Functional Description

This VCI target is a memory mapped peripheral implementing a vectorized interrupt controller. It can concentrate up to 32 independent interrupt lines **p_irq_in[i]** to a single **p_irq** interrupt line. The active state is high, and the output interrupt is the logical OR of all input interrupts. Each input interrupt can be individually masked using a memory mapped 32 bits register. This component can be addresse to return the index of the highest priority active interrupt **p_irq_[i]**. The priority scheme is fixed : The lower indexes have the highest priority. The memory segment allocated to this component must be aligned on 8 bytes boundary. This hardware component cheks for segmentation violation, and can be used as a default target.

This component contains 2 memory mapped registers:

- **ICU_INDEX** : ADDRESS[2:0] = 0x0

It is actually a 32 bits pseudo-register: A read request returns the index value of the highest priority active input interrupt. If there is no active interrupt, it returns the 32 value. This register is read-only.

- **ICU_MASK** : ADDRESS[2:0] = 0x4

Each bit i of this 32 bits register enables the corresponding **p_irq_in[i]** input interrupt, when this bit is one. This register can be read or written.A write request of a zero gives resets this register. This register is write-only.

CABA Implementation

The caba implementation is in

- source:trunk/soclib/systemc/include/caba/target/vci_icu.h
- source:trunk/soclib/systemc/src/caba/target/vci_icu.cc

Template parameters:

- The VCI parameters

Constructor parameters

```
VciIcu(
     sc_module_name name,  // Component Name
     const soclib::common::InTab &index, // Target index
     const soclib::common::MappingTable &mt,  // Mapping Table
     size_t nirq);  // Number of input interrupts
```

Ports

- sc_in<bool> p_resetn : Global system reset
- sc_in<bool> p_clk : Global system clock
- soclib::caba::VciTarget<vci_param> p_vci : VCI port
- sc_out<bool> p_irq : Output interrupt port
- sc_in<bool> *p_irq_in : pointer to the input interrupts ports table