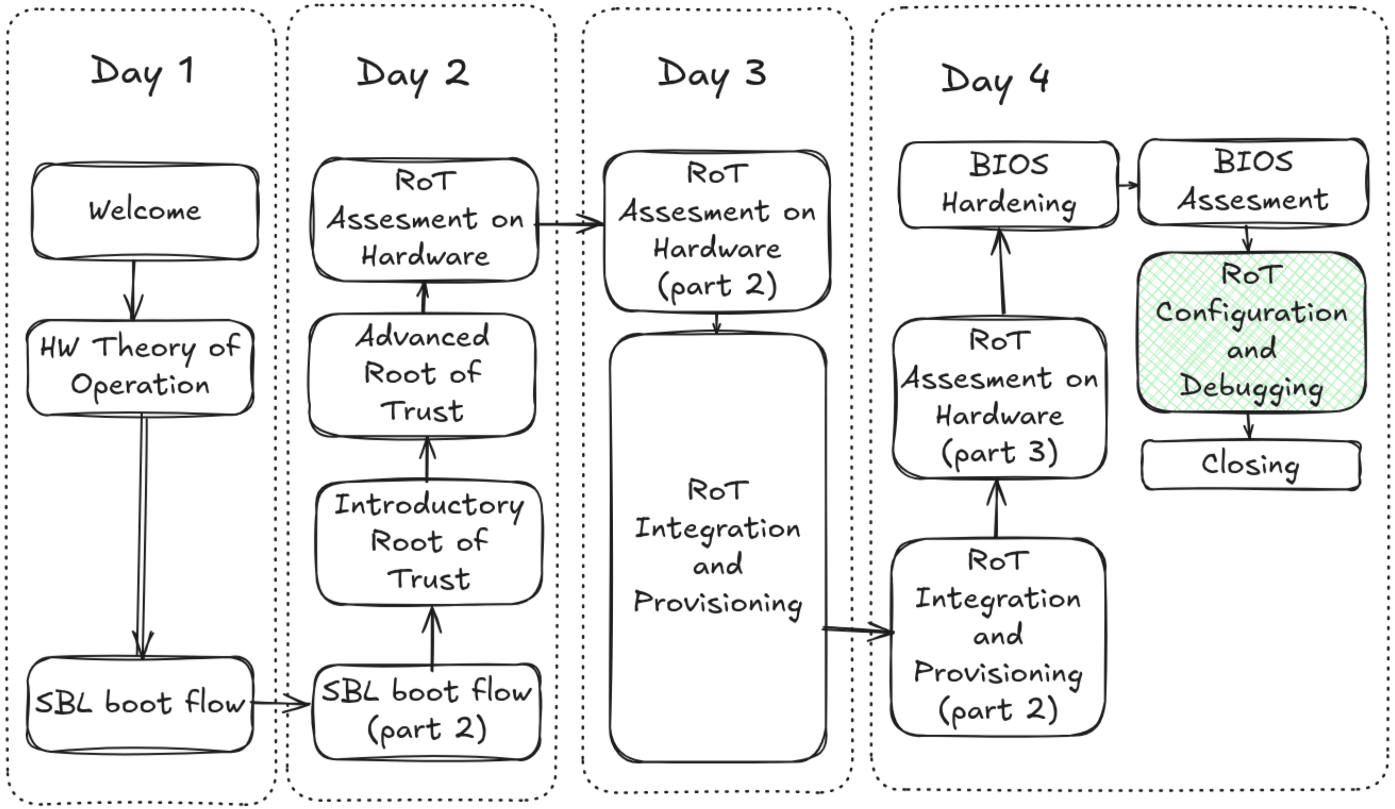
DS09SBL: Dasharo TrustRoot Training

Advanced Intel Boot Guard configuration and debugging

Dasharo Trust Root Training

Where we are in the course


Day 1
Welcome
HW Theory of Operation
SBL boot flow
Day 2
SBL boot flow (part 2)
Introductory Root of Trust
Advanced Root of Trust
RoT Assesment on Hardware
Day 3
RoT Assesment on Hardware (part 2)
RoT Integration and Provisioning
Day 4
RoT Integration and Provisioning (part 2)
RoT Assesment on Hardware (part 3)
BIOS Hardening
BIOS Assesment
RoT Configuration and Debugging
Closing

Goals of the Presentation

In this presentation, we aim to:

- Debug Intel Boot Guard errors
- Try various Boot Guard profiles and HAP to see how it affects Boot Guard behavior
- Debug platform with Intel System Bring-up Toolkit and hardware debuggers
- Prepare the platform for production shipment
    - Learn about End of Manufacturing process

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Boot Guard debugging

- Boot Guard exposes a set of registers holding various information about Boot Guard state and error codes which can be read by host.
- To read the registers, the platform must be able to boot though.
- The best profile for debugging is profile 3, unlimited recovery time and all features enabled (verified and measured boot)
    - However, it does not mean ME will always release the CPU from reset, depending on error severity.

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Boot Guard debugging tips

- Have MEInfo at hand in the operating system you will boot
- Have **Intel ME** `<version>` **BIOS Specification** at hand to decode ME FW status registers carrying a lot of Boot Guard status information (ME Info will print FW status registers but will not parse all fields)
- Sometimes ME is not available if Boot Guard fails so the CPU registers are the only source of knowledge what has gone wrong
- Unlock I/O and memory access by providing `iomem=relaxed` parameter to the kernel commandline when booting, we will need access to Boot Guard CPU registers. iotools are very useful for such operations, have them installed on the Linux OS
- Have ACM error list CSV file at hand from SINIT ACM package or from the BIOS ACM kit to decode errors
- Sometimes ME ends up in a limbo and does not release the CPU from reset, CMOS clear helps in such case

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

BOOT_GUARD_SACM_INFO_MSR (0x13A)

Bits	Description
63:36	Reserved
35	No Reset Secrets Protection (new in CBnT)
34	Server TXT Capability (new in CBnT)
33	Reserved
32	CPU Boot Guard capability, if set CPU is Boot Guard capable
31:8	Reserved
7	Module Revoked - at least one Boot Guard structure has been revoked
6	Verified Boot
5	Measured Boot
4	FACB (Force Anchor Cove Boot)
3	TPM success, 0 means a TPM failure occurred
2:1	TPM type: 0 - No TPM, 1 - TPM1.2 TIS, 2 - TPM 2.0 TIS, 3 - PTT (fTPM)
0	NEM Enabled, if set, Boot Guard has executed and successfully verified IBB

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

BOOT_STATUS (SPAD) (MMIO 0xFED300A0)

Bits	Description
63	S-ACM Startup Success - legacy bit, identical to bit 31
62	CPU error (Startup ACM authentication failed)
61	Boot Guard Startup error
60	Intel TXT disabled by policy
59	BIOS trusted
49:58	Reserved
48	Boot Guard failed
47	Memory power down executed
46:34	Reserved
33	PFR S-CRTM Startup success
32	Block Boot enabled
31	Boot Guard Startup success
30	Intel TXT Startup success
29:0	Reserved

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

ACM_STATUS (MMIO 0xFED30328, Boot Guard 1.0 only)

Bits	Description
31	Valid bit, if set this register contents are valid
30:25	Reserved
24	BP.RSTR.BBP - Boot Policy Restrictions BBP - verified boot enforcement
23	BP.RSTR.BPE - BIOS Protect Enable, means Boot Guard ACM copies IBB to cache
22	BP.RSTR.DBI - Disable BSP #INIT, means to block INIT signals to mitigate code refetch
21	BP.RSTR.DCD - Disable CPU debugging, if set DCI JTAG that would tamper with ACM execution is disabled
20	BP.RSTR.HAP - High Assurance Platform, the bit that is used to instruct ME to halt right after paltform initialization
19:16	Key Manifest ID
15	ACM started, 1 means it has started
14:10	ACM major error code
9:4	ACM class code
3:0	Module Type, the value of 3 means Boot Guard ACM

ACM_STATUS (MMIO 0xFED30328, Boot Guard 1.0 only)

Bits	Description
14:10	ACM major error code
9:4	ACM class code

Note

Errors can be decoded using the Table 5-4 in Boot Guard BIOS Specification (doc 557867, SKL/KBL only) or the error list CSV file included with the ACMs package (recommended, if available).

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

ACM_STATUS (MMIO 0xFED30328, CBnT only)

Bits	Description
31	Valid bit, if set this register contents are valid
30:28	Reserved
27:16	ACM minor error code
15	ACM started, 1 means it has started
14:10	ACM major error code
9:4	ACM class code
3:0	Module Type, the value of 3 means Boot Guard ACM

Note

Errors can be decoded using the error list CSV file included with the ACMs package or an older version of TXT Software Development Guide. It gives more or less accurate information what went wrong and what has to be fixed in the firmware or provisioning process.

