VciXicu

1) Functional Description

This VCI target is a memory mapped peripheral implementing a vectorized interrupt controller, a timer controller, and an Inter-processor interrupt controller: It is an interrupt hub, concentrating 3 types of interrupts:

- up to 32 internal programmable timer interrupts (PTI),
- up to 32 external hardware interrupt lines (HWI),
- up to 32 internal software-triggered interrupts (WTI).

All these interrupt sources can be routed to up to 32 interrupt outputs. Each output can mask individual interrupt sources. Priority between interrupt source types is left to the handling operating system. Priority of interrupts inside an interrupt source type is from the lowest index(highest priority) to the highest index (lower priority).

1.1) Constructor Parameters

All hardware implementations of this component may not implement all the up-to-32 PTI (Timers), up-to-32 HWI lines, up-to-32 WTI registers and up-to-32 OUTPUTlines. The following parameters allow the system designer to get just the needed hardware.

- pti count (in range 0..32): number of programmable timers
- hwi count (in range 0..32): number of external hardware interrupt lines
- wti count (in range 0..32): number of write-triggered interrupt sources
- irqcount (in range 1..32): number of output interrupt lines

1.2) Programmers's View

This component can be mapped anywhere in the address space, on a 4-KBytes boundary. This component is 32-bit data-word based: arbitrary byte access is not supported. The 12 lower address bits are used the following way:

FUNC	INDEX	00
5 bits	5bits	

- **FUNC** indicates the functionnality.
- **INDEX** can be either an input index, or an output index, depending on the functionnality.

MODE	Register	FUNC	INDEX
R/W	XICU_WTI_REG	00000	WTI_INDEX
R/W	XICU_PTI_PER	00001	PTI_INDEX
R/W	XICU_PTI_VAL	00010	PTI_INDEX
W	X_ICUPTI_ACK	00011	PTI_INDEX
R/W	XICU_MSK_PTI	00100	0UT_INDEX
W	XICU_MSK_PTI_ENABLE	00101	OUT_INDEX
W	XICU_MSK_PTI_DISABLE	00110	OUT_INDEX
R	XICU_PTI_ACTIVE	00110	OUT_INDEX
	Reserved	00111	

R/W	XICU_MSK_HWI	01000	OUT_INDEX
W	XICU_MSK_HWI_ENABLE	01001	OUT_INDEX
W	XICU_MSK_HWI_DISABLE	01010	OUT_INDEX
R	XICU_HWI_ACTIVE	01010	OUT_INDEX
	Reserved	01011	
R/W	XICU_MSK_WTI	01100	OUT_INDEX
W	XICU_MSK_WTI_ENABLE	01101	OUT_INDEX
W	XICU_MSK_WTI_DISABLE	01110	OUT_INDEX
R	XICU_WTI_ACTIVE	01110	OUT_INDEX
R	XICU_PRIO	01111	OUT_INDEX
R	XICU_CONFIG	10000	unused

Software can use the following macro to access registers:

```
#define XICU_REG(func, index) (((func)<<5)|(index))
```

WTI_REG[WTI_INDEX] : Write-Triggered Interrupt Register

This register retains the value written. It can be used as a mailbox between the software interrupt source and the target if there is only one source. In case of several sources, two different sources may write sequentially to this register, overwriting the value present in register.

- On write : Raises WTI[WTI_INDEX]
- On read : Acknowledges WTI[WTI_INDEX]

PTI_PER[PTI_INDEX] : Programmable Timer Period Register

This register contains the reset value for TIMER[PTI_INDEX] when it wraps to 0. If this register is set to 0, the corresponding timer is disabled and no interrupt is ever raised. If there is a pending interrupt, it is cleared, without need to read PTI_ACK[PTI_INDEX]. If the written value is non-zero, and the timer is currently running, the corresponding timer counter is not reset.

- On write : Resets the period of TIMER[PTI_INDEX].
- On read : Gets the period of TIMER[PTI_INDEX].

PTI_VAL[PTI_INDEX] : Programmable Timer Value Register

This register is decremented by 1 on each clock's raising edge. When it gets to 0, the value is reset to the corresponding period register value (PTI_PER[PTI?INDEX]), and the corresponding timer interrupt line is asserted until acknowledged. Decrementation goes on whether interrupt is acknowledged or not.

- On write : Resets the current value of TIMER[PTI_INDEX]. Writing a value greater than PTI_PER[PTI_INDEX] in this register has no particular side-effect: value will normally decrement to 0 and then be reset to PTI_PER[PTI_INDEX] when wrapping.
- On read : Gets the current value of TIMER[PTI_INDEX].

