Western Digital®

The future of RISC-V Supervisor Binary Interface (SBI)

Atish Patra

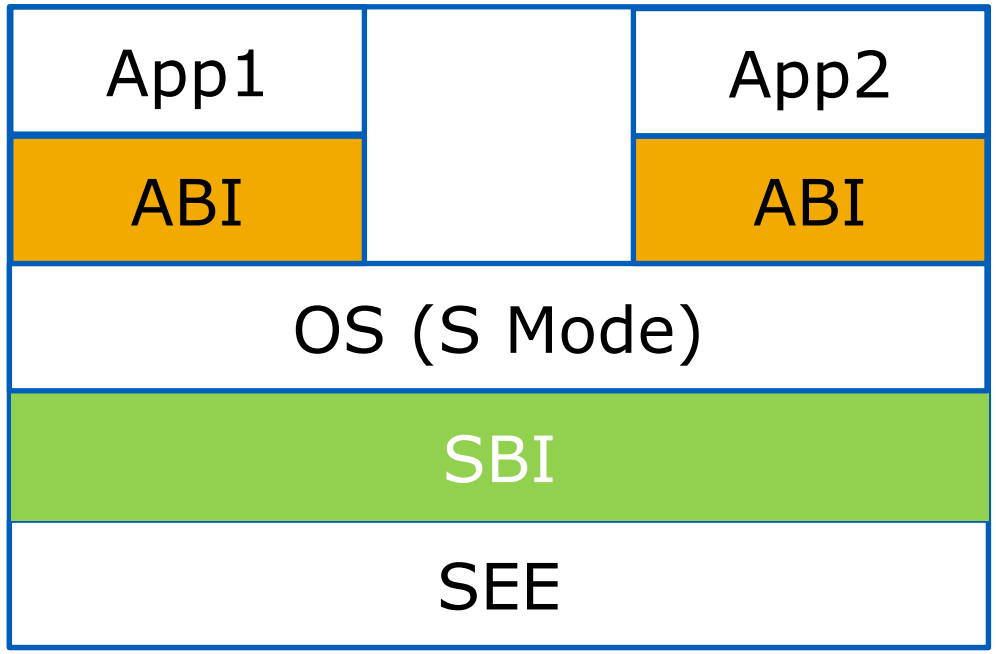
Principal R&D Engineer

Outline

- RISC-V Supervisor Binary Interface(SBI)
- Current status
- Limitations
- Extension proposal
- OpenSBI project
- Boot Flow
- Demo

RISC-V Supervisor Binary Interface (SBI)

- Provides an interface to access machine mode only registers
- Clear separation between Supervisor & Supervisor Execution Environment (SEE)
- Helps to run single OS image across different SEE
- Currently implemented by the Berkeley Boot Loader (BBL)
- Calling convention
  - S mode traps into M mode using *ecall* instructions
  - Arguments are passed via registers a0-a2
  - SBI call type is specified via register a7
  - Unsupported SBI returns -38 (ENOSYS in Linux)
  - a0 is clobbered register
- Documentation available at
  - https://github.com/riscv/riscv-sbi-doc



Current Interface

Type	Function	Function ID
Timer	sbi_set_timer	0
Console	sbi_console_putchar sbi_console_getchar	1 2
IPI	sbi_clear_ipi sbi_send_ipi	3 4
Memory Model	sbi_remote_fence_i sbi_remote_sfence_vma sbi_remote_sfence_vma_asid	5 6 7
Shutdown	sbi_shutdown	8

Limitations

- Fixed, Not extendable
- No way to modify existing function signatures
- Changes cannot easily maintain backward compatibility
- No clean way to add new SBI function calls
  - Power management
  - Hart hotplug
  - Vendor specific extensions

SBI Scope

- Shouldn’t be treated as a kitchen sink
- New functionality only if absolutely necessary
- Backward compatibility
- No mandated usage of DT or ACPI
- Any functionality can be replaced by S-mode in future
- Anything else ??

SBI proposal working model

- SBI specification to be part of the RISC-V Unix class platform specification
- Need to be approved by RISC-V Unix class platform specification working group
- Streamline the proposal discussion and quick turn around time
  - A mandatory base SBI spec
  - Existing SBI spec will be considered as legacy extension
  - Every other SBI feature set will be a separate extension based on the base spec
  - Every extension will be a sub-specification
  - Can be discussed in parallel once the base SBI specification is finalized
- Need to have an implementation before freeze

SBI Base specification - I

- Calling Convention
  - Follows existing calling convention except return type
  - May return a structure

```
struct sbi_ret {
        long value;
        long error;
}
```

  - Value in a0 as return value or error from SBI function
  - Error in a1 as any error SBI library wants to return

- A versioning scheme
  - 32bit unsigned integer
  - <major.minor> scheme

```
31                    16                  0
-----------------------------------------
Major Version         |  Minor Version    |
```

  - The existing legacy SBI version will be 0.1
  - The proposed base SBI version will be 0.2

SBI Base specification - II

- A SBI function ID scheme
  - A combination of function set number and function type number.

```
31            20          16                 0
-----------------------------------------------
Function Set |  Reserved |   Function Type  |
```

  - Statically defined function set numbers.
  - Both Hart/System power management functions will be a sub-specification.
  - Use Reserved set in case of function set is not clear and may be standardized in future
  - Vendors specific functionalities should use Vendor function set.