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

ACM_POLICY_STATUS (MMIO 0xFED30378, CBnT only)

Bits	Description
15	TPM Success - indicates if ACM successfully executed all TPM commands
14:13	TPM type: 0 - No TPM, 1 - dTPM1.2 TIS, 2 - TPM 2.0 TIS, 3 - PTT (fTPM)
12	Reserved
11	BP.RSTR.PBE - Protect BIOS Environment, means Boot Guard ACM copies IBB to cache
10	BP.RSTR.DBI - Disable BSP #INIT, means to block INIT signals to mitigate code refetch
9	BP.RSTR.DCD - Disable CPU debugging, if set DCI JTAG that would tamper with ACM execution is disabled
8	Reserved
7	BP.TYPE.T - TXT Supported
6	BP.TYPE.HAP - High Assurance Platform, the bit that is used to instruct ME to halt right after platform initialization
5	BP.TYPE.V - Verified Boot, Boot Guard ACM verifies IBB
4	BP.TYPE.M - Measured Boot, if set Boot Guard ACM measures IBB to TPM
3:0	Key Manifest ID

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

ACM_POLICY_STATUS (MMIO 0xFED30378, CBnT only) continued

Bits	Description
63:37	Reserved
36	TPM Startup Locality: 0 - Locality 3, 1 - Locality 0
35	CPU Co-Signing
34:32	S-CRTM Status, S-CRTM established by: 0 - None, 1 - BTG, 2 - TXT, 3 - PFR
31:30	Reserved
29	IBB DMA protection
28:27	Reserved
26:25	Memory scrubbing policy
24:20	Intel TXT profile selection
19:18	Backup action: 0 - Memory power down, 1 - BTG Unbreakable Shutdown, 2 - PFR Recovery, 3 - Reserved
17	BP_KEYTYPE.P - Indicates PFR supported
16	Reserved

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Practice 309 Exercise #1

Decoding error codes in ACM Status register.

1. Build Slim Bootloader debug image:

```
user@OST2-VM:~$ DEBUG=1 ./build.sh odroid_h4
```

2. Provision it for Boot Guard:

```
user@OST2-VM:~$ python Platform/AlderlakeBoardPkg/Script/StitchIfwi.py \
    -b vm \
    -p adln \
    -w ~/training_materials/src/slimbootloader \
    -s Outputs/odroid_h4/SlimBootloader.bin \
    -c Platform/AlderlakeBoardPkg/Script/StitchIfwiConfig_adln.py \
    -d 0xAAFFFF0C \
    -k SblTestKeys/ \
    -o 16MB;
# Copy the descriptor to the final image so that straps are not modified
user@OST2-VM:~$ echo -ne '\x00' | dd of=sbl_ifwi_adln.bin bs=1 seek=$((0x00fffff6)) conv=notrunc
user@OST2-VM:~$ dd if=Outputs/odroid_h4/descriptor.bin of=sbl_ifwi_adln.bin conv=notrunc
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

3. Corrupt both top swap regions to cause Boot Guard failure.

```
user@OST2-VM:~$ echo -ne '\x00' | dd of=sbl_ifwi_adln.bin bs=1 seek=$((0x00fffff6)) conv=notrunc
user@OST2-VM:~$ echo -ne '\x00' | dd of=sbl_ifwi_adln.bin bs=1 seek=$((0x00f7fff4)) conv=notrunc
```

4. Transfer the image onto USB stick with DTS.
5. Plug the stick to the ODROID H4 platform and boot with vendor BIOS image to DTS shell.
6. Switch the BIOS jumper and flash the whole image.
7. Gather the serial output to a file and power cycle the platform:

```
user@OST2-VM:~$  minicom -D /dev/ttyUSB0 -C /tmp/output.log
```

8. Get the `AcmStatus`, `BootStatus` and `Acm Info` register values and decode them with CBnT BIOS Writers guide (doc 575623) or using previous slides.
9. Decode the error class, major and minor code with the CSV included in the ACM package.

ACM error decoding

```
user@OST2-VM:~$ cat /tmp/output.log | grep "Boot Guard"
[Boot Guard] AcmStatus : 0×C0039910
[Boot Guard] BootStatus: 0×10410000
[Boot Guard] Boot Guard Failed or is Disabled!
[Boot Guard] Acm Info: 0×730000006E
[Boot Guard] Verified Boot Status: Enabled
[Boot Guard] Measured Boot Status: Enabled
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

ACM error decoding - Boot Status

BootStatus: 0x10410000

- Bit 62 clear - CPU error did not occur
- Bit 61 clear - Boot Guard Startup error did not occur
- Bit 60 set - Intel TXT disabled by policy
- Bit 54 set - undocumented
- Bit 48 set - Boot Guard failed

Note

The 32bit value in Slim Bootloader logs should be shifted left by 32 bits (only the upper 32bit part was read and logged).

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

ACM error decoding - ACM Status

AcmStatus : 0xC0039910

- Bit 31 set - Register has valid content
- Bit 30 set - Error induced from external software (based on TXT ErrorCode Register)
- Bits 26:16 - the value of ACM Minor Error Code is `0x3`
- Bit 15 set - ACM has started
- Bits 14:10 - the value of ACM Major Error Code is `0x6`
- Bits 9:4 - the value of ACM Class Code is `0x11`
- Bits 3:0 - error code originates from BIOS ACM (`0x0`)

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

ACM error decoding

`ADL_RPL_StartupACM_1.18.19_SINITACM_1.18.18/Tools/Errors.csv`

Based on the errors CSV file:

- ACM Class Code is `0x11` : ~~`CLASS_STM_CHECK`~~ or `CLASS_BPT_INTEGRITY`
- ACM Major Error is `0x6` : ~~`ERR_STM_IS_REQUIRED`~~ or `ERR_IBB`
- ACM Minor Code is `0x3` : ~~not provided~~ or `ERR_BPM_SE_IBB_DGST_VALUE_INVALID`

Note

We don’t use SMI Transfer Monitor (STM) here, so we should look at the latter class code.

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Provisioning a different profile and enabling HAP

1. Provision Slim Bootloader for Boot Guard profile 5 (`fvme`):

```
user@OST2-VM:~$ python Platform/AlderlakeBoardPkg/Script/StitchIfwi.py \
    -b fvme \
    -p adln \
    -w ~/training_materials/src/slimbootloader \
    -s Outputs/odroid_h4/SlimBootloader.bin \
    -c Platform/AlderlakeBoardPkg/Script/StitchIfwiConfig_adln.py \
    -d 0×AAFFFF0C \
    -k SblTestKeys/ \
    -o 16MB;
# Copy the descriptor to the final image so that straps are not modified
user@OST2-VM:~$ dd if=Outputs/odroid_h4/descriptor.bin of=sbl_ifwi_adln.bin conv=notrunc
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

2. Enable HAP with `mFIT` (disabling PD is needed to make the command pass):

```
./Fit/mfit --decompose sbl_ifwi_adln.bin --setvalues \
"DescriptorPlugin:PchStraps:PCH_Strap_CSME_CSE_Reserved_Softstrap_16=Yes;\
DescriptorPlugin:PmcStraps:PMC_Strap_pmc_smip_PD0_Type_C_Port_Enabled=No;\
DescriptorPlugin:PmcStraps:PMC_Strap_pmc_smip_PD1_Type_C_Port_Enabled=No" \
--build sbl_ifwi_adln_hap.bin
```

3. Transfer the `sbl_ifwi_adln_hap.bin` onto USB stick with DTS.
4. Plug the stick to the ODROID H4 platform and boot with vendor BIOS image to DTS shell.
5. Switch the BIOS jumper and flash the whole image.
6. Gather the serial output to a file and power cycle the platform:

```
user@OST2-VM:~$  minicom -D /dev/ttyUSB0 -C /tmp/output.log
```

7. Check Boot Guard state in the log:

```
user@OST2-VM:~$ cat /tmp/output.log |grep "Boot Guard"
```

Boot Guard vs HAP

```
user@OST2-VM:~$ cat /tmp/output.log |grep -iE "Boot Guard|tpm"
[Boot Guard] AcmStatus : 0×C0008000
[Boot Guard] BootStatus: 0×98400000
[Boot Guard] Boot Guard is Enabled Successfully.
[Boot Guard] Acm Info: 0×730000007F
[Boot Guard] Verified Boot Status: Enabled
[Boot Guard] Measured Boot Status: Enabled
Boot Guard ACM Status = C0008000
Boot Guard Boot Status = 9840000080000000
TPM Type is 3
Boot Guard Support status: 1
Unable to talk to TPM !!
TPM Lib Private Data not found
Tpm Initialization failed  Device Error !!
```

HAP causes the ME to halt early. Result? fTPM becomes unavailable.

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Boot Guard vs HAP

Acm Info: 0x730000007F

- Bit 6 set - Boot Guard Verified Boot active
- Bit 5 set - Boot Guard Measured Boot active
- Bit 4 force - FACB (Force Anchor Cove Boot) active
- Bit 3 set - TPM success
- Bits 2:1 both set - PTT TPM present
- Bit 0 set - NEM enabled, Boot Guard verified IBB configured Cache as RAM/Non Evict Mode

Note

Despite fTPM is not available in Slim Bootloader, it seems to be still available during ACM execution. So HAP halts the ME after Boot Guard related actions are finished.

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Practice 309 Exercise #3

Boot Guard behavior in production.

1. Provision Slim Bootloader for Boot Guard profile 5 (`fvme`):

```
user@OST2-VM:~$ python Platform/AlderlakeBoardPkg/Script/StitchIfwi.py \
    -b fvme \
    -p adln \
    -w ~/training_materials/src/slimbootloader \
    -s Outputs/odroid_h4/SlimBootloader.bin \
    -c Platform/AlderlakeBoardPkg/Script/StitchIfwiConfig_adln.py \
    -d 0xAAFFFF0C \
    -k SblTestKeys/ \
    -o 16MB;
# Copy the descriptor to the final image so that straps are not modified
user@OST2-VM:~$ dd if=Outputs/odroid_h4/descriptor.bin of=sbl_ifwi_adln.bin conv=notru
```

Practice 309 Exercise #3

3. Corrupt both top swap regions to cause Boot Guard failure.

