Western Digital.

RISC-V SBI vs ARM PSCI

Comparison between SBI v0.2 and PSCI v1.1

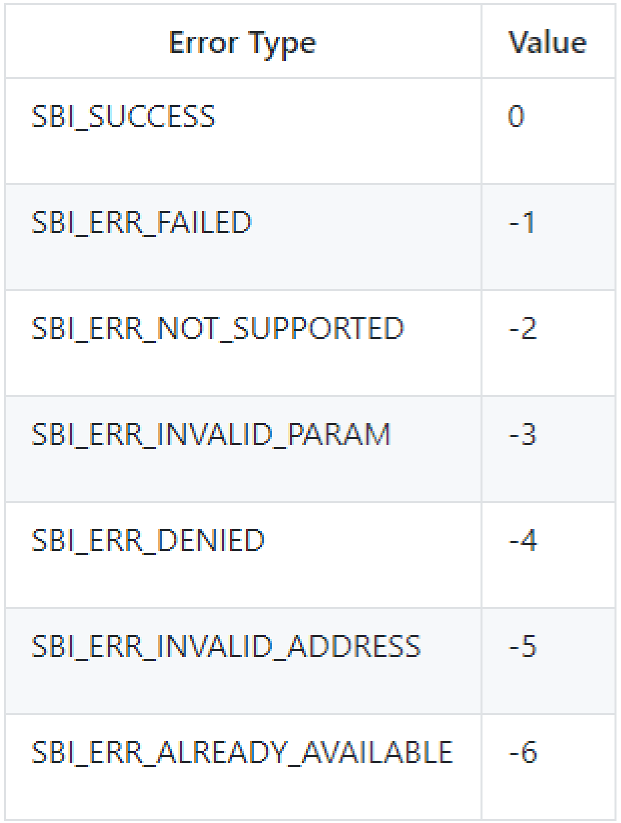
Anup Patel <anup.patel@wdc.com> and Atish Patra <atish.patra@wdc.com>

Western Digital System Software Research

RISC-V SBI v0.2 Calling Convention

Low-level details of SBI calls

- An “ecall” instruction executed in S-mode (or VS-mode) is treated as SBI call
- SBI call parameters:
  - A0 – Function Parameter0
  - A1 – Function Parameter1
  - A2 – Function Parameter2
  - A3 – Function Parameter3
  - A4 – Function Parameter4
  - A5 – Function Parameter5
  - A6 – Function ID
  - A7 – Extension ID
- SBI call return value:
  - A0 – Error Code
  - A1 – Value



Error Type	Value
SBI_SUCCESS	0
SBI_ERR_FAILED	-1
SBI_ERR_NOT_SUPPORTED	-2
SBI_ERR_INVALID_PARAM	-3
SBI_ERR_DENIED	-4
SBI_ERR_INVALID_ADDRESS	-5
SBI_ERR_ALREADY_AVAILABLE	-6

RISC-V SBI v0.2 Functions

List of functions in latest RISC-V SBI

Function	Extension Name	Extension ID	Function ID	Mandatory	SBI version	Note
sbi_get_sbi_spec_version	BASE	0x10	0x0	Mandatory	0.2	Used by Linux
sbi_get_sbi_impl_id			0x1	Mandatory	0.2	Used by Linux
sbi_get_sbi_impl_version			0x2	Mandatory	0.2	Used by Linux
sbi_probe_extension			0x3	Mandatory	0.2	Used by Linux
sbi_get_mvendorid			0x4	Mandatory	0.2	Not used by Linux
sbi_get_marchid			0x5	Mandatory	0.2	Not used by Linux
sbi_get_mimpid			0x6	Mandatory	0.2	Not used by Linux
sbi_set_timer	TIMER	0x54494D45	0x0	Optional	0.2	Used by Linux
sbi_send_ipi	IPI	0x735049	0x0	Optional	0.2	Used by Linux
sbi_remote_fence_i	RFENCE	0x52464E43	0x0	Optional	0.2	Used by Linux
sbi_remote_sfence_vma			0x1	Optional	0.2	Used by Linux
sbi_remote_sfence_vma_asid			0x2	Optional	0.2	Used by Linux
sbi_remote_hfence_gvma_vmid			0x3	Optional	0.2	Used by KVM, Xvisor
sbi_remote_hfence_gvma			0x4	Optional	0.2	Used by KVM, Xvisor
sbi_remote_hfence_vvma_asid			0x5	Optional	0.2	Used by KVM, Xvisor
sbi_remote_hfence_vvma			0x6	Optional	0.2	Used by KVM, Xvisor
sbi_hart_start	HSM	0x48534D	0x0	Optional	0.2	Used by Linux
sbi_hart_stop			0x1	Optional	0.2	Used by Linux
sbi_hart_get_status			0x2	Optional	0.2	Used by Linux
sbi_legacy_<xyz>	LEGACY	0x00-0x0F	0x0	Optional	0.1	Used by older Linux kernel lacking SBI v0.2 support
sbi_experimental_<xyz>	EXPERIMENTAL	0x08000000 - 0x08FFFFFF	---	Optional	0.2	Not used by Linux
sbi_vendor_<xyz>	VENDOR	0x09000000 - 0x09FFFFFF	---	Optional	0.2	Not used by Linux
sbi_firmware_<xyz>	FIRMWARE	0x0A000000 - 0x0AFFFFFF	---	Optional	0.2	Not used by Linux