Function Set	Value	Description
Legacy	0x000	Existing SBI Functions. Not mandatory.
Base	0x001	Base Functions mandatory for any SBI version.
Hart PM	0x002	Hart power management.
System PM	0x003	System-level power management
Reserved	0x010-0x7ff	Reserved for experimental extensions
Vendor	0x800-0xfff	Vendor specific Functions

SBI Base version Functions

Function Type	Description
sbi_get_version(void)	Returns SBI specification version
sbi_set_sbiimp_version(void)	Returns SBI implementation version
sbi_is_function_set(u32 fset)	Check if given function set is valid or not
sbiret sbi_is_function_type(u32 ftype, u32 fset)	Check if a function type is implemented for a give function set.
sbiret sbi_get_vendor_id(void)	Returns the vendor ID
sbiret sbi_get_mimp_id(void)	Returns the machine implementation ID
sbi_get_sbiimp_id	Returns the SBI implementation ID

OpenSBI project – Why ?

- BBL/Coreboot provides separate SBI implementation
- More fragmentations expected going forward considering vendor specific usage
- Difficult to maintain & track the SBI changes as it evolves
- Need a BBL replacement.
- Need easy plugin model for different platform/soc vendors
- OpenSBI to the rescue!!

OpenSBI project – What ?

- An Open Source SBI implementation project
- Driven by community
- Licensed under BSD-2 clause
- Builds a static library that any M-mode boot loader can link
- Provides a reference implementation of platform code
- Provides a reference implementation of firmware code as well
- Protects firmware using PMP support
- Source level documentation using Doxygen
- Available at
  - https://github.com/riscv/opensbi

OpenSBI project – How ?

- libsbi.a
  - A static library that provides SBI implementation
  - Other M-mode boot loader may just link this for SBI functionality
  - Every future proposed SBI extension will be implemented

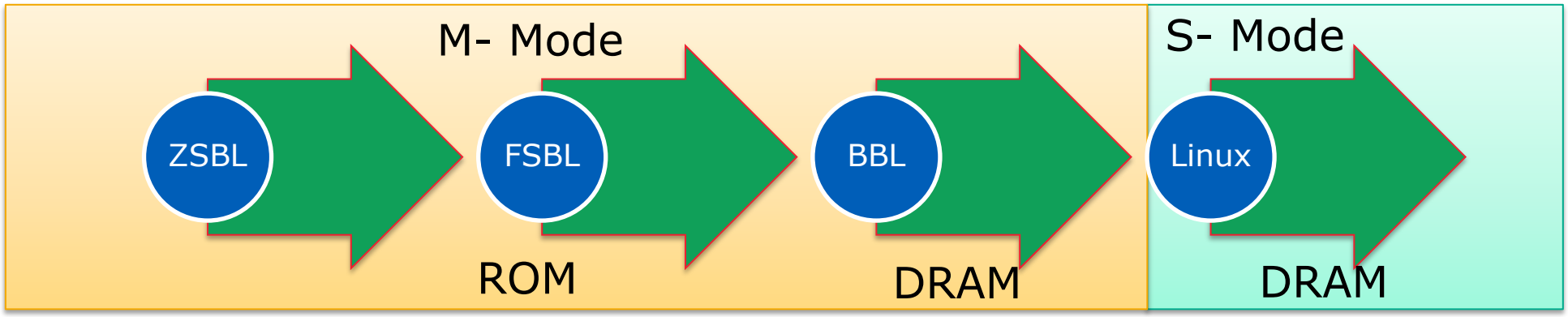
- libplatsbi.a
  - A static library that provides reference platform implementation
  - Contains minimal platform drivers required for bringup
  - Links libsbi.a for sbi implementation
  - Platform vendors are welcome to add their platform support

- Currently supported platforms
  - QEMU Virt
  - QEMU SiFive_u
  - SiFive Fu540 (HiFive Unleashed)
  - Kendryte K210 SoC

OpenSBI project – How ?

