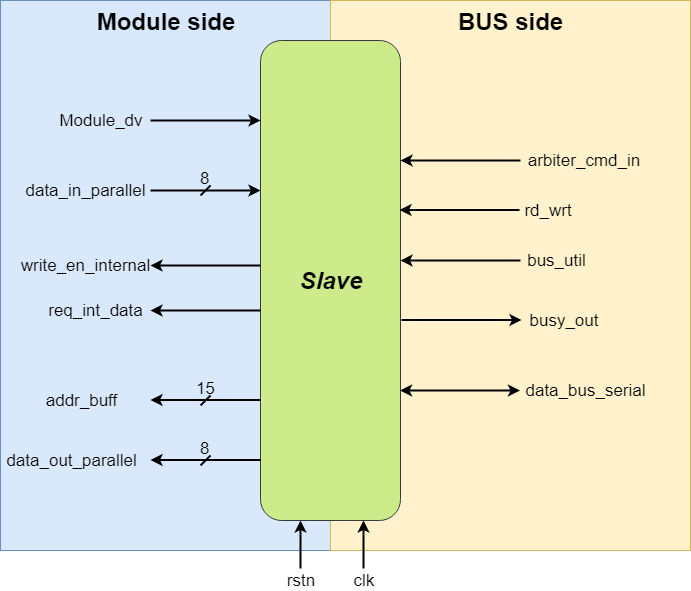
# Slave module

## Structure



## Operation

The slave waits in an idle state until a master pulls the normally high driven serial line down. Wen the serial is pulled down, the slave samples the first three bits of the address from the serial line and compares it to its own ID. If the ID is not matched, slave goes into a wait state, from which it will recover once the bus utilization ends. If the ID is matched, then the slave reads the rest of the address, sends the acknowledgement and then decides whether the transaction is read or write depending on the state of the rd\_wrt line.

If the transaction is a read, then the slave passes the received address to its internal memory, goes to a busy state and reads from it. Once the read is complete, busy signal is lowered and the slave waits until the arbiter tells it that the correct master is reconnected. Once the connection is established, slave sends the read data through the serial, waits for the acknowledgement from the master and when it is received, goes back to idle.

In case of a write transaction, the slave waits for an incoming data, goes to a busy state and writes the data to the memory. Once the write is done, the slave sends an acknowledgment to the master and goes to the idle state.

## Port description

* Module\_dv

This is the signal that the module gives to the slave interface to indicate that the module has successfully read the data available on the incoming data/address buffer, or it has placed the data on the out. Slave waits in a busy status until this signal is received, and once received, slave becomes ready for the next transaction.

* Data\_in\_parallel

This is an 8-bit wide line for the module to place whatever data that is to be sent to the master. Once the data is properly placed, module\_dv needs to be driven high.

* Write\_en\_internal

This is the signal given to the module by the interface to indicate that it has placed the data on the incoming buffer and instructs the module to read the data from the buffer.

* Req\_int\_data

This signal tells the module to place the data on the outgoing data buffer to be sent to the master.

* Addr\_buff

This 15-bit wide line outputs the address that was sent by the master to be read from.

* Data\_out\_parallel

This line is 8 bits wide and outputs the data byte received from the master.

* Arbiter\_cmd\_in

Signal from the arbiter indicating that the proper master has been connected again after an interrupted transaction.

* Rd\_wrt

Command from the master indicating whether the current transaction is a write or read.

* Bus\_util

A shared line indicating if the bus is currently in use.

* Busy\_out

Output to the arbiter indicating whether the slave is currently busy.

* Data\_bus\_serial

The bidirectional line used to communicate data and addresses.

* Rstn

Asynchronous active low reset signal. When this is pulled down, all the submodules inside the interface is reset at once.

* Clk

Clock signal input.