RISC-V SBI v0.2 Highlights

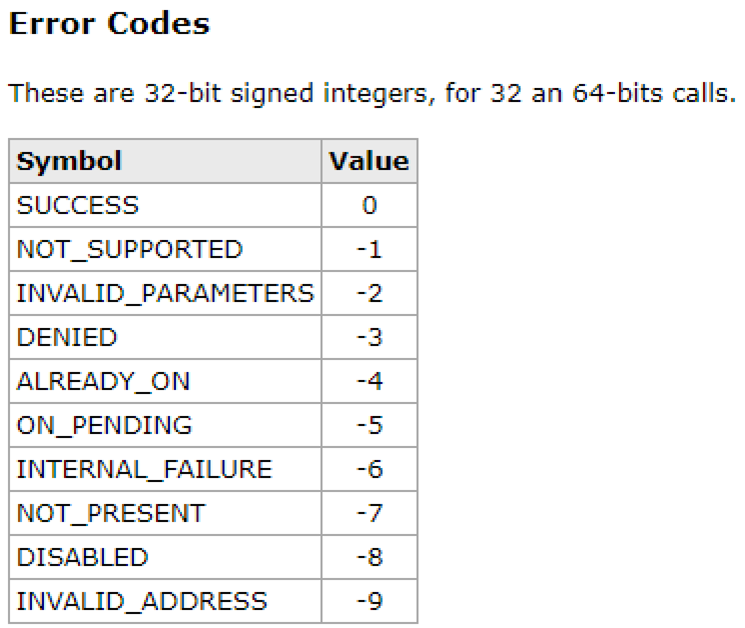
Salient features of SBI v0.2 specification

- SBI v0.2 is based on same principles of minimalism and modularity as RISC-V ISA
- Only the SBI v0.2 BASE extension is mandatory
- All optional SBI v0.2 extensions are discoverable using SBI v0.2 BASE extension
- Legacy SBI v0.1 calls supported as optional SBI v0.2 extensions
- Provision for EXPERIMENTAL and VENDOR extensions
- All SBI calls are designed to work for both RV32 and RV64
- New SBI extensions only defined when required
- New SBI extensions/calls are thoroughly reviewed in public forum

ARM PSCI Calling Convention

Low-level details of PSCI calls

- A “smc” instruction from EL2 and “smc/hvc” instruction from EL1 is treated as PSCI call
- PSCI call parameters:
  - X0/W0 – Function ID
  - X1/W1 – Function Parameter0
  - X2/W2 – Function Parameter1
  - X3/W3 – Function Parameter2
- PSCI call return values:
  - X0/W0 – Error Code
- A PSCI call is characterized by 32bit Function ID:
  - Type 0x8400XXXX for SMC32/HVC32 functions (ARM32 caller)
  - Type 0xC400XXXX for SMC64/HVC64 functions (ARM64 caller)



Error Codes

These are 32-bit signed integers, for 32 an 64-bits calls.

Symbol	Value
SUCCESS	0
NOT_SUPPORTED	-1
INVALID_PARAMETERS	-2
DENIED	-3
ALREADY_ON	-4
ON_PENDING	-5
INTERNAL_FAILURE	-6
NOT_PRESENT	-7
DISABLED	-8
INVALID_ADDRESS	-9

