RISC-V "stimecmp / vstimecmp" Extension

Sstc

Version 0.5.4-3f9ed34, 2021-10-13: frozen

Table of Contents

Preamble

Document Source Information

See github.com/riscv/riscv-time-compare for document source. The document version includes the source control tag or commit hash used to produce this document.

Introduction

The current Privileged arch specification only defines a hardware mechanism for generating machine-mode timer interrupts (based on the mtime and mtimecmp registers). With the resultant requirement that timer services for S-mode/HS-mode (and for VS-mode) have to all be provided by M-mode - via SBI calls from S/HS-mode up to M-mode (or VS-mode calls to HS-mode and then to M-mode). M-mode software then multiplexes these multiple logical timers onto its one physical M-mode timer facility, and the M-mode timer interrupt handler passes timer interrupts back down to the appropriate lower privilege mode.

This extension serves to provide supervisor mode with its own CSR-based timer interrupt facility that it can directly manage to provide its own timer service (in the form of having its own stimecmp register) - thus eliminating the large overheads for emulating S/HS-mode timers and timer interrupt generation up in M-mode. Further, this extension adds a similar facility to the Hypervisor extension for VS-mode.

To make it easy to understand the deltas from the current Priv 1.11/1.12 specs, this is written as the actual exact changes to be made to existing paragraphs of Priv spec text (or additional paragraphs within the existing text).

The extension name is "Sstc" ('Ss' for Privileged arch and Supervisor-level extensions, and 'tc' for timecmp). This extension adds the S-level stimecmp CSR and the VS-level vstimecmp CSR.

Chapter 1. Machine and Supervisor Level Additions

1.1. Supervisor Timer Register (stimecmp)

This extension adds this new CSR.

The stimecmp CSR is a 64-bit register and has 64-bit precision on all RV32 and RV64 systems. In RV32 only, accesses to the stimecmp CSR access the low 32 bits, while accesses to the stimecmph CSR access the high 32 bits of stimecmp.

The CSR numbers for stimecmp / stimecmph are 0x14D / 0x15D (within the Supervisor Trap Setup block of CSRs).

A supervisor timer interrupt becomes pending - as reflected in the STIP bit in the mip and sip registers - whenever time contains a value greater than or equal to stimecmp, treating the values as unsigned integers. Writes to stimecmp are guaranteed to be reflected in STIP eventually, but not necessarily immediately. The interrupt remains posted until stimecmp becomes greater than time - typically as a result of writing stimecmp. The interrupt will be taken based on the standard interrupt enable and delegation rules.

Non-normative

A spurious timer interrupt might occur if an interrupt handler advances stimecmp then immediately returns, because STIP might not yet have fallen in the interim. All software should be written to assume this event is possible, but most software should assume this event is extremely unlikely. It is almost always more performant to incur an occasional spurious timer interrupt than to poll STIP until it falls.

Non-normative

In systems in which a supervisor execution environment (SEE) provides timer facilities via an SBI function call, this SBI call will continue to support requests to schedule a timer interrupt. The SEE will simply make use of stimecmp, changing its value as appropriate. This ensures compatibility with existing S-mode software that uses this SEE facility, while new S-mode software takes advantage of stimecmp directly.)

1.2. Machine Interrupt Registers (mip and mie)

This extension modifies the description of the STIP/STIE bits in these registers as follows:

If supervisor mode is implemented, its mip.STIP and mie.STIE are the interrupt-pending and interrupt-enable bits for supervisor-level timer interrupts. If the stimecmp register is not implemented, STIP is writable in mip, and may be written by M-mode software to deliver timer interrupts to S-mode. If the stimecmp (supervisor-mode timer compare) register is implemented, STIP is read-only in mip and reflects the supervisor-level timer interrupt signal resulting from stimecmp. This timer interrupt signal is cleared by writing stimecmp with a value greater than the current time value.

1.3. Supervisor Interrupt Registers (sip and sie)

This extension modifies the description of the STIP/STIE bits in these registers as follows:

Bits sip.STIP and sie.STIE are the interrupt-pending and interrupt-enable bits for supervisor level timer

interrupts. If implemented, STIP is read-only in sip, and is either set and cleared by the execution environment (if stimecmp is not implemented), or reflects the timer interrupt signal resulting from stimecmp (if stimecmp is implemented). The sip.STIP bit, in response to timer interrupts generated by stimecmp, is set and cleared by writing stimecmp with a value that respectively is less than or equal to, or greater than, the current time value.

