VciVgmn

1) Functional Description

This hardware component is a generic micro-network respecting the VCI advanced protocol. It supports several simultaneous transaction, and behaves as two fully independent packet switched network for VCI commands and VCI responses.

When several initiators try to reach the same target, the arbitration policy is round-robin. It gives the system designer a generic "communication black-box" with a parametrized number of VCI initiator ports (NB_INITIATOR), and a parameterized number of VCI target ports (NB_TARGET). It can be used to build a "flat" interconnect, where all VCI initiators and targets are identified by a single index:

- The VCI targets must be indexed from 0 to (NB_TARGET - 1).
- The VCI initiators must be indexed from 0 to (NB_INITIATOR - 1).

As any VCI advanced compliant interconnect, this component uses the MSB bits of the VCI ADDRESS field to route the command packets to the proper target, thanks to a routing table, implemented as a ROM. This routing table is build by the constructor from the informations stored in the mapping table. It uses the VCI RSRCID field to route the response packet to the initiator.

This component has two "structural" parameters, that can be used to fit the behaviour of a specific physical micro-network:

- The MIN_LATENCY parameter is a number of cycles that defines the latency of an empty network.
- The FIFO_DEPTH parameter can be increased to improve the saturation threshold.

2) Component definition and usage

source:trunk/soclib/soclib/module/network_component/vci_vgmn/caba/metadata/vci_vgmn.sd?

See SoclibCc/VciParameters

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

3) CABA Implementation

CABA sources

- interface : source:trunk/soclib/soclib/module/network_component/vci_vgmn/caba/source/include/vci_vgmn.h?
- implementation : source:trunk/soclib/soclib/module/network_component/vci_vgmn/caba/source/src/vci_vgmn.cpp?

CABA Constructor parameters

```
VciVgmn(
     sc_module_name name,   //  instance name
     const soclib::common::MappingTable &mt, // mapping table
     size_t nb_initiator,  // number of initiators
```

```
    size_t nb_target,  // number of targets
    size_t min_latency,  // minimal latency (one way)
    size_t fifo_depth );  //  internal FIFO depth
```

CABA Ports

- **p_resetn** : Global system reset
- **p_clk** : Global system clock
- **p_to_initiator[i]** : Ports to VCI initiators
- **p_to_target[i]** : Ports to VCI targets

CABA Implementation notes

Each micro network (the Command network and the Response network) is implemented within the same templated code. This code is instanciated twice, once for each network.

There is one InputRouter object per input port, it handles incoming packets and forwards them to the right OutputPortQueue.

There is one OutputPortQueue object per output port. Each output port j contains as many fifo[i,j] as input ports, and one single delay line. The delay line implements the MIN_LATENCY parameter, and fifo[i,j] correspond to the distributed fifos on the path between the input port i and the output port j. minimal latency parameter, and the handles putting packets on output port.

4) TLM-DT Implementation

TLM-DT sources

- interface : source:trunk/soclib/soclib/module/network_component/vci_vgmn/tlmdt/source/include/vci_vgmn.h?
- implementation : source:trunk/soclib/soclib/module/network_component/vci_vgmn/tlmdt/source/src/vci_vgmn.cpp?

TLM-DT Constructor parameters

```
VciVgmn(
    sc_module_name name,   //  instance name
    const soclib::common::MappingTable &mt, // mapping table
    size_t nb_initiator,  // number of initiators
    size_t nb_target,  // number of targets
    size_t min_latency,  // minimal latency (one way)
    size_t fifo_depth );  //  internal FIFO depth (unused in TLM-DT)
```

TLM-DT Ports

- **p_to_initiator[i]** : ports to VCI initiators (from [0] to [nb_initiator - 1])
- **p_to_target[i]** : ports to VCI targets (from [0] to [nb_target - 1])