- Builds several firmware binary
- A reference implementation
- Platform specific bootable firmware binary
- Firmware with Payload
  - Any higher stage boot loader i.e. U-Boot binary as payload
  - Supervisor OS i.e. Linux as a direct payload
  - Allows separate device tree linking
- Firmware with Jump Address
  - Jumps to address of next booting stage entry
  - No need to provide payload binary for next stage
  - Booting stage prior to OpenSBI should be capable of loading next stage module
- Vendors may choose to use one of the firmware as is or build their own

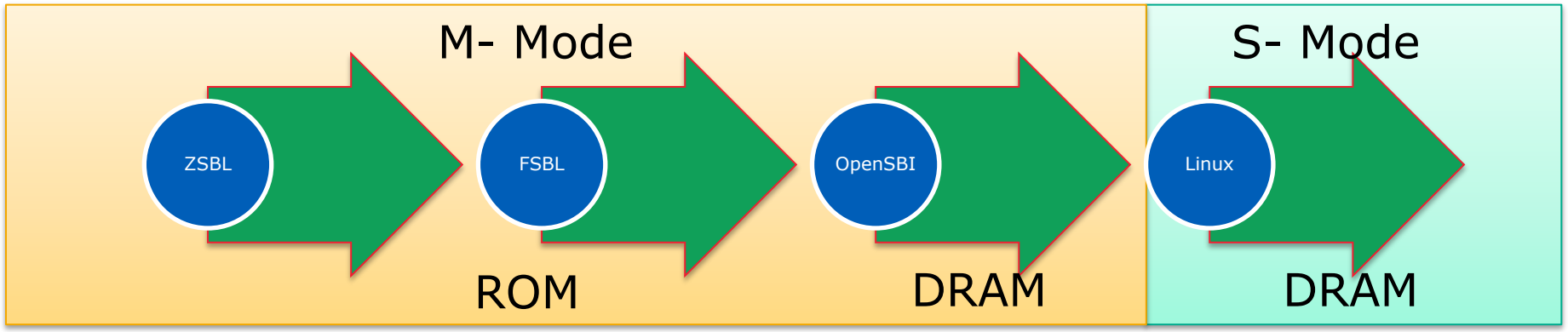
Current Boot flow – BBL + Linux



- Difficult add support for other platforms
- No way to separate DT from kernel image
- No network booting
- Kernel image has to be embedded bbl image

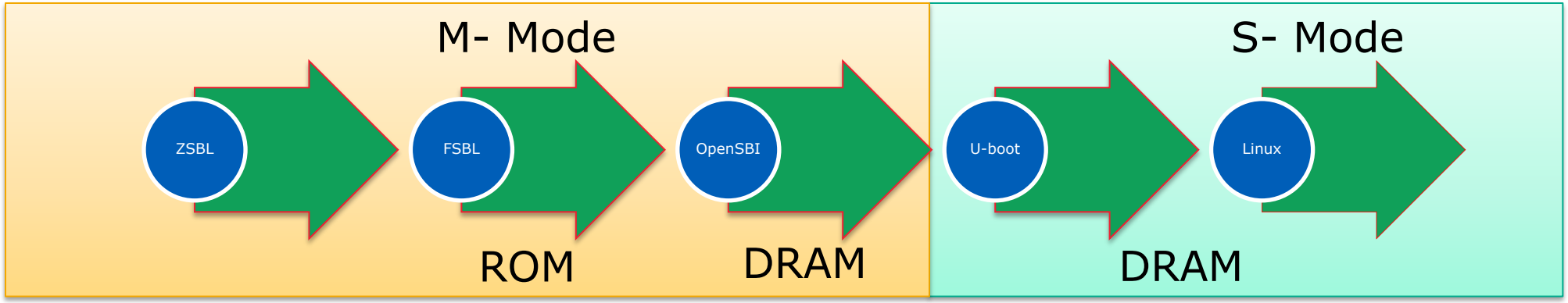
ZSBL – Zero Stage boot loader
FSBL – First Stage boot loader
BBL – Berkeley boot loader

Boot flow – OpenSBI + Linux



- Full SMP support
- Follows existing RISC-V boot flow model
- Takes Linux Image file as a payload
- Can accept separate DT file as well

Boot flow – OpenSBI + U-Boot + Linux



- Follows a standard boot flow
- Uses U-Boot as the last stage boot loader
- U-Boot binary as the payload to OpenSBI
- Linux image loaded via tftp
- Linux SMP (all cores) will be usable once SBI hart power management extension is available in OpenSBI

Booting – QEMU

- Instructions available at docs/platform/qemu_virt.md
- Linux Image as a direct payload
  - Building

```
make PLATFORM=qemu/virt FW_PAYLOAD_PATH=<linux_build_directory>/arch/riscv/boot/Image
```

  - Running

```
qemu-system-riscv64 -M virt -m 256M -display none -serial stdio -kernel build/platform/qemu/virt/firmware/fw_payload.elf \
        -drive file=<path_to_linux_rootfs>,format=raw,id=hd0 \
        -device virtio-blk-device,drive=hd0 \
        -append "root=/dev/vda rw console=ttyS0”
```

- U-Boot image as a payload
  - Building

```
make PLATFORM=qemu/virt FW_PAYLOAD_PATH=<uboot_build_directory>/u-boot.bin
```