```
user@OST2-VM:~$ echo -ne '\x00' | dd of=sbl_ifwi_adln.bin bs=1 seek=$((0x00fffff6)) conv=notrunc
user@OST2-VM:~$ echo -ne '\x00' | dd of=sbl_ifwi_adln.bin bs=1 seek=$((0x00f7fff4)) conv=notrunc
```

4. Transfer the image onto USB stick with DTS.
5. Plug the stick to the ODROID H4 platform and boot with vendor BIOS image to DTS shell.
6. Switch the BIOS jumper and flash the whole image.
7. Gather the serial output to a file and power cycle the platform:

```
user@OST2-VM:~$  minicom -D /dev/ttyUSB0 -C /tmp/output.log
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Practice 309 Exercise #3

Results?

- Nothing on the serial console
- The status leds on ODROID-H4 are lit for a brief moment after pressing power button and then go off
- **Boot Guard performs unbreakable shutdown due to IBB verification failure**

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Debugging with Intel DCI

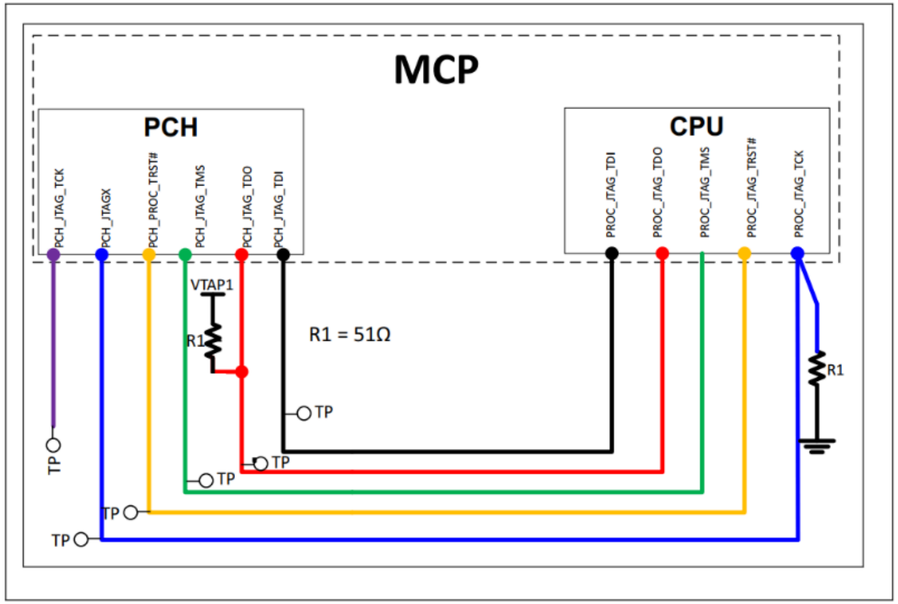
DCI - Direct Connect Interface

- Can be used for closed chassis debugging
- Re-uses USB-A/USB-C receptacles for the connection
- Methods of connection:
    - USB 2.0 DbC (Debug Class) interface
    - USB 3.x DbC (Debug Class) interface
    - DCI Out Of Band (OOB) aka BSSB - 2-wire or 4-wire interface
- DCI OOB requires a special adapter known as Intel SVT CCA (Silicon View Technology Closed Chassis Adapter)

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Intel DCI hardware requirements

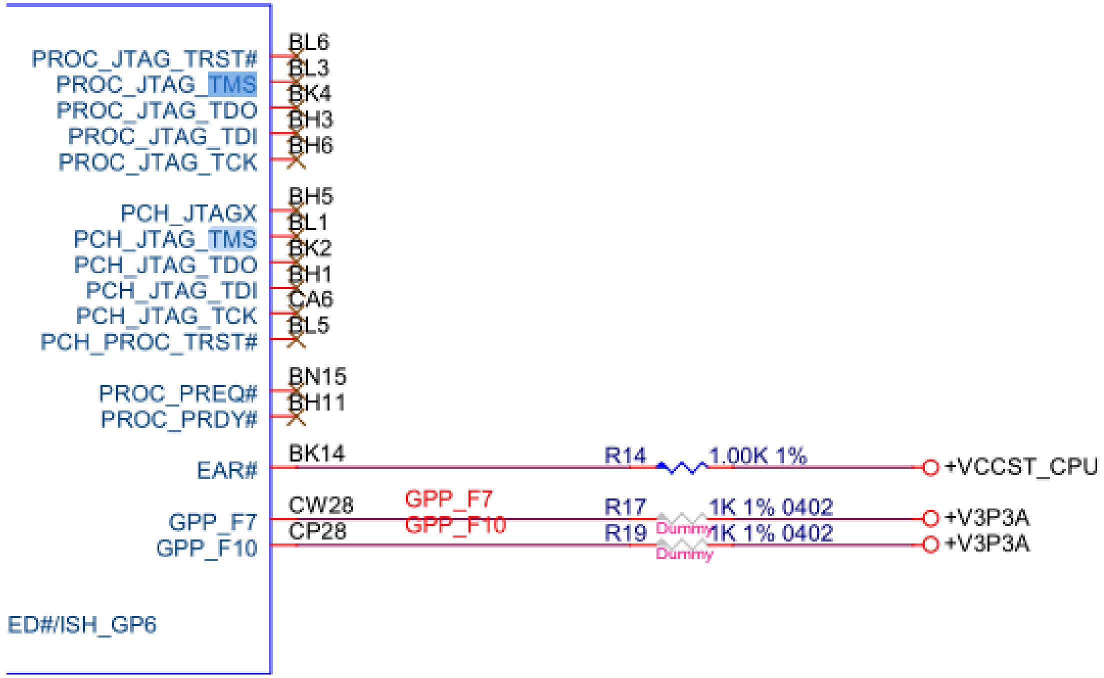
- PCH USB 3.x port type-A receptacle exposed
- JTAG signals connected between CPU and PCH:



- **Intel® DCI.OOB Configurations Supporting USB3 Type A**

Alder Lake N and Twin Lake Platform Design Guide (doc 646929)

Intel DCI hardware requirements



ODROID H4 schematics

Intel DCI firmware requirements

FSPM-M UPDs:

```
FspmConfig->PlatformDebugConsent = 2; // all probes
FspmConfig->JtagC10PowerGateDisable = 0;
FspmConfig->DciEn = 1;
FspmConfig->DciDbcMode = 4; // no change
FspmConfig->DciModphyPg = 0;
FspmConfig->DciUsb3TypecUfpDbg = 2; //no change
FspmConfig->PchTraceHubMode = 2; // host debugger
FspmConfig->PchTraceHubMemReg0Size = 2; // 8MB
FspmConfig->PchTraceHubMemReg1Size = 2; // 8MB
FspmConfig->CpuTraceHubMode = 2; // host debugger
FspmConfig->CpuTraceHubMemReg0Size = 2; // 8MB
FspmConfig->CpuTraceHubMemReg1Size = 2; // 8MB
FspmConfig->DebugInterfaceEnable = 1;
FspmConfig->DebugInterfaceLockEnable = 1;
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Intel DCI firmware requirements

ME configuration in XML:

- `Intel(R) DCI DbC Interface Enabled=Yes`
- `DCI OOB over USB3 PortN Enabled=Yes` for each PCH USB 3.x port N exposed on the chassis
- `PMC Hub Debug Messages Enabled=Yes`
- `USB2 DbC port enable` - set to port for exclusive USB2 DbC functionality (optional, it disables normal functionality of USB 2.0 data lines for the sole purpose of debugging), or leave None
- `USB Connectors Associated USB3 Port enable` - set to port paired with USB2 DbC functionality configured in `USB2 DbC port enable` (optional, it disables USB 3.x data lines completely for the USB2 DbC) or leave None
- `Enable early USB2 DbC connection` - set to `Yes` if using the exclusive USB2 DbC port with `USB2 DbC port enable` (optional). It is supposed to let ME initialize debug session very early over USB2 DbC.
- `CPU Debugging=Enable`

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Additional documentation

- Intel Platform Closed Chassis Debug User Guide (doc 626623)
- Intel Direct Connect Interface (Intel DCI) Enabling Guidance for OEMs Technical White Paper (doc 646201)
- Open and Closed Chassis Debug User Guide (doc 730814)
- Platform Design Guide for given microarchitecture
- Intel Debug Technology (doc 660257, public)
- Intel System Bring-up Toolkit
- Enabling Platform Development Tools with Intel System Bring-up Toolkit NDA User Guide (doc 626306)

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Intel System Bring-up Toolkit

- Set of tools, utilities and debugging software for use with Intel silicon and Intel hardware debuggers
- Required to setup JTAG connection over USB DbC or SVT CCA adapter
- Contains necessary drivers for the OS and Eclipse-based IDE
- How to Obtain Intel System Bring-up Toolkit NDA (doc 616675)
- Toolkit download center: https://lemcenter.intel.com/productDownload/

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Debugging an Intel platform using USB DbC and DCI OOB method.

1. Install Intel System Bring-up Toolkit.
2. Consult Documentation section of Intel System Bring-up Toolkit how to get started.
3. Prepare the firmware for target platform with Boot Guard and DCI required settings configured properly.
4. Flash the firmware and confirm Boot Guard works.
5. Connect the Intel SVT CCA adapter to the board's USB port configured for DCI.
6. Launch `ipccli` shell from Intel System Bring-up Toolkit:

```
user@OST2-VM:~$ cd $HOME/inteloneapi/system_debugger/latest
user@OST2-VM:~$ source isd_env.sh
user@OST2-VM:~$ ipccli
```

6. Verify the connection is established with both PCH and CPU.
7. List the detected CPU cores and the halt them:

```
ipccli> ipc.status()
ipccli> ipc.halt()
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Practice #309 Exercise #4

8. Resume the CPUs and quit:

```
ipccli> ipc.go()
ipccli> quit
```

9. Switch from the SVT CCA adapter to USB 3.x A to A cable (isolate Vbus pin on one end of the cable beforehand).
10. Reconnect to the platform by reopening the `ipccli`. Confirm the CPUs can be stopped:

```
user@OST2-VM:~$ ipccli
ipccli> ipc.status()
ipccli> ipc.halt()
```

Intel SVT CCA:

```
$ source isd_env.sh
Use local libstdc++.
Sourcing $HOME/inteloneapi/system_debugger/latest/env.d/linux/10-debugfoundation-env.sh
Sourcing $HOME/inteloneapi/system_debugger/latest/env.d/linux/10-python.sh
Sourcing $HOME/inteloneapi/system_debugger/latest/env.d/linux/70-tca.sh
Sourcing $HOME/inteloneapi/system_debugger/latest/env.d/linux/70-trace.sh
Sourcing $HOME/inteloneapi/system_debugger/latest/env.d/linux/85-isysdbg-env.sh
Sourcing $HOME/inteloneapi/system_debugger/latest/env.d/linux/95-gdbserverproxy-env.sh
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Intel SVT CCA:

```
$ ipccli
Python 3.10.14 | packaged by conda-forge | (main, Mar 20 2024, 12:45:18) [GCC 12.3.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.25.0 -- An enhanced Interactive Python. Type '?' for help.
Connecting to IPC API....
IPC-CLI: 4.24.22202.100, OpenIPC:ef5e09a13988754fdee0f88 : 24.22.13640.200
Initializing IPC API....

Info - Detecting remote probes.
Info - OpenIPC using automatic configuration

Info - DCI: A DCI device has been detected, attempting to establish connection
Info - DCI: Read FPGA Version: 0x8002012a
Info - DCI: CCA FPGA image is already up to date
Info - DCI: Sending BSSB connect pattern...
Info - DCI: Target connection has been fully established
Info - DCI: Connected to Target's USB Port 0
Info - Added debug port 0 for probe CCA Host USB PortPath 3-3
Info - Detected JSP A1 on JTAG chain 0 at position 0
Info - Detected JSL_CLTAP A0 on JTAG chain 1 at position 0
Info - Survivability Preset was requested, but no supported devices were detected.
  Preset will be applied when a supported device is detected.
TargetEvent: Power : On                 -- 14:39:44.407700 2025-08-21
TargetEvent: PowerDomain : PCH : On     -- 14:39:44.408083 2025-08-21
TargetEvent: PowerDomain : CPU : On     -- 14:39:44.408390 2025-08-21
In [1]:
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Intel SVT CCA:

```
$ ipccli
Python 3.10.14 | packaged by conda-forge | (main, Mar 20 2024, 12:45:18) [GCC 12.3.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.25.0 -- An enhanced Interactive Python. Type '?' for help.
Connecting to IPC API....
IPC-CLI: 4.24.22202.100, OpenIPC:ef5e09a13988754fdee0f88 : 24.22.13640.200
Initializing IPC API....

Info - Detecting remote probes.
Info - OpenIPC using automatic configuration

Info - DCI: A DCI device has been detected, attempting to establish connection
Info - DCI: Read FPGA Version: 0×8002012a
Info - DCI: CCA FPGA image is already up to date
Info - DCI: Sending BSSB connect pattern...
Info - DCI: Target connection has been fully established
Info - DCI: Connected to Target's USB Port 0
Info - Added debug port 0 for probe CCA Host USB PortPath 3-3
Info - Detected JSP A1 on JTAG chain 0 at position 0
Info - Detected JSL_CLTAP A0 on JTAG chain 1 at position 0
Info - Survivability Preset was requested, but no supported devices were detected.
  Preset will be applied when a supported device is detected.
TargetEvent: Power : On                   -- 14:39:44.407700 2025-08-21
TargetEvent: PowerDomain : PCH : On       -- 14:39:44.408083 2025-08-21
TargetEvent: PowerDomain : CPU : On       -- 14:39:44.408390 2025-08-21
In [1]:
```

Intel SVT CCA:

```
$ ipccli
Python 3.10.14 | packaged by conda-forge | (main, Mar 20 2024, 12:45:18) [GCC 12.3.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.25.0 -- An enhanced Interactive Python. Type '?' for help.
Connecting to IPC API....
IPC-CLI: 4.24.22202.100, OpenIPC:ef5e09a13988754fdee0f88 : 24.22.13640.200
Initializing IPC API....

Info - Detecting remote probes.
Info - OpenIPC using automatic configuration

Info - DCI: A DCI device has been detected, attempting to establish connection
Info - DCI: Read FPGA Version: 0x8002012a
Info - DCI: CCA FPGA image is already up to date
Info - DCI: Sending BSSB connect pattern...
Info - DCI: Target connection has been fully established
Info - DCI: Connected to Target's USB Port 0
Info - Added debug port 0 for probe CCA Host USB PortPath 3-3
Info - Detected JSP A1 on JTAG chain 0 at position 0
Info - Detected JSL_CLTAP A0 on JTAG chain 1 at position 0
Info - Survivability Preset was requested, but no supported devices were detected.
  Preset will be applied when a supported device is detected.
TargetEvent: Power : On                  -- 14:39:44.407700 2025-08-21
TargetEvent: PowerDomain : PCH : On      -- 14:39:44.408083 2025-08-21
TargetEvent: PowerDomain : CPU : On      -- 14:39:44.408390 2025-08-21
In [1]:
```

Intel SVT CCA:

```
In [1]: ipc.status()
Status for         : JSL_TNT0_C0_T0
  Processor        : Running
  Processor mode : Unavailable while Running
Status for         : JSL_TNT0_C1_T0
  Processor        : Running
  Processor mode : Unavailable while Running
Status for         : JSL_TNT0_C2_T0
  Processor        : Running
  Processor mode : Unavailable while Running
Status for         : JSL_TNT0_C3_T0
  Processor        : Running
  Processor mode : Unavailable while Running

In [2]: ipc.halt()

    [JSL_TNT0_C0_T0] HLT Instruction Break at [0x38:0000000077bef131] -- 14:27:33.033882 2025-08-21
    [JSL_TNT0_C1_T0] HLT Instruction Break at [0x38:0000000077bef131] -- 14:27:33.034647 2025-08-21
    [JSL_TNT0_C2_T0] HLT Instruction Break at [0x38:0000000077bef131] -- 14:27:33.034914 2025-08-21
    [JSL_TNT0_C3_T0] HLT Instruction Break at [0x38:0000000077bef131] -- 14:27:33.035144 2025-08-21
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Intel SVT CCA:

```
In [7]: ipc.go()



    [JSL_TNT0_C0_T0] Resuming -- 14:27:52.726647 2025-08-21
    [JSL_TNT0_C1_T0] Resuming -- 14:27:52.727925 2025-08-21
    [JSL_TNT0_C2_T0] Resuming -- 14:27:52.728600 2025-08-21
    [JSL_TNT0_C3_T0] Resuming -- 14:27:52.729222 2025-08-21
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

USB DbC:

```
$ ipccli
Python 3.10.14 | packaged by conda-forge | (main, Mar 20 2024, 12:45:18) [GCC 12.3.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.25.0 -- An enhanced Interactive Python. Type '?' for help.
Connecting to IPC API....
IPC-CLI: 4.24.22202.100, OpenIPC:ef5e09a13988754fdee0f88 : 24.22.13640.200
Initializing IPC API....

Info - Detecting remote probes.
Info - OpenIPC using automatic configuration

ReconfigEvent: Started             -- 14:37:43.775105 2025-08-21
Info - DCI: A DCI device has been detected, attempting to establish connection
Info - DCI: Target connection has been fully established
Info - DCI: Connected to Target's USB Port 9
Warning - DCI: Jtag.TclkRate configured = 7500000, actual = 6250000 for device 0x00012001 (CPU)
Info - Added debug port 0 for probe DirectConnect Host USB PortPath 4-3
Info - Detected JSP A1 on JTAG chain 0 at position 0
Info - Detected JSL_CLTAP A0 on JTAG chain 1 at position 0
Info - Survivability Preset was requested, but no supported devices were detected.
  Preset will be applied when a supported device is detected.
TargetEvent: Power : On                 -- 14:37:44.046494 2025-08-21
TargetEvent: PowerDomain : PCH : On     -- 14:37:44.046698 2025-08-21
ReconfigEvent: Updating devicelist -- 14:37:44.046803 2025-08-21
ReconfigEvent: Complete            -- 14:37:44.046803 2025-08-21
TargetEvent: PowerDomain : CPU : On     -- 14:37:44.046841 2025-08-21
In [1]:
```