PTI_ACK[PTI_INDEX] : Programmable Timer Acknowledge Register

This register is used by the software to deassert an interrupt raised by wrapping of the PTI_VAL[PTI_INDEX] register.

- On write : Unsupported

- On read : Acknowledges the interrupt associated to TIMER[PTI_INDEX]. Read value has no useful meaning.

MSK_PTI[OUT_INDEX] : Programmable Timer Mask for IRQ[OUT_INDEX]

Each bit in this register is a mask for the corresponding timer IRQ. A 1 in bit x enables the timer x as an interrupt source for IRQ[OUT_INDEX].

- On write : Sets the current mask
- On read : Gets the current mask

MSK_PTI_ENABLE[OUT_INDEX] : Programmable Timer Mask Enabler for IRQ[OUT_INDEX]

Each bit written here unmasks the corresponding timer IRQ. Writing a 1 in bit x enables the timer x as an interrupt source for IRQ[OUT_INDEX].

- On write : ORs MSK_PTI[OUT_INDEX] with the written value.
- On read : Unsupported

MSK_PTI_DISABLE[OUT_INDEX] : Programmable Timer Mask Disabler for IRQ[OUT_INDEX]

Each bit written here masks the corresponding timer IRQ. Writing a 1 in bit x disables the timer x as an interrupt source for IRQ[OUT_INDEX].

- On write : ANDs MSK_PTI[OUT_INDEX] with the complement of the written value.
- On read : Unsupported

PTI_ACTIVE[OUT_INDEX] : Current Unmasked Timer IRQs for IRQ[OUT_INDEX]

Each bit read here corresponds to an active and unmasked timer IRQ, according to current global timer status and MSK_PTI[OUT_INDEX].

- On write : Unsupported
- On read : Gets the current timer status for IRQ[OUT_INDEX].

MSK_HWI[OUT_INDEX] : Hardware Interrupts Mask for IRQ[OUT_INDEX]

Each bit in this register is a mask for the corresponding hardware interrupt. A 1 in bit x enables the hardware interrupt x as an interrupt source for IRQ[OUT_INDEX].

- On write : Sets the current mask
- On read : Gets the current mask

MSK_HWI_ENABLE[OUT_INDEX] : Hardware Interrupt Mask Enabler for IRQ[OUT_INDEX]

Each bit written here unmasks the corresponding hardware interrupt. Writing a 1 in bit x enables the hardware interrupt x as an interrupt source for IRQ[OUT_INDEX].

- On write : ORs MSK_PTI[OUT_INDEX] with the written value.
- On read : Unsupported

MSK_HWI_DISABLE[OUT_INDEX] : Hardware Interrupt Mask Disabler for IRQ[OUT_INDEX]

Each bit written here masks the corresponding hardware interrupt. Writing a 1 in bit x disables the hardware interrupt x as an interrupt source for IRQ[OUT_INDEX].

- On write : ANDs MSK_PTI[OUT_INDEX] with the complement of the written value.
- On read : Unsupported

HWI_ACTIVE[OUT_INDEX] : Current Unmasked Hardware Interrupts for IRQ[OUT_INDEX]

Each bit read here corresponds to an active and unmasked hardware interrupt, according to current global hardware interrupt status and MSK_HWI[OUT_INDEX].

- On write : Unsupported
- On read : Gets the current hardware interrupts status for IRQ[OUT_INDEX].

MSK_WTI[OUT_INDEX] : Write-Triggered Interrupts Mask for IRQ[OUT_INDEX]

Each bit in this register is a mask for the corresponding software interrupt. A 1 in bit x enables the software interrupt x as an interrupt source for IRQ[OUT_INDEX].

- On write : Sets the current mask
- On read : Gets the current mask

MSK_WTI_ENABLE[OUT_INDEX] : Write-Triggered Interrupt Mask Enabler for IRQ[OUT_INDEX]

Each bit written here unmasks the corresponding software interrupt. Writing a 1 in bit x enables the software interrupt x as an interrupt source for IRQ[OUT_INDEX].

- On write : ORs MSK_PTI[OUT_INDEX] with the written value.
- On read : Unsupported

MSK_WTI_DISABLE[OUT_INDEX] : Write-Triggered Interrupt Mask Disabler for IRQ[OUT_INDEX]

Each bit written here masks the corresponding software interrupt. Writing a 1 in bit x disables the software interrupt x as an interrupt source for IRQ[OUT_INDEX].