  - Running

```
qemu-system-riscv64 -M virt -m 256M -display none -serial stdio -kernel build/platform/qemu/virt/firmware/fw_payload.elf
```

Booting – HiFive Unleashed

- Instructions also available at

  docs/platform/sifive_fu540.md

- Linux Image as a direct payload

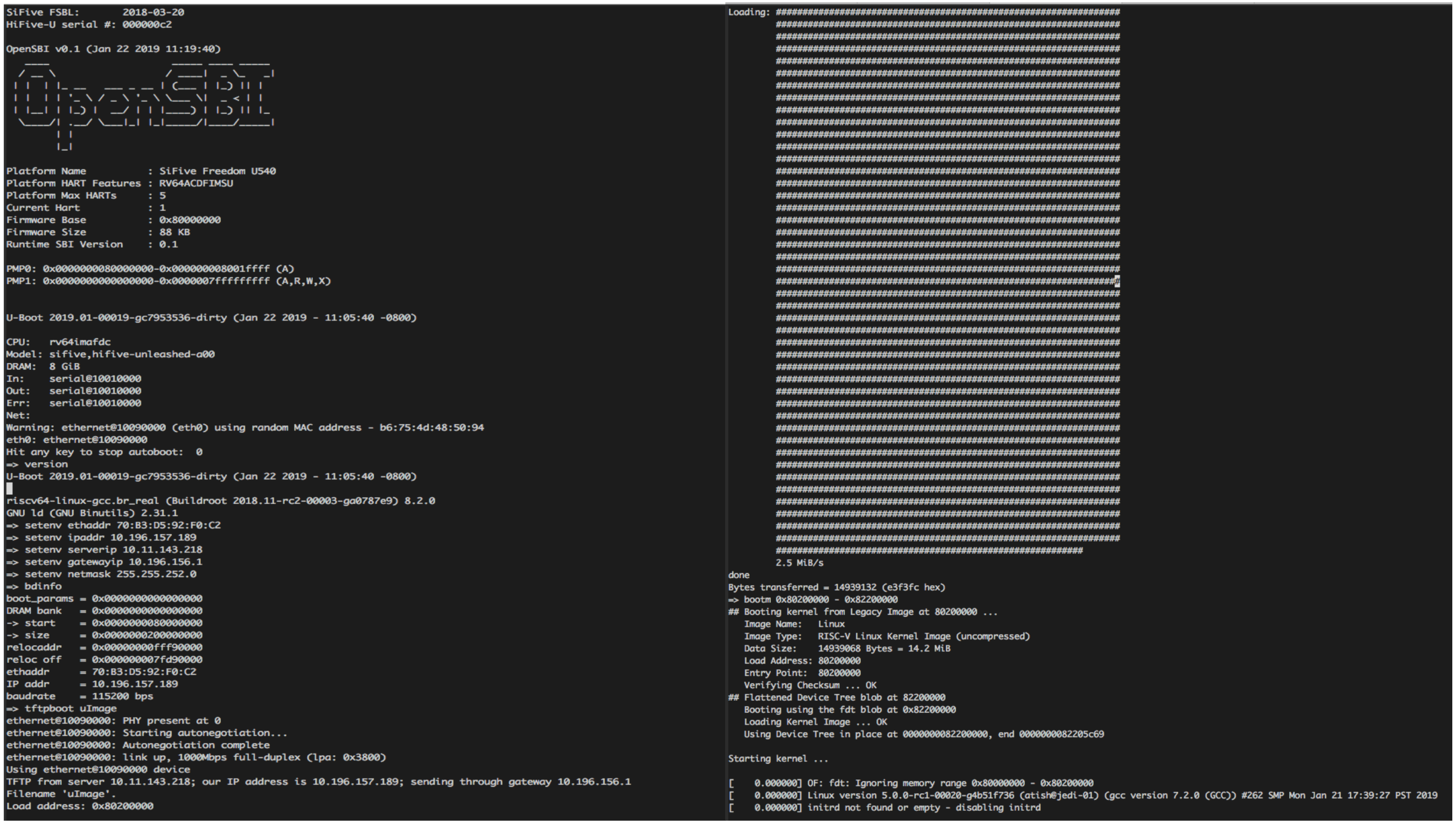
```
make PLATFORM=sifive/fu540 \
FW_PAYLOAD_PATH=<linux_build_directory>/arch/riscv/boot/Image
```

