portion of the class. For now just take my word for it: this is not good

# SMI Suppression

- As we'll see in the next section, there is one secondary defense that could still work to prevent an attacker from being able to flash the BIOS under these circumstances
  - However they can't be used to protect the UEFI variables because those must always be writeable
- Locking down the SMI\_EN register is something that vendors don't really know about:
- 3216 of 8005 (~40%) systems measured did not have SMI\_LOCK set
  - The numbers are much higher if you rollback the BIOS to a vulnerable revision, which is typically permitted

# SMI Suppression Prevention 1: GEN\_PMCON1

## GEN\_PMCON\_1—General PM Configuration 1 Register (PM—D31:F0)

Offset Address:	A0h	Attribute:	R/W, RO, R/WO
Default Value:	0000h	Size:	16-bit
Lockable:	No	Usage:	ACPI, Legacy
		Power Well:	Core

4	<b>SMI_LOCK</b> — R/WO. When this bit is set, writes to the GLB_SMI_EN bit (PMBASE + 30h, bit 0) will have no effect. Once the SMI_LOCK bit is set, writes of 0 to SMI_LOCK bit will have no effect (i.e., once set, this bit can only be cleared by PLTRST#).
---	--

- GEN\_PMCON1 is located in the LPC D31:F0 Power Management registers
- The vendor must (must must!) assert SMI\_LOCK in the GEN\_PMCON\_1 register
- Don't give attackers the option of suppressing SMI#
- Especially since the system depends on SMM to protect the BIOS Flash!!!

# SMI Suppression Prevention 2: BIOS\_CNTL.SMM\_BWP

## 13.1.32 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	<b>SMM BIOS Write Protect Disable (SMM_BWP)</b> — 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.

- We'll cover this register during the BIOS flash portion of the course
- Raises the security level of the platform
- Suffice to say this is a very useful (newish) feature
  - That isn't utilized very often