ARM PSCI v1.1 Function

List of functions in latest ARM PSCI

Function	SMC32 ID	SMC64 ID	Mandatory	PSCI version	Affinity Level Parameter*	Note
PSCI_VERSION	0x84000000	NA	Mandatory	0.2	No	Used by Linux
CPU_SUSPEND	0x84000001	0xC4000001	Mandatory	0.1	No	Used by Linux
CPU_OFF	0x84000002	NA	Mandatory	0.1	No	Used by Linux
CPU_ON	0x84000003	0xC4000003	Mandatory	0.1	Yes*	Used by Linux
AFFINITY_INFO	0x84000004	0xC4000004	Mandatory	0.2	Yes*	Used by Linux
MIGRATE	0x84000005	0xC4000005	Optional	0.1	Yes*	Not used by Linux
MIGRATE_INFO_TYPE	0x84000006	NA	Optional	0.2	No	Mandatory when MIGRATE is implemented. Not used by Linux.
MIGRATE_INFO_UP_CPU	0x84000007	0xC4000007	Optional	0.2	No	Not used by Linux
SYSTEM_OFF	0x84000008	NA	Mandatory	0.2	No	Used by Linux
SYSTEM_RESET	0x84000009	NA	Mandatory	0.2	No	Used by Linux
PSCI_FEATURES	0x8400000A	NA	Mandatory	1.0	No	Used by Linux
CPU_FREEZE	0x8400000B	NA	Optional	1.0	No	Not used by Linux
CPU_DEFAULT_SUSPEND	0x8400000C	0xC400000C	Optional	1.0	No	Not used by Linux
NODE_HW_STATE	0x8400000D	0xC400000D	Optional	1.0	Yes*	Not used by Linux
SYSTEM_SUSPEND	0x8400000E	0xC400000E	Optional	1.0	No	Used by Linux
PSCI_SET_SUSPEND_MODE	0x8400000F	NA	Optional	1.0	No	Used by Linux
PSCI_STAT_RESIDENCY	0x84000010	0xC4000010	Optional	1.0	Yes*	If either one is implemented, the other must be too Note used by Linux
PSCI_STAT_COUNT	0x84000011	0xC4000011	Optional	1.0	Yes*	
SYSTEM_RESET2	0x84000012	0xC4000012	Optional	1.1	No	Used by Linux
MEM_PROTECT	0x84000013	NA	Optional	1.1	No	Not used by Linux
MEM_PROTECT_CHECK_RANGE	0x84000014	0xC4000014	Optional	1.1	No	Not used by Linux

* ARM specification defines MPIDR_EL1 MSR which is physical ID of a CPU having multiple affinity levels (core, cluster, system)

ARM PSCI v1.1 Difficulties

Some odd or peculiar specifications due to organic growth

- Too many combinations ARM PSCI power states (core, cluster, and system) because PSCI CPU_ON and AFFINITY_INFO calls takes ARM affinity level as parameter
  - Linux cpuidle only does CPU-level power states changes and Cluster-level power state changes are done implicitly by the platform firmware
  - Default strategy for Cluster-level power states is Platform-coordinated mode (ARM affinity level based)
- Few ARM PSCI calls were added to tackle missing features:
  - Three different ARM PSCI calls for system power-off (SYSTEM_OFF), cold reboot (SYSTEM_RESET), and warm reboot (SYSTEM_RESET2) which could have been a single ARM PSCI call
  - The CPU_DEFAULT_SUSPEND call was added because original CPU_SUSPEND call lacked the notion of "platform default" CPU power state but is not used by Linux
  - The PSCI_SET_SUSPEND_MODE call was added to allow changing the strategy for Cluster-level power state changes so that it is more suitable for Linux power-management
  - The CPU_FREEZE was added because original CPU_SUSPEND and CPU_OFF calls did not have provision power-down CPU without affecting CPU interrupts
- ARM PSCI MIGRATE related calls exist since PSCI v0.1 but never used by OP-TEE