USB DbC:

```
$ ipccli
Python 3.10.14 | packaged by conda-forge | (main, Mar 20 2024, 12:45:18) [GCC 12.3.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.25.0 -- An enhanced Interactive Python. Type '?' for help.
Connecting to IPC API....
IPC-CLI: 4.24.22202.100, OpenIPC:ef5e09a13988754fdee0f88 : 24.22.13640.200
Initializing IPC API....

Info - Detecting remote probes.
Info - OpenIPC using automatic configuration

ReconfigEvent: Started             -- 14:37:43.775105 2025-08-21
Info - DCI: A DCI device has been detected, attempting to establish connection
Info - DCI: Target connection has been fully established
Info - DCI: Connected to Target's USB Port 9
Warning - DCI: Jtag.TclkRate configured = 7500000, actual = 6250000 for device 0x00012001 (CPU)
Info - Added debug port 0 for probe DirectConnect Host USB PortPath 4-3
Info - Detected JSP A1 on JTAG chain 0 at position 0
Info - Detected JSL_CLTAP A0 on JTAG chain 1 at position 0
Info - Survivability Preset was requested, but no supported devices were detected.
  Preset will be applied when a supported device is detected.
TargetEvent: Power : On                 -- 14:37:44.046494 2025-08-21
TargetEvent: PowerDomain : PCH : On     -- 14:37:44.046698 2025-08-21
ReconfigEvent: Updating devicelist -- 14:37:44.046803 2025-08-21
ReconfigEvent: Complete            -- 14:37:44.046803 2025-08-21
TargetEvent: PowerDomain : CPU : On     -- 14:37:44.046841 2025-08-21
In [1]:
```

USB DbC:

```
$ ipccli
Python 3.10.14 | packaged by conda-forge | (main, Mar 20 2024, 12:45:18) [GCC 12.3.0]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.25.0 -- An enhanced Interactive Python. Type '?' for help.
Connecting to IPC API....
IPC-CLI: 4.24.22202.100, OpenIPC:ef5e09a13988754fdee0f88 : 24.22.13640.200
Initializing IPC API....

Info - Detecting remote probes.
Info - OpenIPC using automatic configuration

ReconfigEvent: Started             -- 14:37:43.775105 2025-08-21
Info - DCI: A DCI device has been detected, attempting to establish connection
Info - DCI: Target connection has been fully established
Info - DCI: Connected to Target's USB Port 9
Warning - DCI: Jtag.TclkRate configured = 7500000, actual = 6250000 for device 0x00012001 (CPU)
Info - Added debug port 0 for probe DirectConnect Host USB PortPath 4-3
Info - Detected JSP A1 on JTAG chain 0 at position 0
Info - Detected JSL_CLTAP A0 on JTAG chain 1 at position 0
Info - Survivability Preset was requested, but no supported devices were detected.
  Preset will be applied when a supported device is detected.
TargetEvent: Power : On                 -- 14:37:44.046494 2025-08-21
TargetEvent: PowerDomain : PCH : On     -- 14:37:44.046698 2025-08-21
ReconfigEvent: Updating devicelist -- 14:37:44.046803 2025-08-21
ReconfigEvent: Complete            -- 14:37:44.046803 2025-08-21
TargetEvent: PowerDomain : CPU : On     -- 14:37:44.046841 2025-08-21
In [1]:
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Practice #309 Exercise #5

Prove the CPU Debugging Disable fuse prevent debugging an Intel platform using Intel DCI.

1. Use the firmware from the previous exercise and boot to DTS Shell.
2. Transfer Linux64 variant of Flash Programming Tool (FPT) from the ME toolkit to the booted Linux OS on target platform.
3. Disable CPU debug by altering the fuse (confirm it was enabled before the change and is disabled after change):

```
root@DTS# ./FPT -R "CPU Debugging"
root@DTS# ./FPT -U -N "CPU Debugging" -V Disabled
root@DTS# ./FPT -R "CPU Debugging"
```

4. Issue global reset to apply the change:

```
root@DTS# ./FPT -GRESET
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

5. Connect the Intel SVT CCA adapter to the board's USB port configured for DCI.
6. Launch `ipccli` shell from Intel System Bring-up Toolkit:

```
user@OST2-VM:~$ cd $HOME/inteloneapi/system_debugger/latest
user@OST2-VM:~$ source isd_env.sh
user@OST2-VM:~$ ipccli
```

6. Verify the connection is established with both PCH and CPU. List the detected CPU cores:

```
ipccli> ipc.status()
```

8. Attempt to halt the CPU cores:

```
ipccli> ipc.halt()
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Practice #309 Exercise #5

```
In [1]: ipc.status()
Status for         : JSL_TNT0_C0_T0
  Processor        : Running
  Processor mode : Unavailable while Running
Status for         : JSL_TNT0_C1_T0
  Processor        : Running
  Processor mode : Unavailable while Running
Status for         : JSL_TNT0_C2_T0
  Processor        : Running
  Processor mode : Unavailable while Running
Status for         : JSL_TNT0_C3_T0
  Processor        : Running
  Processor mode : Unavailable while Running
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Practice #309 Exercise #5

```
In [9]: ipc.halt()
──────────────────────────────────────────────────────────────────────────────────
IPC_Error                                  Traceback (most recent call last)
Cell In[9], line 1
────→ 1 ipc.halt()

File ~/inteloneapi/system_debugger/2426-nda/tools/python310/lib/python3.10/\
site-packages/ipccli/ipc_env/ipc_baseaccess.py:1291, in baseaccess.halt(self)
   1288 run_control_was_disabled = not gpc.isruncontrolenabled()
   1290 operation = runsrvc.HaltAll()
→ 1291 operation.Flush()
   1293 # If run control was disabled and the halt attempt succeeded, then
   1294 # update the supported breaks on devices
   1295 if run_control_was_disabled:
...

IPC_Error: RunControl_Timeout == 0×800a0004
. Timeout occurred while waiting for threads to halt with 0 retries.
...
```

Production settings

- Intel recommends configuring a bunch of settings for production/shipment
  - Some of them are being set as a part of End of Manufacturing flow
- We will run through the settings and ensure they are configured with security in mind:
  - Boot Guard policy
  - Debug capabilities
  - Flash protections
  - Fuse configuration

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Production settings

Boot Guard Policy

Boot Guard Profile	Old Name	New Name
0	Boot Guard Profile 0 - No_FVME	Boot Guard Disabled
1 (deprecated)	Boot Guard Profile 1 - VE	-
2 (deprecated)	Boot Guard Profile 2 - VME	-
3	Boot Guard Profile 3 - VM	Debug Profile
4	Boot Guard Profile 4 - FVE	Verified Boot
5	Boot Guard Profile 5 - FVME	Verified and Measured Boot

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Production settings

Debug capabilities and fuse configuration

- There are lots of debug settings in MFIT
- Investigate the XML under `<sbl_root>/Temp/udpated.xml` after provisioning and check for keyword `debug` or `ship`.
- `Debug` - everything under this XML node should be disabled or left empty
- `BtGuardCpuDebugEnable` - should be disabled to disallow CPU debugging
- `BtGuardBspInitEnable` - should be disabled to avoid LAPIC INIT signal attack
- In Slim Bootloader, ensure `MEMORY_CFG_DATA.PlatformDebugConsent` is set to 0.

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Production settings

Flash protections

- Slim Bootloader already sets SMM BIOS Write Protection (SMM BWP) and Flash Protected Ranges (FPR)
  - Flash protected ranges will prevent changes to Slim Bootlaoder code partitions
  - SMM BIOS Write Protection will prevent software from writing to BIOS region in the flash, unless an authorized entity will do it via SMI handler. BIOS controls installation of SMI handlers, so only the BIOS is trusted to perform writes to BIOS region.
  - Slim Bootloader does not support SMM, thus an attempt to read or write to flash ends up with a hang (missing SMI handler for SMM BWP)
- The protections only work for software programming methods on target and do not protect against external programmers
- The protections are not set during the boot on flash update in Slim Bootloader, when an capsule update is invoked

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Production settings

Flash protections

```
root@DTS-ODROID# flashrom -p internal

flashrom v1.2-1039-ga961af7a on Linux 6.6.21-yocto-standard (x86_64)
flashrom is free software, get the source code at https://flashrom.org

Using clock_gettime for delay loops (clk_id: 1, resolution: 1ns).
Found chipset "Intel Alder Lake-N".
Enabling flash write... Warning: BIOS region SMM protection is enabled!
Warning: Setting BIOS Control at 0×dc from 0×aa to 0×89 failed.
New value is 0×ab.
SPI Configuration is locked down.
FREG0: Flash Descriptor region (0×00000000-0×00000fff) is read-write.
FREG1: BIOS region (0×00600000-0×00ffffff) is read-write.
FREG2: Management Engine region (0×00001000-0×00413fff) is read-write.
PR0: Warning: 0×00600000-0×00919fff is read-only.
PR1: Warning: 0×0095a000-0×00ffffff is read-only.
At least some flash regions are write protected. For write operations,
you should use a flash layout and include only writable regions. See
manpage for more details.
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Production settings

Flash protections