- On write : ANDs MSK_PTI[OUT_INDEX] with the complement of the written value.
- On read : Unsupported

WTI_ACTIVE[OUT_INDEX] : Current Unmasked Write-Triggered Interrupts for IRQ[OUT_INDEX]

Each bit read here corresponds to an active and unmasked software interrupt, according to current global software interrupt status and MSK_WTI[OUT_INDEX].

- On write : Unsupported
- On read : Gets the current software interrupts status for IRQ[OUT_INDEX].

PRIO[OUT_INDEX] : Priority Encoder for IRQ[OUT_INDEX]

This read-only register holds the index of the highest priority, active and unmasked interrupt for each source type.

- On write : Unsupported
- On read : Get the three highest-priority active interrupt indexes for IRQ[OUT_INDEX].

000 PRIO_WTI 000 PRIO_HWI 000 PRIO_PTI 00000 W H T

- Bit T is set if there is at least one unmasked timer interrupt.
- Bit H is set if there is at least one unmasked hardware interrupt.
- Bit W is set if there is at least one unmasked software interrupt.
- Field PRIO_PTI (5 bits) contains the index of the highest priority timer interrupt.
- Field PRIO_HWI (5 bits) contains the index of the highest priority hardware interrupt.
- Field PRIO_WTI (5 bits) contains the index of the highest priority software interrupt.

CONFIG : Global XICU configuration register

This read-only register return the XICU hardware parameters.

- On write : Unsupported
- On read : returns the IRQ_COUNT, WTI_COUNT, HWI_COUNT, PTI_COUNT values. Each value (between 0 & 32) coded on 6 bits

00IRQ_COUNT 00WTI_COUNT 00HWI_COUNT 00PTI_COUNT

- IRQ_COUNT : total number of output IRQs.
- WTI_COUNT : total number of software triggered interrupts.
- HWI_COUNT : total number of hardware interrupts.
- PTI_COUNT : total number of programmable timer interrupts.

Complete specification is in xicu-1.0.pdf.

2) Component definition & usage

source:trunk/soclib/module/infrastructure_component/interrupt_infrastructure/vci_xicu/caba/metadata/vci_xicu.sd

```
Uses( 'vci_xicu' )
```

3) CABA Implementation

CABA sources

- interface : source:trunk/soclib/soclib/module/infrastructure_component/interrupt_infrastructure/vci_xicu/caba/source/include/vc
- implementation : source:trunk/soclib/soclib/module/infrastructure_component/interrupt_infrastructure/vci_xicu/caba/source/src/vci_xic

CABA Constructor parameters

```
VciXicu(
     sc_module_name name,  // Component Name
     const soclib::common::MappingTable &mt,  // Mapping Table
     const soclib::common::InTab &index, // Target index
     size_t pti_count,   // Number of programmable timers
     size_t hwi_count,   // Number of hardware interrupt lines
     size_t wti_count,   // Number of write-triggerred interrupts (IPI)
     size_t irq_count);  // Number of output lines
```

CABA Ports

- sc_in<bool> **p_clk** : Global system clock
- sc_in<bool> **p_resetn** : Global system reset
- soclib::caba::VciTarget<vci_param> **p_vci** : VCI port
- sc_out<bool> ***p_irq** : Output interrupt ports (irq_count)
- sc_in<bool> ***p_hwi** : Input interrupts ports (hwi_count)

4) TLM-DT Implementation

TLM-DT sources

- interface :
  source:trunk/soclib/soclib/module/infrastructure_component/interrupt_infrastructure/vci_xicu/tlmdt/source/include/vc
- implementation :
  source:trunk/soclib/soclib/module/infrastructure_component/interrupt_infrastructure/vci_xicu/tlmdt/source/src/vci_xi

TLM-DT Constructor parameters

```
VciXicu(
     sc_module_name name,  // Component Name
     const soclib::common::InTab &index, // Target index
     const soclib::common::MappingTable &mt,  // Mapping Table
     size_t pti_count,   // Number of programmeble timers
     size_t hwi_count,   // Number of hardware interrupt lines
     size_t wti_count,   // Number of write-triggerred interrupts (IPI)
     size_t irq_count);  // Number of output lines
```

TLM-DT Ports

- **p_vci** : VCI target port
- **p_irq[irq_count]** : Output interrupt ports
- **p_hwi[hwi_count]** : Input interrupts ports