RISC-V SBI and ARM PSCI Functional Overlap

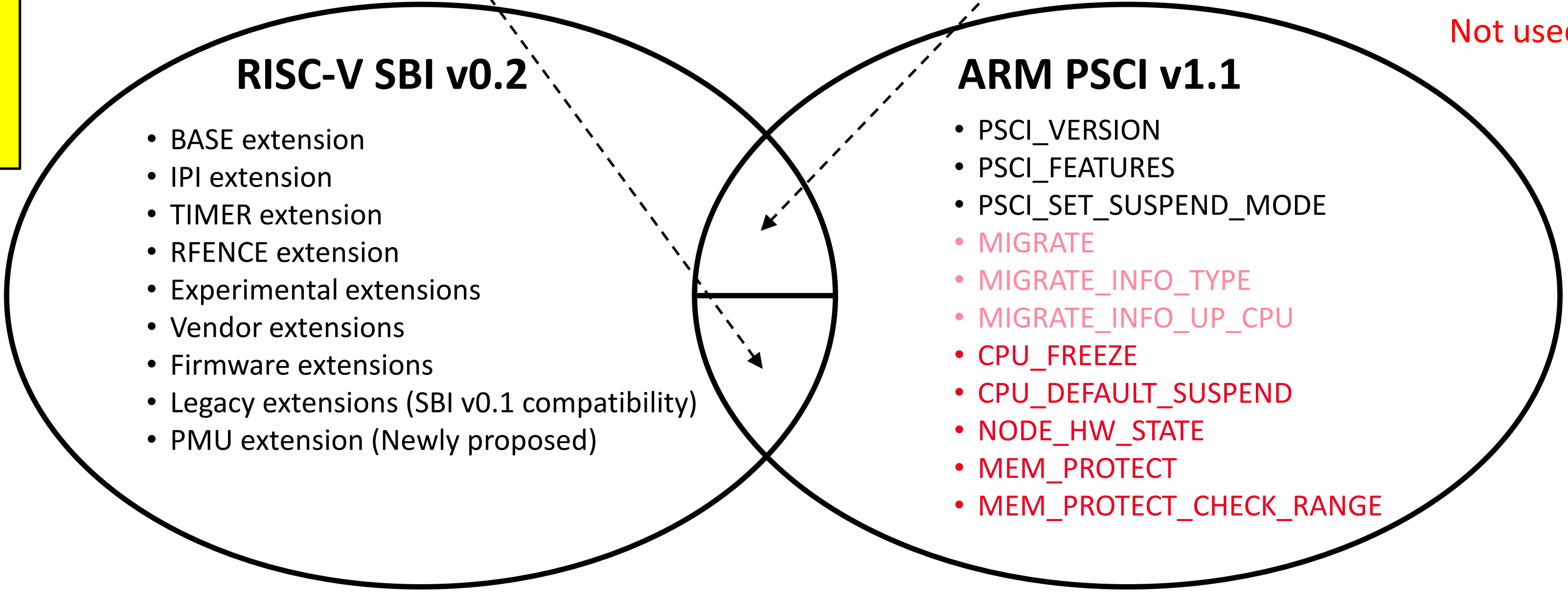
Functional overlap between SBI v0.2 and PSCI v1.1

Overlapping PSCI Calls covered by Existing SBI calls
- CPU_ON (sbi_hsm_start)
- CPU_OFF (sbi_hsm_stop)
- AFFINITY_INFO (sbi_hsm_get_status)

Already available in:
- OpenSBI
- Linux kernel upstream
- EDK2
- KVM
- Xvisor

Overlapping PSCI Calls covered by Proposed/Future SBI calls
- SYSTEM_OFF (sbi_srst_reset(0)) (Linux and OpenSBI Source Available)
- SYSTEM_RESET (sbi_srst_reset(1)) (Linux and OpenSBI Source Available)
- SYSTEM_RESET2 (sbi_srst_reset(2)) (Linux and OpenSBI Source Available)
- PSCI_STAT_COUNT (will be covered by SBI PMU software events)
- **CPU_SUSPEND (missing in SBI HSM HART SUSPEND call)**
- **SYSTEM_SUSPEND (missing in SBI SSPD SUSPEND call)**

Not used by OP-TEE
Not used by Linux



RISC-V SBI v0.2
- BASE extension
- IPI extension
- TIMER extension
- RFENCE extension
- Experimental extensions
- Vendor extensions
- Firmware extensions
- Legacy extensions (SBI v0.1 compatibility)
- PMU extension (Newly proposed)

ARM PSCI v1.1
- PSCI_VERSION
- PSCI_FEATURES
- PSCI_SET_SUSPEND_MODE
- MIGRATE
- MIGRATE_INFO_TYPE
- MIGRATE_INFO_UP_CPU
- CPU_FREEZE
- CPU_DEFAULT_SUSPEND
- NODE_HW_STATE
- MEM_PROTECT
- MEM_PROTECT_CHECK_RANGE

RISC-V SBI and ARM PSCI Functional Overlap

Only few more power management calls needed in RISC-V SBI

Function	SMC32 ID	SMC64 ID	Mandatory	PSCI version	Affinity Level Parameter*	Note
CPU_SUSPEND	0x84000001	0xC4000001	Mandatory	0.1	No	SUSPEND call in SBI HSM extension can be used
SYSTEM_SUSPEND	0x8400000E	0xC400000E	Optional	1.0	No	SYSTEM SUSPEND extension needs to be defined
SYSTEM_OFF	0x84000008	NA	Mandatory	0.2	No	Handled by SBI SRST extension
SYSTEM_RESET	0x84000009	NA	Mandatory	0.2	No	Handled by SBI SRST extension
SYSTEM_RESET2	0x84000012	0xC4000012	Optional	1.1	No	Handled by SBI SRST extension