1.4. Machine Counter-Enable Register (mcounteren)

This extension adds to the description of the TM bit in this register as follows:

In addition, when the TM bit in the mcounteren register is clear, attempts to access the stimecmp register while executing in S-mode will cause an illegal instruction exception. When this bit is set, access to the stimecmp register (if implemented) is permitted in S-mode.

Chapter 2. Hypervisor Extension Additions

2.1. **Virtual Supervisor Timer Register (vstimecmp)**

This extension adds this new CSR.

The vstimecmp CSR is a 64-bit register and has 64-bit precision on all RV32 and RV64 systems. In RV32 only, accesses to the vstimecmp CSR access the low 32 bits, while accesses to the vstimecmph CSR access the high 32 bits of vstimecmp.

The proposed CSR numbers for vstimecmp / vstimecmph are 0x24D / 0x25D (within the Virtual Supervisor Registers block of CSRs, and mirroring the CSR numbers for stimecmp/stimecmph).

A virtual supervisor timer interrupt becomes pending - as reflected in the VSTIP bit in the hip register - whenever (time + htimedelta), truncated to 64 bits, contains a value greater than or equal to vstimecmp, treating the values as unsigned integers. Writes to vstimecmp and htimedelta are guaranteed to be reflected in VSTIP eventually, but not necessarily immediately. The interrupt remains posted until vstimecmp becomes greater than (time + htimedelta) - typically as a result of writing vstimecmp. The interrupt will be taken based on the standard interrupt enable and delegation rules while V=1.

Non-normative

In systems in which a supervisor execution environment (SEE) implemented by an HS-mode hypervisor provides timer facilities via an SBI function call, this SBI call will continue to support requests to schedule a timer interrupt. The SEE will simply make use of vstimecmp, changing its value as appropriate. This ensures compatibility with existing guest VS-mode software that uses this SEE facility, while new VS-mode software takes advantage of vstimecmp directly.)

2.2. Hypervisor Interrupt Registers (hvip, hip, and hie)

This extension modifies the description of the VSTIP/VSTIE bits in the hip/hie registers as follows:

Bits hip.VSTIP and hie.VSTIE are the interrupt-pending and interrupt-enable bits for VS-level timer interrupts. VSTIP is read-only in hip, and is the logical-OR of hvip.VSTIP and the timer interrupt signal resulting from vstimecmp (if vstimecmp is implemented). The hip.VSTIP bit, in response to timer interrupts generated by vstimecmp, is set and cleared by writing vstimecmp with a value that respectively is less than or equal to, or greater than, the current (time + htimedelta) value. The hip.VSTIP bit remains defined while V=0 as well as V=1.

2.3. Hypervisor Counter-Enable Register (hcounteren)

This extension adds to the description of the TM bit in this register as follows:

In addition, when the TM bit in the hcounteren register is clear, attempts to access the vstimecmp register (via stimecmp) while executing in VS-mode will cause a virtual instruction exception if the same bit in mcounteren is 1. When this bit is set, access to the vstimecmp register (if implemented) is permitted in VS-mode.

Chapter 3. Environment Config (menvcfg/henvcfg) Support

Enable/disable bits for this extension are provided in the new menvcfg / henvcfg CSRs.

Bit 63 of menvcfg (or bit 31 of menvcfgh) - named STCE (STimecmp Enable) - enables stimecmp for S-mode when set to one, and the same bit of henvcfg enables vstimecmp for VS-mode. These STCE bits are WARL and are hard-wired to 0 when this extension is not implemented.

When STCE in menvcfg is zero, an attempt to access stimecmp or vstimecmp in a mode other than M-mode raises an illegal instruction exception, STCE in henvcfg is read-only zero, and STIP in mip and sip reverts to its defined behavior as if this extension is not implemented.

When STCE in menvcfg is one but STCE in henvcfg is zero, an attempt to access stimecmp (really vstimecmp) when $V = 1$ raises a virtual instruction exception, and VSTIP in hip reverts to its defined behavior as if this extension is not implemented.

Index

Bibliography


RISC-V