Mips Processor Functional Description

This hardware component is a Mips R3000 processor core. This is only an ISS, which should be wrapped with an IssWrapper.

The simulation model is actually an instruction set simulator, organised as a three-stage pipeline:

- First stage: instruction fetch, with access to the external instruction cache.
- Second stage: instruction is executed with a possible access to the external data cache.
- Third stage: read memory access is written back to registers

The main functional specifications are the following:

- LWL/LWR instructions are not yet implemented even if patent expired on 2006-12-23 (?source)
- The floating point instructions are not supported
- There is no TLB, and 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 GPR[0] destination register, a cache line invalidation request is sent to the data cache.

It exists in Big-endian and Little-endian forms.

Component definition

Available in source:trunk/soclib/desc/soclib/mipsel.sd and source:trunk/soclib/desc/soclib/mipseb.sd

Usage

Mips has no parameters.

```
Uses( 'mipsel')
```

or

```
Uses( 'mipseb')
```

Mips Processor ISS Implementation

The implementation is in

- source:trunk/soclib/systemc/include/common/iss/mips.h
- source:trunk/soclib/systemc/src/common/iss/mips.cc
- source:trunk/soclib/systemc/src/common/iss/mips_instructions.cc

Template parameters

This component has no template parameters.

Constructor parameters

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

or

```
MipsEbIss( name, ident);
```

Visible registers

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

- r_pc : Program counter
- m_ins : Instruction register
- r_gpr[i] : General registers (0 < i < 32)
- r_hi & r_lo : Intermediate registers for multiply / divide instructions
- r_cp0[i] : Coprocessor 0 registers (0<=i<32). Implemented values:
    - 8: BAR : Bad address register
    - 12: SR : Status register
    - 13: CR : Cause register
    - 14: EPC : Exception PC register
    - 15: INFOS : CPU identification number on bits [9:0]

Interrupts

Mips defines 6 interrupts lines. Le lowest number has the hiest priority.

Ports

None, it is to the wrapper to provide them.