VciXcache Functional Description

This VCI initiator is a generic cache controller, fully compliant with the VCI advanced protocol. Thanks to a normalized interface (source:trunk/soclib/communication/xcache/caba/source/include/xcache_signals.h), this blocking cache controller can be used by several RISC processors (such as Mips R3000, Sparc V8, or OpenRisc 1000). It contains two separated instruction and data caches, sharing the same VCI interface. (source:trunk/soclib/communication/vci/caba/source/include/vci_signals.h)

- The VCI ADDRESS and DATA fields must have 32 bits, and the VCI ERROR field must have 1 bit.
- The number of lines must be a power of 2, and cannot be larger than 1024.
- The number of words must be a power of 2, and cannot be larger than 32.

In order to garanty the memory consistency, this component does NOT start a new VCI transaction until the previous transaction is completed. Therefore, it does not use the VCI PKTID and TRDID fields.

Instruction Cache

- The Instruction cache is direct mapping and read-only.
- It uses the Mapping Table to support uncached segments.
- In case of MISS, the processor is stalled until the missing cache line is available.
- The only VCI transaction generated by the Instruction cache is a read burst corresponding to a missing cache line.

Data Cache

- The Data cache is direct mapping, and the write policy is WRITE-THROUGH (the data is immediately written in memory, and the cache is updated only in case of HIT).
- The data cache contains a parametrably-sized write buffer. The cache controller builds a burst packet when there are successive writes in incrementing addresses.
- It uses the Mapping Table to support uncached segments.
- The Data cache accepts a line invalidate command.
- In case of MISS, the processor is stalled until the missing cache line is available.
- Three types of VCI transactions can be generated by the data cache:
    - read burst of fixed length, corresponding to a cached read MISS,
    - read one word, corresponding to an uncached read,
    - write burst of variable length,
- The processor is stalled in case of cached read MISS, in case of uncached read, or in case of write, if the write buffer is full.

Component definition

Available in source:trunk/soclib/module/internal_component/vci_xcache/caba/metadata/vci_xcache.sd

Usage

VciXcache has VCI ones and an optional write_buffer_depth defaulting to 8, it may be used like others, see SoclibCc/VciParameters, with an additional parameter prepended.

```
Uses( 'vci_xcache', **vci_parameters )
```

```
Uses( 'vci_xcache', write_buffer_depth = 16, **vci_parameters )
```

VciXcache CABA Implementation

The caba implementation is in

- source:trunk/soclib/module/internal_component/vci_xcache/caba/source/include/vci_xcache.h
- source:trunk/soclib/module/internal_component/vci_xcache/caba/source/src/vci_xcache.cpp

Template parameters

```
template<size_t write_buffer_depth, typename vci_param>
```

Standard size for write buffer is 8 words.

Constructor parameters

```
VciXCache(
    sc_module_name insname,
    const soclib::common::MappingTable &mt,
    const soclib::common::IntTab &index,
    size_t icache_lines,
    size_t icache_words,
    size_t dcache_lines,
    size_t dcache_words );
```

Ports

- sc_in<bool> **p_resetn** : Global system reset
- sc_in<bool> **p_clk** : Global system clock
- soclib::caba:ICacheCachePort **p_icache** : Icache interface
- soclib::caba:DCacheCachePort **p_dcache** : Dcache interface
- soclib::caba::VciInitiator<vci_param> **p_vci** : The VCI port