```
root@DTS-ODROID# flashrom -p internal

flashrom v1.2-1039-ga961af7a on Linux 6.6.21-yocto-standard (x86_64)
flashrom is free software, get the source code at https://flashrom.org

Using clock_gettime for delay loops (clk_id: 1, resolution: 1ns).
Found chipset "Intel Alder Lake-N".
Enabling flash write... Warning: BIOS region SMM protection is enabled!
Warning: Setting BIOS Control at 0×dc from 0×aa to 0×89 failed.
New value is 0×ab.
SPI Configuration is locked down.
FREG0: Flash Descriptor region (0×00000000-0×00000fff) is read-write.
FREG1: BIOS region (0×00600000-0×00ffffff) is read-write.
FREG2: Management Engine region (0×00001000-0×00413fff) is read-write.
PR0: Warning: 0×00600000-0×00919fff is read-only.
PR1: Warning: 0×0095a000-0×00ffffff is read-only.
At least some flash regions are write protected. For write operations,
you should use a flash layout and include only writable regions. See
manpage for more details.
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Production settings

Flash protections

```
root@DTS-ODROID# flashrom -p internal

flashrom v1.2-1039-ga961af7a on Linux 6.6.21-yocto-standard (x86_64)
flashrom is free software, get the source code at https://flashrom.org

Using clock_gettime for delay loops (clk_id: 1, resolution: 1ns).
Found chipset "Intel Alder Lake-N".
Enabling flash write... Warning: BIOS region SMM protection is enabled!
Warning: Setting BIOS Control at 0xdc from 0xaa to 0x89 failed.
New value is 0xab.
SPI Configuration is locked down.
FREG0: Flash Descriptor region (0x00000000-0x00000fff) is read-write.
FREG1: BIOS region (0x00600000-0x00ffffff) is read-write.
FREG2: Management Engine region (0x00001000-0x00413fff) is read-write.
PR0: Warning: 0x00600000-0x00919fff is read-only.
PR1: Warning: 0x0095a000-0x00ffffff is read-only.
At least some flash regions are write protected. For write operations,
you should use a flash layout and include only writable regions. See
manpage for more details.
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Production settings

Flash protections

```
root@DTS-ODROID# flashrom -p internal

flashrom v1.2-1039-ga961af7a on Linux 6.6.21-yocto-standard (x86_64)
flashrom is free software, get the source code at https://flashrom.org

Using clock_gettime for delay loops (clk_id: 1, resolution: 1ns).
Found chipset "Intel Alder Lake-N".
Enabling flash write... Warning: BIOS region SMM protection is enabled!
Warning: Setting BIOS Control at 0×dc from 0×aa to 0×89 failed.
New value is 0×ab.
SPI Configuration is locked down.
FREG0: Flash Descriptor region (0×00000000-0×00000fff) is read-write.
FREG1: BIOS region (0×00600000-0×00ffffff) is read-write.
FREG2: Management Engine region (0×00001000-0×00413fff) is read-write.
PR0: Warning: 0×00600000-0×00919fff is read-only.
PR1: Warning: 0×0095a000-0×00ffffff is read-only.
At least some flash regions are write protected. For write operations,
you should use a flash layout and include only writable regions. See
manpage for more details.
```

Production settings

Flash protections

```
root@DTS-ODROID# flashrom -p internal

flashrom v1.2-1039-ga961af7a on Linux 6.6.21-yocto-standard (x86_64)
flashrom is free software, get the source code at https://flashrom.org

Using clock_gettime for delay loops (clk_id: 1, resolution: 1ns).
Found chipset "Intel Alder Lake-N".
Enabling flash write... Warning: BIOS region SMM protection is enabled!
Warning: Setting BIOS Control at 0×dc from 0×aa to 0×89 failed.
New value is 0×ab.
SPI Configuration is locked down.
FREG0: Flash Descriptor region (0×00000000-0×00000fff) is read-write.
FREG1: BIOS region (0×00600000-0×00ffffff) is read-write.
FREG2: Management Engine region (0×00001000-0×00413fff) is read-write.
PR0: Warning: 0×00600000-0×00919fff is read-only.
PR1: Warning: 0×0095a000-0×00ffffff is read-only.
At least some flash regions are write protected. For write operations,
you should use a flash layout and include only writable regions. See
manpage for more details.
```

Production settings

Flash protections

- Flash descriptor region protection is important for the End of Manufacturing flow and preparing the platform for production shipment
  - Intel recommends that each flash master should be able to write only to its own region (with small exceptions for BIOS flash master being able to write to EC)
  - Intel recommended flash protection settings can be found in SPI Programming Guide inside the ME toolkit.
  - The document also describes other recommended settings that should be performed by the BIOS (Slim Bootloader lock everything as it should)

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Production settings - Flash protections

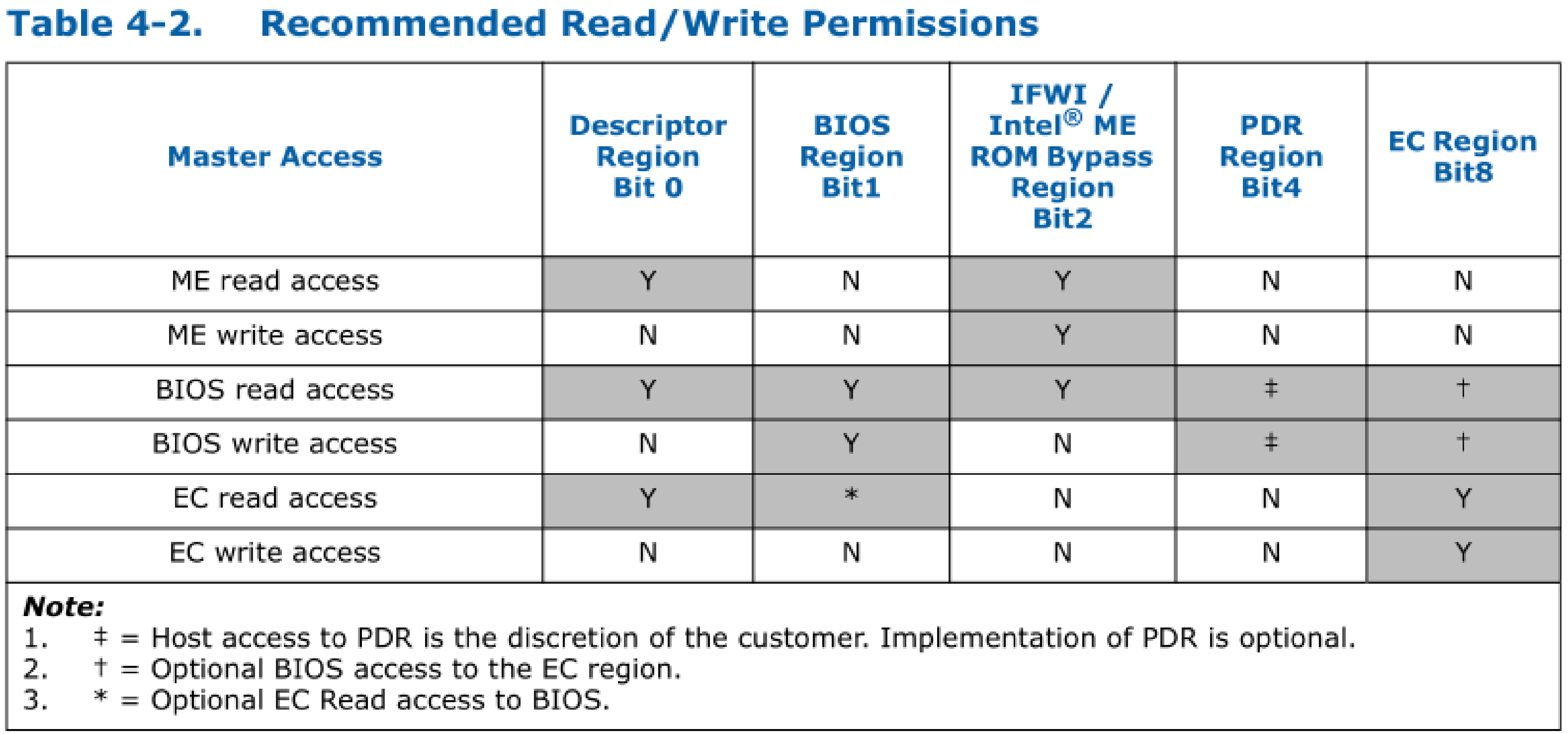


Table 4-2. Recommended Read/Write Permissions

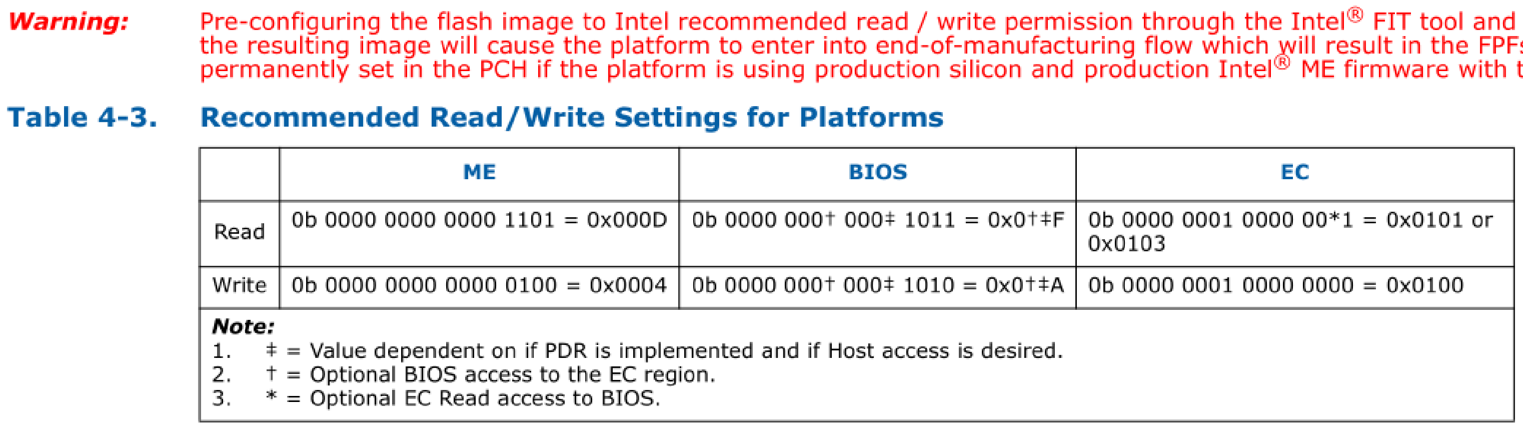
Master Access	Descriptor Region Bit 0	BIOS Region Bit1	IFWI / Intel® ME ROM Bypass Region Bit2	PDR Region Bit4	EC Region Bit8
ME read access	Y	N	Y	N	N
ME write access	N	N	Y	N	N
BIOS read access	Y	Y	Y	‡	†
BIOS write access	N	Y	N	‡	†
EC read access	Y	*	N	N	Y
EC write access	N	N	N	N	Y

Note:
1. ‡ = Host access to PDR is the discretion of the customer. Implementation of PDR is optional.
2. † = Optional BIOS access to the EC region.
3. * = Optional EC Read access to BIOS.

Alderlake-N Client SPI Programming Guide.pdf

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Production settings - Flash protections



Warning: Pre-configuring the flash image to Intel recommended read / write permission through the Intel® FIT tool and then flashing the resulting image will cause the platform to enter into end-of-manufacturing flow which will result in the FPFs being permanently set in the PCH if the platform is using production silicon and production Intel® ME firmware with the PV bit set.

Table 4-3. Recommended Read/Write Settings for Platforms

	ME	BIOS	EC
Read	0b 0000 0000 0000 1101 = 0x000D	0b 0000 000† 000‡ 1011 = 0x0†‡F	0b 0000 0001 0000 00*1 = 0x0101 or 0x0103
Write	0b 0000 0000 0000 0100 = 0x0004	0b 0000 000† 000‡ 1010 = 0x0†‡A	0b 0000 0001 0000 0000 = 0x0100

Note:
1. ‡ = Value dependent on if PDR is implemented and if Host access is desired.
2. † = Optional BIOS access to the EC region.
3. * = Optional EC Read access to BIOS.

Alderlake-N Client SPI Programming Guide.pdf

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

End of Manufacturing

- End of Manufacturing (EOM) is a process that transitions ME from Manufacturing Mode to production mode, in short. But that's not all that happens.
- Depending on EOM settings in ME (set by MFIT), the OEM process may:
  - be invoked at first boot
  - may lock OEM configuration
  - may lock flash descriptor (set recommended region access permissions)
- What EOM process always does:
  - Sets `/home/mca/eom` file in MFS which triggers other actions in ME
  - Blow the Field Programmable Fuses (FPFs) based on current contents of emulated fuses in Unified Emulation Partition (UEP)

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

End of Manufacturing

- Behavior of EOM may be altered using Flexible EOM settings in MFIT

