Mips Processor Functional Description

This hardware component is a Mips R3000 processor core. This uses the generic VciXcache? component to interface a VCI advanced interconnect.

The simulation model is actually an instruction set simulator (ISS), organised as a two-stage pipeline:

- TIn the first cycle, the instruction fetch, with access to the external instruction cache.
- In the second cycle, the instruction is executed with a possible access to the external data cache.
- The "delayed branch" is accurately modelized, but not the "delayed load".

The main functional specifications are the following:

- The floating point instructions are not supported
- There is no TLB : no hardware support for virtual memory
- All Mips R3000 exceptions are handled, including the memory addressing X_IBE and X_DBE, but the write errors are not precise, due to the posted write buffer in the cache controller.
- A data cache line invalidation mechanism is supported : when a *lw* instruction is executed with the R0 destination register, a cache line invalidation request is sent to the data cache.

Mips Processor CABA Implementation

The caba implementation is in

- source:trunk/soclib/systemc/include/caba/processor/Mips.h
- source:trunk/soclib/systemc/src/caba/processor/Mips.cc

Template parameters

This component has no template parameters.

Constructor parameters

```
Mips(
     sc_module_name name,   //  Instance Name
     int  ident);   // processor id
```

Visible registers

The following internal registers define the processor internal state, and can be inspected:

- PC : program counter
- IR : Instruction register
- GR[i] : General registers (0 < i < 32)
- HI & LO : intermediate registers for multiply / divide instructions
- IDENT : processor id register = CP0[0]
- BAR : Bad address register = CP0[8]
- SR : Status register = CP0[12]
- CR : Cause register = CP0[13]

- EPC : Exception PC register = CP0[14]

Ports

- sc_in<bool> **p_resetn** : Global system reset
- sc_in<bool> **p_clk** : Global system clock
- sc_in<bool> ***p_irq[6]** : The six interrupt requests
- soclib::caba::IcacheProcesssorPort **p_icache** : Instruction cache interface to the VciXcache
- soclib::caba::DcacheProcesssorPort **p_dcache** : Data cache interface to the VciXcache