

VciDma

1) Functional Description

This VCI component is both a target and an initiator.

- Addressing as a target allows to configure it for a transfer.
- Initiator will do the transfer

There is only one DMA context handled at a time

An IRQ is optionally asserted when transfer is finished.

This hardware component checks for segmentation violation, and can be used as a default target.

This component has 4 memory-mapped registers :

- DMA_SRC

The physical address of the source buffer.

- DMA_DST

The physical address of the destination buffer.

- DMA_LEN

Length of transfer, in bytes. Writing to this register initiates the transfer, you should write to it after src and dst. This register gets back to 0 when transfer is finished.

- DMA_IRQ_ENABLED
 - ◆ Writing a non zero value enables the IRQ line (0 is disabling)
 - ◆ Reading a zero value indicates the completion of the transfer

For extensibility issues, you should access the DMA using globally-defined offsets.

You should include file `soclib/dma.h` from your software, it defines `DMA_SRC`, `DMA_DST`, `DMA_LEN`, `DMA_IRQ_ENABLED`.

Sample code:

```
#include "soclib/dma.h"

static const volatile void* dma = 0xc0000000;

void * memcpy(void *dst, const void *src, const size_t len)
{
    soclib_io_set( dma, DMA_DST, dst );
    soclib_io_set( dma, DMA_SRC, src );
    soclib_io_set( dma, DMA_LEN, len );
    while( soclib_io_get( dma, DMA_LEN ) )
        ;
    return dst;
}
```

```
}
```

(add -I/path/to/soclib/include to your compilation command-line)

2) Component definition & usage

[source:trunk/soclib/soclib/module/infrastructure_component/dma_infrastructure/vci_dma/caba/metadata/vci_dma.sd?](#)

See [SoclibCc/VciParameters](#)

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

3) CABA Implementation

CABA sources

- interface :
[source:trunk/soclib/soclib/module/infrastructure_component/dma_infrastructure/vci_dma/caba/source/include/vci_dma.h](#)
- implementation :
[source:trunk/soclib/soclib/module/infrastructure_component/dma_infrastructure/vci_dma/caba/source/src/vci_dma.cpp](#)

CABA Constructor parameters

```
VciDma(  
    sc_module_name name,    // Component Name  
    const soclib::common::IntTab & index,    // Target index  
    const soclib::common::MappingTable &mt,    // MappingTable  
    const size_t burst_size );    // Number of bytes transfered in a burst
```

CABA Ports

- sc_in<bool> **p_resetn** : Global system reset
- sc_in<bool> **p_clk** : Global system clock
- soclib::caba::VciTarget<vci_param> **p_vci_target** : The VCI target port
- soclib::caba::VciInitiator<vci_param> **p_vci_initiator** : The VCI initiator port
- sc_out<bool> **p_irq** : Interrupt port

4) TLM-T implementation

The TLM-T implementation is not available yet.