```xml
<EndofManufacturingConfiguration label="End of Manufacturing Configuration">
  <FlexibleEomSettings value="Lock Descriptor and OEM Configs"
    value_list="['Lock Descriptor and OEM Configs', 'Lock OEM Configs Only',
    'Lock Descriptor Only', 'Do not lock Descriptor and OEM Configs']"
    label="Flexible EOM setting options"
    help_text="This setting deteremines which settings will be automatically
    commited during End of Manufacturing flows. Note: The FPFs, RPMB / RPMC
    and set manufacturing mode settings are mandatory and cannot be overridden
    revenue parts. Simulation can be done on non-revenue part with the Hardware
    binding set to disabled."
    key="CsePlugin:EomNvar:EOM_Config#FlexibleEomSettings"
  />
  <EomFirstBootEnabled value="No" value_list="['No', 'Yes']"
    label="EOM on First Boot Enabled"
    help_text="This setting determines if End of Manufacturing will be triggered
     on first boot of the platform after flashing. Note: When this setting is
     enabled Intel(R) CSME will enter End of Manufacturing regardless of the
     descriptor settings."
     key="CsePlugin:EomNvar:EOM#EomFirstBootEnabled"
   />
</EndofManufacturingConfiguration>
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

End of Manufacturing

- Behavior of EOM may be altered using Flexible EOM settings in MFIT

```xml
<EndofManufacturingConfiguration label="End of Manufacturing Configuration">
  <FlexibleEomSettings value="Lock Descriptor and OEM Configs"
    value_list="['Lock Descriptor and OEM Configs', 'Lock OEM Configs Only',
    'Lock Descriptor Only', 'Do not lock Descriptor and OEM Configs']"
    label="Flexible EOM setting options"
    help_text="This setting deteremines which settings will be automatically
    commited during End of Manufacturing flows. Note: The FPFs, RPMB / RPMC
    and set manufacturing mode settings are mandatory and cannot be overridden
    revenue parts. Simulation can be done on non-revenue part with the Hardware
    binding set to disabled."
    key="CsePlugin:EomNvar:EOM_Config#FlexibleEomSettings"
  />
  <EomFirstBootEnabled value="No" value_list="['No', 'Yes']"
    label="EOM on First Boot Enabled"
    help_text="This setting determines if End of Manufacturing will be triggered
     on first boot of the platform after flashing. Note: When this setting is
     enabled Intel(R) CSME will enter End of Manufacturing regardless of the
     descriptor settings."
     key="CsePlugin:EomNvar:EOM#EomFirstBootEnabled"
   />
</EndofManufacturingConfiguration>
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

End of Manufacturing

- Behavior of EOM may be altered using Flexible EOM settings in MFIT

```
<EndofManufacturingConfiguration label="End of Manufacturing Configuration">
  <FlexibleEomSettings value="Lock Descriptor and OEM Configs"
    value_list="['Lock Descriptor and OEM Configs', 'Lock OEM Configs Only',
    'Lock Descriptor Only', 'Do not lock Descriptor and OEM Configs']"
    label="Flexible EOM setting options"
    help_text="This setting deteremines which settings will be automatically
    commited during End of Manufacturing flows. Note: The FPFs, RPMB / RPMC
    and set manufacturing mode settings are mandatory and cannot be overridden
    revenue parts. Simulation can be done on non-revenue part with the Hardware
    binding set to disabled."
    key="CsePlugin:EomNvar:EOM_Config#FlexibleEomSettings"
  />
  <EomFirstBootEnabled value="No" value_list="['No', 'Yes']"
    label="EOM on First Boot Enabled"
    help_text="This setting determines if End of Manufacturing will be triggered
     on first boot of the platform after flashing. Note: When this setting is
     enabled Intel(R) CSME will enter End of Manufacturing regardless of the
     descriptor settings."
     key="CsePlugin:EomNvar:EOM#EomFirstBootEnabled"
   />
</EndofManufacturingConfiguration>
```

End of Manufacturing

Danger

Do not do this unless absolutely sure what you are doing. Below commands will blow the fuses in the chipset/SoC causing irreversible changes to the hardware!

Ways to do the End of Manufacturing (starting from simplest):

1. Set EOM on first boot with MFIT and flash resulting image:

   ```
   ./Fit/mfit --decompose sbl_ifwi_adln.bin --setvalues \
     "CsePlugin:EomNvar:EOM#EomFirstBootEnabled=Yes" \
     --build sbl_ifwi_adln_eom.bin
   ```

2. Set recommended region access permissions in flash descriptor to lock it (choose any tooling you like: flashrom, MFIT or FPT) and then boot the platform with locked descriptor.

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

End of Manufacturing

Danger

Do not do this unless absolutely sure what you are doing. Below commands will blow the fuses in the chipset/SoC causing irreversible changes to the hardware!

3. FPT (on target board in OS):