- U-Boot binary as payload

```
make PLAT=sifive/hifive_u540 \
                FW_PAYLOAD_PATH=~/workspace/u-boot-riscv/u-boot.bin
```

```
SiFive FSBL:      2018-03-20
HiFive-U serial #: 000000c2

OpenSBI v0.1 (Jan 22 2019 15:54:06)
   ____                    ____  ___ ____
  / __ \                  / ___|  _ \|  _ \
 | |  | |_ __   ___ _ __ | (___ | |_) | |_) |
 | |  | | '_ \ / _ \ '_ \ \___ \|  _ <|  _ <
 | |__| | |_) |  __/ | | |____) | |_) | |_) |
  \____/| .__/ \___|_| |_|_____/|____/|____/
        | |
        |_|

Platform Name          : SiFive Freedom U540
Platform HART Features : RV64ACDFIMSU
Platform Max HARTs     : 5
Current Hart           : 3
Firmware Base          : 0x80000000
Firmware Size          : 88 KB
Runtime SBI Version    : 0.1

PMP0: 0x0000000080000000-0x000000008001ffff (A)
PMP1: 0x0000000000000000-0x0000007fffffffff (A,R,W,X)
[    0.000000] OF: fdt: Ignoring memory range 0x80000000 - 0x80200000
[    0.000000] Linux version 5.0.0-rc1-00020-g4b51f736 (atish@jedi-01) (gcc version 7.2.0 (GCC)) #264 SMP Tue Jan 22 15:52:41 PST 2019
[    0.000000] initrd not found or empty - disabling initrd
[    0.000000] Zone ranges:
[    0.000000]   DMA32    [mem 0x0000000080200000-0x00000000ffffffff]
[    0.000000]   Normal   [mem 0x0000000100000000-0x000027ffffffffff]
[    0.000000] Movable zone start for each node
[    0.000000] Early memory node ranges
[    0.000000]   node   0: [mem 0x0000000080200000-0x000000027fffffff]
[    0.000000] Initmem setup node 0 [mem 0x0000000080200000-0x000000027fffffff]
[    0.000000] software IO TLB: mapped [mem 0xfbfff000-0xffffff000] (64MB)
[    0.000000] CPU with hartid=0 has a non-okay status of "masked"
[    0.000000] CPU with hartid=0 has a non-okay status of "masked"
[    0.000000] elf_hwcap is 0x112d
[    0.000000] percpu: Embedded 15 pages/cpu @(____ptrval____) s29720 r0 d31720 u61440
[    0.000000] Built 1 zonelists, mobility grouping on.  Total pages: 2067975
[    0.000000] Kernel command line: earlyprintk
[    0.000000] Dentry cache hash table entries: 1048576 (order: 11, 8388608 bytes)
[    0.000000] Inode-cache hash table entries: 524288 (order: 10, 4194304 bytes)
[    0.000000] Sorting __ex_table...
[    0.000000] Memory: 8178760K/8386560K available (3309K kernel code, 248K rwdata, 872K rodata, 9381K init, 763K bss, 207800K reserved, 0K cma-reserved)
[    0.000000] SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=4, Nodes=1
[    0.000000] rcu: Hierarchical RCU implementation.
[    0.000000] rcu:     RCU event tracing is enabled.
[    0.000000] rcu:     RCU restricting CPUs from NR_CPUS=8 to nr_cpu_ids=4.
[    0.000000] rcu: RCU calculated value of scheduler-enlistment delay is 10 jiffies.
[    0.000000] rcu: Adjusting geometry for rcu_fanout_leaf=16, nr_cpu_ids=4
[    0.000000] NR_IRQS: 0, nr_irqs: 0, preallocated irqs: 0
[    0.000000] plic: mapped 53 interrupts to 4 (out of 9) handlers.
[    0.000000] riscv_timer_init_dt: Registering clocksource cpuid [0] hartid [3]
[    0.000000] clocksource: riscv_clocksource: mask: 0xffffffffffffffff max_cycles: 0x1d854df40, max_idle_ns: 3526361616960 ns
[    0.000007] sched_clock: 64 bits at 1000kHz, resolution 1000ns, wraps every 2199023255500ns
[    0.000210] Console: colour dummy device 80x25
[    0.000889] printk: console [tty0] enabled
[    0.000950] Calibrating delay loop (skipped), value calculated using timer frequency.. 2.00 BogoMIPS (lpj=10000)
[    0.001022] pid_max: default: 32768 minimum: 301
[    0.001526] Mount-cache hash table entries: 16384 (order: 5, 131072 bytes)
[    0.001897] Mountpoint-cache hash table entries: 16384 (order: 5, 131072 bytes)
[    0.003526] rcu: Hierarchical SRCU implementation.
[    0.004332] smp: Bringing up secondary CPUs ...
[    0.005932] smp: Brought up 1 node, 4 CPUs
```