- SBI SRST extension is already available for system shutdown and reset
  - Draft specification (https://github.com/riscv/riscv-sbi-doc/pull/39)
  - Linux implementation
    http://lists.infradead.org/pipermail/linux-riscv/2020-September/002237.html
  - OpenSBI implementation
    http://lists.infradead.org/pipermail/opensbi/2020-October/000265.html
- SBI HSM HART suspend call & SBI SYSTEM SUSPEND extension can be added without much effort

RISC-V SBI HSM HART SUSPEND Efforts

Efforts required for RISC-V SBI HSM HART SUSPEND Call

- Only three HART-level power states: STARTED (ON), STOPPED (Power-down), and SUSPENDED
  - The STARTED and STOPPED states are already defined and used
  - New SUSPENDED state will be a natural extension to existing states
- The Cluster-level power states changes will be platform specific
  - RISC-V SBI specification can provide an example strategy for Cluster-level power state changes
- The “FREEZE” state can be a special power-state for SBI HSM HART SUSPEND call
  - This special power-state will allow HW debuggers to work even after HART is power-down
- **Efforts:**
  - Add HART SUSPEND call to SBI HSM extension specification
  - Implement SBI HSM HART SUSPEND call in OpenSBI
  - Emulate SBI HSM HART SUSPEND call in KVM/Xvisor (similar to WFI emulation)
  - Re-use and extend the CPUILDE driver send-out by Allwinner on LKML

RISC-V SBI SSPD SUSPEND Efforts

Efforts required for RISC-V SBI SSPD SUSPEND Call

- The SBI SSPD SUSPEND call will only work when all HARTs except the calling HART are in STOPPED (Power-down) state.
- Wake-up from SBI SSPD SUSPEND call on the calling HART is very similar to waking up from SBI HSM HART SUSPEND call
- **Efforts:**
  - Add SBI SSPD extension in SBI specification with just one SUSPEND call
  - Implement SBI SSPD SUSPEND call in OpenSBI
  - Emulate SBI SSPD SUSPEND call in KVM/Xvisor
    - Similar to SBI HSM HART SUSPEND emulation but with additional constraints
  - Detect and use SBI SSPD extension in Linux RISC-V SBI (arch/riscv/kernel/sbi.c) to provide "struct platform_suspend_ops"
    - Provide "struct platform_suspend_ops" in arch/riscv/kernel/sbi.c when SBI SSPD extension available
    - It will re-use suspend entry/exit code added in arch/riscv for SBI HSM HART SUSPEND
    - It will be small change similar to Linux patch send-out for SBI SRST extension

RISC-V SBI as a bridge to PSCI

Effort required for specification update

- RISC-V SBI PSCI extension would inherit all difficulties of ARM PSCI specification
- An open/community-driven RISC-V SBI specification would depend on proprietary ARM PSCI specification
  - PSCI specification changes are not discussed in a public forum
  - Future changes in ARM PSCI specification/software can impact RISC-V SBI specification/software
  - If ARM Ltd removes public access to ARM PSCI specification then we will end-up copying a proprietary ARM PSCI specification into the RISC-V SBI specification
- A RISC-V SBI PSCI extension would be needed in RISC-V SBI specification to define
  - How the RISC-V SBI calling convention maps to the ARM PSCI calling convention (and vice-versa)
  - How the ARM PSCI error codes are derived from RISC-V SBI error codes
  - The PSCI version and features to be provided
  - The set of PSCI calls which need to be provided
  - RISC-V HARTID to ARM MPIDR (and vice-versa) mapping as RISC-V does not define affinity levels in HART

RISC-V SBI as a bridge to PSCI

Effort required for implementation

- Development efforts in SBI implementations (OpenSBI/KVM/Xvisor) would be more comparable to dedicated SBI extensions (such as SRST, HART SUSPEND, etc)
  - An SBI PSCI extension with necessary: calling convention translation, error code translation, affinity level to HARTID translation, etc
  - Additional set of PSCI calls required would be: PSCI_VERSION, AFFINITY_INFO, PSCI_FEATURES for compatibility reasons
- Development efforts in arch/riscv for re-using PSCI CPU_SUSPEND and SYSTEM_SUSPEND will be more
  - drivers/firmware/psci.c has minimal code and majority of the code is in arch/arm64 and arch/arm
  - cpu_do_suspend() and cpu_do_resume() in arch/arm64/mm/proc.S
  - Complete arch/arm64/kernel/sleep.S
  - Complete arch/arm64/kernel/suspend.c
  - Majority of arch/arm64/kernel/cpuidle.c

Western Digital®