```
./FPT -CLOSEMNF -NORESET
```

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Verifying EOM process

- `FPF committed` / `FPF SoC Config Lock` is set in 6th ME FW status register (bit 30) to be found in:
  - `/sys/class/mei/mei0/fw_status` output
  - `MEInfo` output (`FPF Committed` is Yes and `SOC Config Lock State` is Enabled in FPF column)
- Recommended region access permissions in flash descriptor are reflected in the 1st ME FW status register (bit 4) to be found in:
  - `/sys/class/mei/mei0/fw_status` output
  - `MEInfo` output (`Flash Protection Mode` is `Unprotected` if permissions are not set to Intel recommended)
- Run `MEInfo` to check if all FPF fuses have the desired values.

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Verifying EOM process

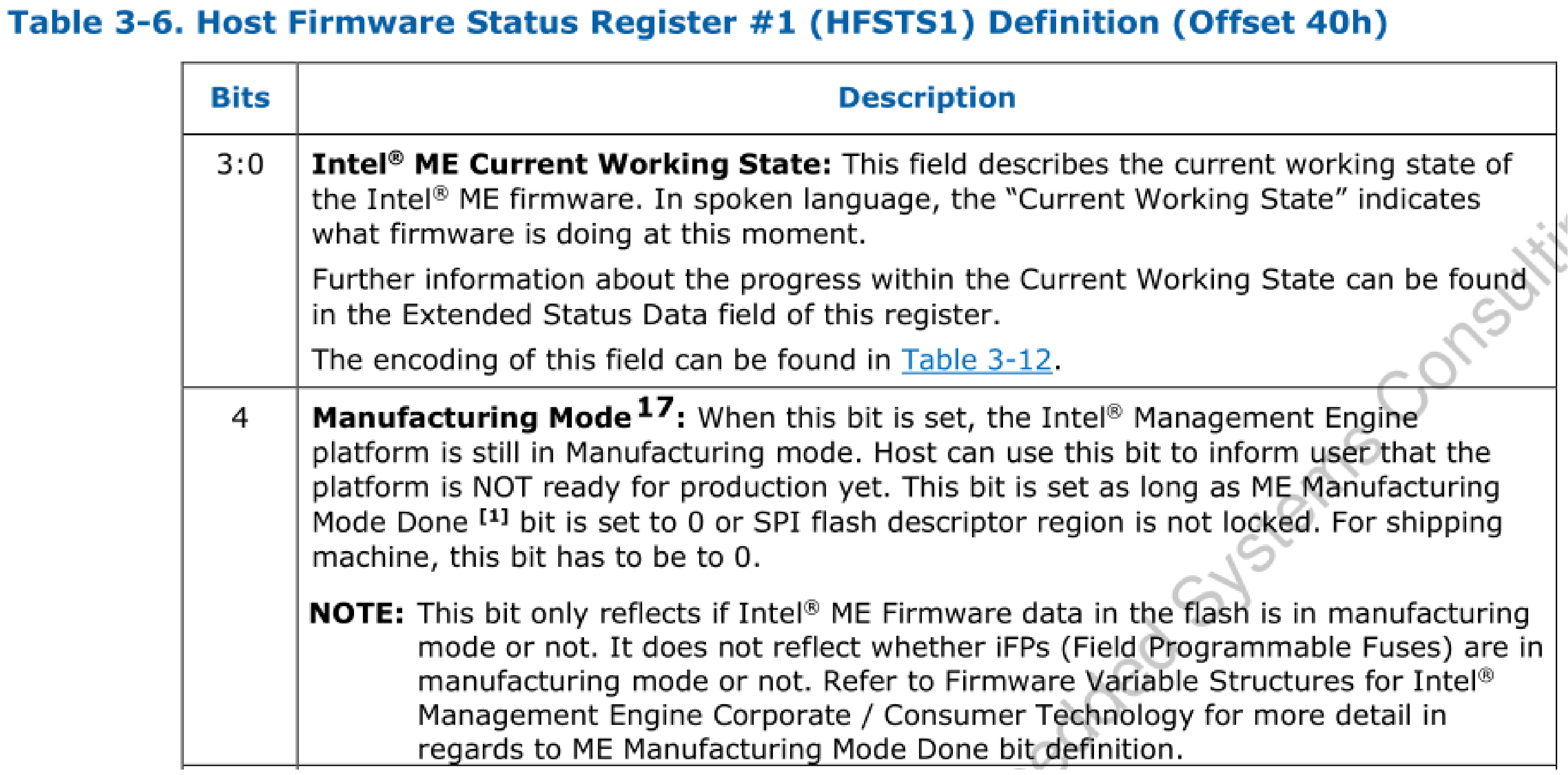


Table 3-6. Host Firmware Status Register #1 (HFSTS1) Definition (Offset 40h)

Bits	Description
3:0	**Intel® ME Current Working State:** This field describes the current working state of the Intel® ME firmware. In spoken language, the "Current Working State" indicates what firmware is doing at this moment. Further information about the progress within the Current Working State can be found in the Extended Status Data field of this register. The encoding of this field can be found in Table 3-12.
4	**Manufacturing Mode** 17**:** When this bit is set, the Intel® Management Engine platform is still in Manufacturing mode. Host can use this bit to inform user that the platform is NOT ready for production yet. This bit is set as long as ME Manufacturing Mode Done [1] bit is set to 0 or SPI flash descriptor region is not locked. For shipping machine, this bit has to be to 0. **NOTE:** This bit only reflects if Intel® ME Firmware data in the flash is in manufacturing mode or not. It does not reflect whether iFPs (Field Programmable Fuses) are in manufacturing mode or not. Refer to Firmware Variable Structures for Intel® Management Engine Corporate / Consumer Technology for more detail in regards to ME Manufacturing Mode Done bit definition.

Skylake/Kaby Lake - ME 11.x BIOS Specification (doc 549522)

Verifying EOM process

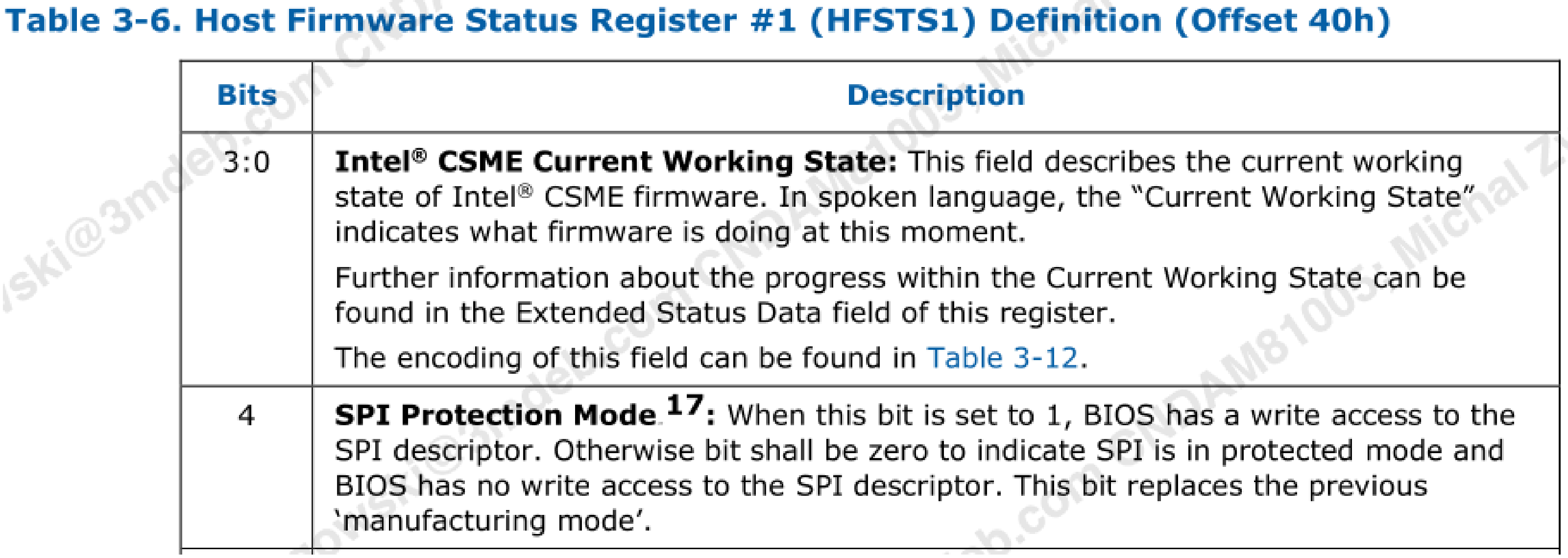


Table 3-6. Host Firmware Status Register #1 (HFSTS1) Definition (Offset 40h)

Bits	Description
3:0	**Intel® CSME Current Working State:** This field describes the current working state of Intel® CSME firmware. In spoken language, the "Current Working State" indicates what firmware is doing at this moment. Further information about the progress within the Current Working State can be found in the Extended Status Data field of this register. The encoding of this field can be found in Table 3-12.
4	**SPI Protection Mode** [17]**:** When this bit is set to 1, BIOS has a write access to the SPI descriptor. Otherwise bit shall be zero to indicate SPI is in protected mode and BIOS has no write access to the SPI descriptor. This bit replaces the previous 'manufacturing mode'.

Tiger Lake - ME 15.0 BIOS Specification (doc 612229)

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Conclusion

In this lecture, we learned about the following:

- What are the recommended settings for Intel ME and BIOS for production.
- How to perform basic CPU debugging with Intel DCI.
- What is the production behavior of Boot Guard with its most restrictive profiles.
- What is the End of Manufacturing (EOM) process and how to perform it.

© Copyright 2025. All Rights Reserved by 3mdeb Sp. z o.o.

Q&A