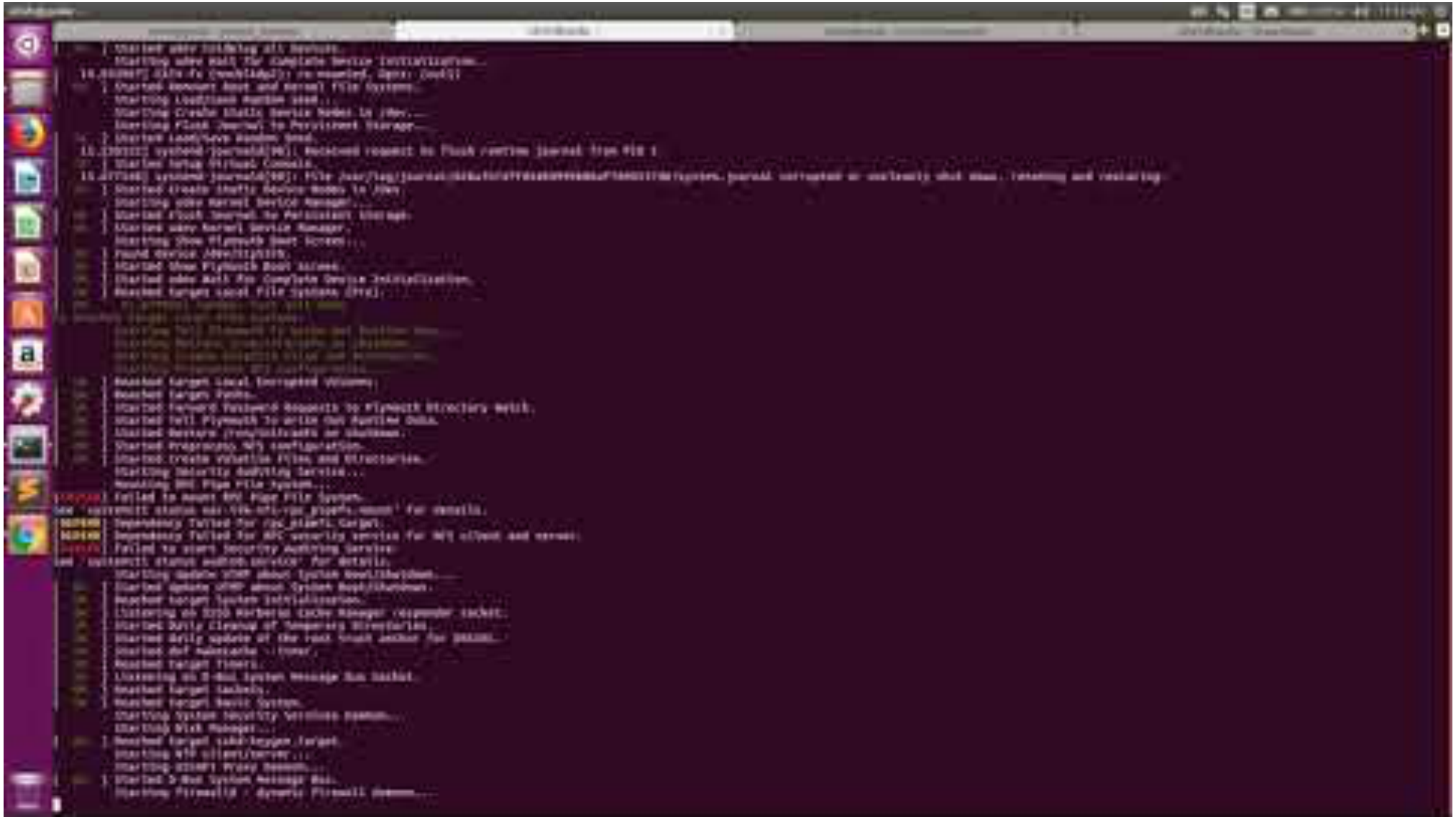
Booting – HiFive Unleashed



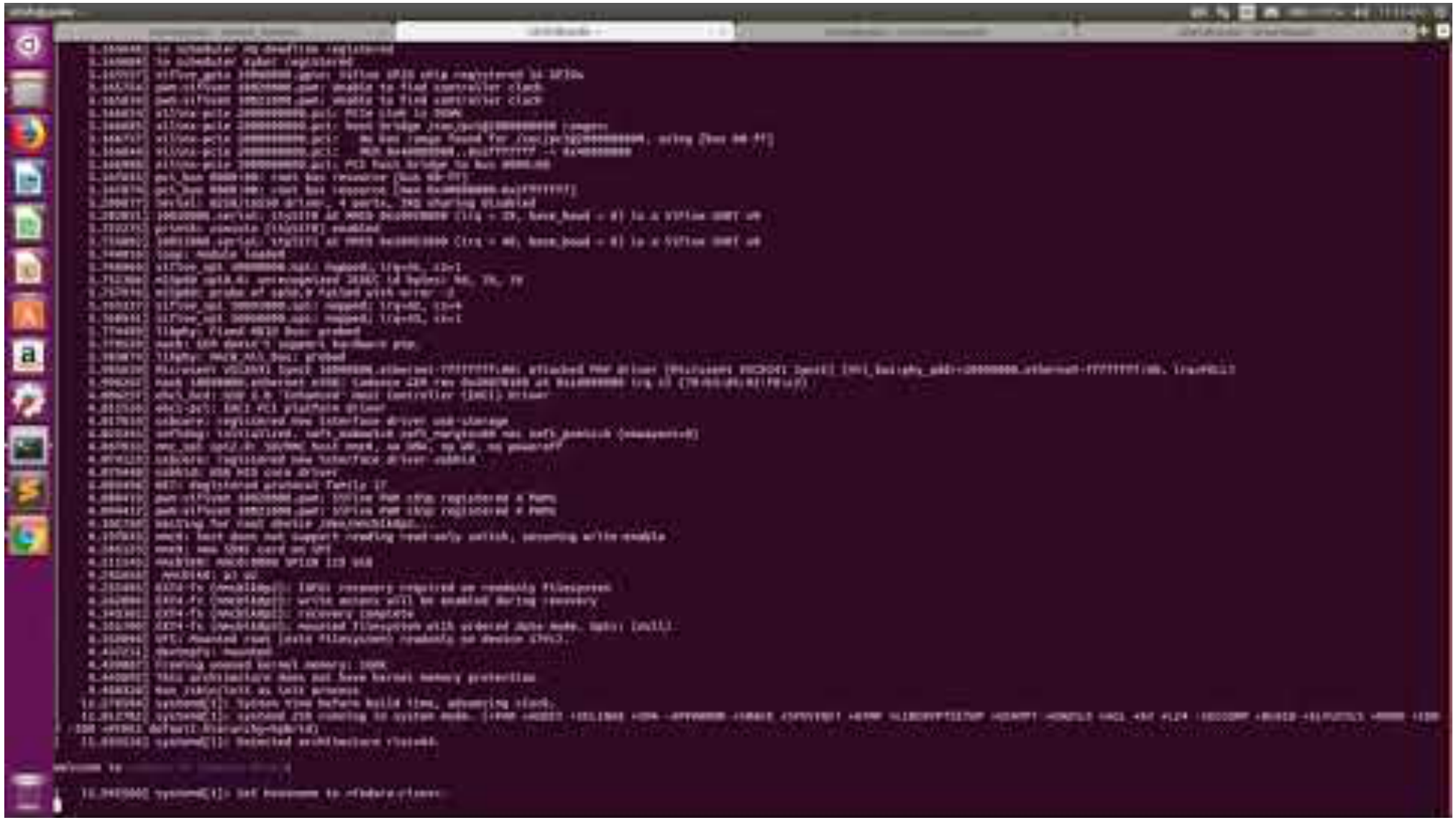
```
SiFive FSBL:       2018-03-20
HiFive-U serial #: 000000c2

OpenSBI v0.1 (Jan 22 2019 11:19:40)
   ____                    _____ ____ _____
  / __ \                  / ____|  _ \_   _|
 | |  | |_ __   ___ _ __ | (___ | |_) || |
 | |  | | '_ \ / _ \ '_ \ \___ \|  _ < | |
 | |__| | |_) |  __/ | | |____) | |_) || |_
  \____/| .__/ \___|_| |_|_____/|____/_____|
        | |
        |_|

Platform Name          : SiFive Freedom U540
Platform HART Features : RV64ACDFIMSU
Platform Max HARTs     : 5
Current Hart           : 1
Firmware Base          : 0x80000000
Firmware Size          : 88 KB
Runtime SBI Version    : 0.1

PMP0: 0x0000000080000000-0x000000008001ffff (A)
PMP1: 0x0000000000000000-0x0000007fffffffff (A,R,W,X)


U-Boot 2019.01-00019-gc7953536-dirty (Jan 22 2019 - 11:05:40 -0800)

CPU:   rv64imafdc
Model: sifive,hifive-unleashed-a00
DRAM:  8 GiB
In:    serial@10010000
Out:   serial@10010000
Err:   serial@10010000
Net:
Warning: ethernet@10090000 (eth0) using random MAC address - b6:75:4d:48:50:94
eth0: ethernet@10090000
Hit any key to stop autoboot:  0
=> version
U-Boot 2019.01-00019-gc7953536-dirty (Jan 22 2019 - 11:05:40 -0800)
█
riscv64-linux-gcc.br_real (Buildroot 2018.11-rc2-00003-ga0787e9) 8.2.0
GNU ld (GNU Binutils) 2.31.1
=> setenv ethaddr 70:B3:D5:92:F0:C2
=> setenv ipaddr 10.196.157.189
=> setenv serverip 10.11.143.218
=> setenv gatewayip 10.196.156.1
=> setenv netmask 255.255.252.0
=> bdinfo
boot_params = 0x0000000000000000
DRAM bank   = 0x0000000000000000
-> start    = 0x0000000080000000
-> size     = 0x0000000200000000
relocaddr   = 0x00000000fff90000
reloc off   = 0x000000007fd90000
ethaddr     = 70:B3:D5:92:F0:C2
IP addr     = 10.196.157.189
baudrate    = 115200 bps
=> tftpboot uImage
ethernet@10090000: PHY present at 0
ethernet@10090000: Starting autonegotiation...
ethernet@10090000: Autonegotiation complete
ethernet@10090000: link up, 1000Mbps full-duplex (lpa: 0x3800)
Using ethernet@10090000 device
TFTP from server 10.11.143.218; our IP address is 10.196.157.189; sending through gateway 10.196.156.1
Filename 'uImage'.
Load address: 0x80200000
```

```
Loading: #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         #################################################################
         ###########################################################
         2.5 MiB/s
done
Bytes transferred = 14939132 (e3f3fc hex)
=> bootm 0x80200000 - 0x82200000
## Booting kernel from Legacy Image at 80200000 ...
   Image Name:   Linux
   Image Type:   RISC-V Linux Kernel Image (uncompressed)
   Data Size:    14939068 Bytes = 14.2 MiB
   Load Address: 80200000
   Entry Point:  80200000
   Verifying Checksum ... OK
## Flattened Device Tree blob at 82200000
   Booting using the fdt blob at 0x82200000
   Loading Kernel Image ... OK
   Using Device Tree in place at 0000000082200000, end 0000000082205c69

Starting kernel ...

[    0.000000] OF: fdt: Ignoring memory range 0x80000000 - 0x80200000
[    0.000000] Linux version 5.0.0-rc1-00020-g4b51f736 (atish@jedi-01) (gcc version 7.2.0 (GCC)) #262 SMP Mon Jan 21 17:39:27 PST 2019
[    0.000000] initrd not found or empty - disabling initrd
```

Demo – OpenSBI + Linux



Demo – OpenSBI + U-Boot + Linux



OpenSBI future work

- 32-bit support
- Yocto recipe for OpenSBI in meta-riscv
- SBI v0.2 support (after SBI extension spec frozen)
- SBI Hart power management support
- Link libsbi in U-Boot M mode
- Link libsbi in Coreboot (Volunteer ??)
- CLIC use in OpenSBI (???)
- More platforms …

Thank you!!

- Q&A ?

Function type list

SBI Function List in both SBI v0.2 and v0.1

Function Type	Function Set	ID(v0.2)	ID (v0.1)
sbi_set_timer	Legacy	0x0000 0000	0
sbi_console_putchar	Legacy	0x0000 0001	1
sbi_console_getchar	Legacy	0x0000 0002	2
sbi_clear_ipi	Legacy	0x0000 0003	3
sbi_send_ipi	Legacy	0x0000 0004	4
sbi_remote_fence_i	Legacy	0x0000 0005	5
sbi_remote_sfence_vma	Legacy	0x0000 0006	6
sbi_remote_sfence_vma_asid	Legacy	0x0000 0007	7
sbi_shutdown	Legacy	0x0000 0008	8
---	---	---	---
sbi_get_spec_version	Base	0x0010 0001	-
sbi_set_sbiimp_version	Base	0x0010 0002	-
sbi_is_function_set	Base	0x0010 0003	-
sbi_is_function_type	Base	0x0010 0003	-
sbi_get_vendor_id	Base	0x0010 0004	-
sbi_get_mimp_id	Base	0x0010 0005	-
sbi_get_sbiimp_id	Base	0x0010 0006	-
---	---	---	---
sbi_set_timer	Exp-1	0x0100 0000	-
sbi_console_putchar	Exp-1	0x0100 0001	-
sbi_console_getchar	Exp-1	0x0100 0002	-
sbi_clear_ipi	Exp-1	0x0100 0003	-
sbi_send_ipi	Exp-1	0x0100 0004	-
sbi_remote_fence_i	Exp-1	0x0100 0005	-
sbi_remote_sfence_vma	Exp-1	0x0100 0006	-
sbi_remote_sfence_vma_asid	Exp-1	0x0100 0007	-

Error code table

Error Type	Value
SBI_SUCCESS	0
SBI_ERR_FAILURE	-1
SBI_ERR_NOT_SUPPORTED	-2
SBI_ERR_INVALID_PARAM	-3
SBI_ERR_DENIED	-4
SBI_ERR_INVALID_ADDRESS	-5

OpenSBI usage constraints

- With libsbi.a, firmware has to provide the platform specific hooks
- RISC-V *MSCRATCH* CSR must point to a valid OpenSBI scratch space
- RISC-V *SP* register (i.e. stack pointer) must be set per-HART pointing to distinct non-overlapping stacks
- Only calls two functions
  - sbi_init – gets called when hart boots up
  - *sbi_trap_handler* – Forward all traps and interrupts or at least for the following
    - M-mode timer interrupt
    - M-mode software interrupt
    - Illegal instruction trap
    - Misaligned load trap
    - Misaligned store trap
    - Supervisor ecall trap
    - Hypervisor ecall trap