**GMII BFM** –Quick Reference

**BFM**

This is a stripped-down version of GMII with only data lines.

For general information see UVVM Essential Mechanisms located in uvvm\_vvc\_framework/doc.

|  |
| --- |
| init\_gmii\_to\_dut\_if (VOID) |
| Example: gmii\_to\_dut\_if <= init\_gmii\_to\_dut\_if(VOID); |

*gmii\_bfm\_pkg.vhd*

|  |
| --- |
| gmii\_write (data, msg, gmii\_to\_dut\_if, [scope, [msg\_id\_panel, [config]]]) |
| Example: gmii\_write(v\_write\_bytes, “Send 10 bytes of data”, gmii\_to\_dut\_if); |

|  |
| --- |
| gmii\_read (data, msg, gmii\_from\_dut\_if, [scope, [msg\_id\_panel, [config]]]) |
| Example: gmii\_read(v\_receive\_bytes, “Read 10 bytes of data”, gmii\_from\_dut\_if); |



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | BFM Configuration record ´**t\_gmii\_bfm\_config´** | | | | **Record element** | **Type** | **C\_GMII\_BFM\_CONFIG\_DEFAULT** | | clock\_period | time | 8 ns | | setup\_time | time | 2 ns | | hold\_time | time | 2 ns | | timeout | time | 1 us | | timeout\_severity | t\_alert\_level | TB\_ERROR | | id\_for\_bfm | t\_msg\_id | ID\_BFM | | id\_for\_bfm\_wait | t\_msg\_id | ID\_BFM\_WAIT | | |  |  | | --- | --- | | Signal record ´**t\_gmii\_to\_dut\_if´** | | | **Record element** | **Type** | | gtxclk | std\_logic | | txd | std\_logic\_vector(7 downto 0) | | txen | std\_logic |  |  |  | | --- | --- | | Signal record ´**t\_dut\_to\_gmii\_if´** | | | **Record element** | **Type** | | rxclk | std\_logic | | rxd | std\_logic\_vector(7 downto 0) | | rxdv | std\_logic | |

BFM non-signal parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Example(s)** | **Description** |
| data | t\_byte\_array | (x”1A”, x”1B”, x”2A”) | The data value written to or read from the DUT. |
| msg | string | “Write to Peripheral 1” | A custom message to be appended in the log/alert. |
| scope | string | "GMII BFM" | A string describing the scope from which the log/alert originates. In a simple single sequencer typically "GMII BFM". In a verification component typically "GMII\_VVC ". |
| msg\_id\_panel | t\_msg\_id\_panel | shared\_msg\_id\_panel | Optional msg\_id\_panel, controlling verbosity within a specified scope. Defaults to a common ID panel defined in the adaptations package. |
| config | t\_gmii\_bfm\_config | C\_GMII\_BFM\_CONFIG\_DEFAULT | Configuration of BFM behaviour and restrictions. See section 2 for details. |

BFM signal parameters

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| gtxclk | std\_logic | Clock signal for TX signals. |
| txd | std\_logic\_vector | Data from DUT. |
| txen | std\_logic | Transmit enable from DUT. |
| rxclk | std\_logic | Clock signal for RX signals. |
| rxd | std\_logic\_vector | Data to DUT. |
| rxdv | std\_logic | Data valid signal to DUT. |

BFM details

# BFM procedure details and examples

|  |  |
| --- | --- |
| **Procedure** | **Description** |
| **init\_gmii\_to\_dut\_if()** | **init\_gmii\_to\_dut\_if (VOID)**  This function initializes the gmii\_to\_dut\_if interface. All output-signals are set to '0', input signals are set to ‘Z’.  Example:  gmii\_to\_dut\_if <= init\_gmii\_to\_dut\_if(VOID); |
| **gmii\_write()** | **gmii\_write(data, msg, gmii\_to\_dut\_if, [scope, [msg\_id\_panel, [config]]])**  The gmii\_write() procedure writes the given data to the DUT, using the GMII protocol.  Example:  gmii\_write(v\_write\_data\_bytes, “Write data to DUT”, gmii\_to\_dut\_if); |
| **gmii\_read()** | **gmii\_read(data, msg, gmii\_from\_dut\_if, [scope, [msg\_id\_panel, [config]]])**  The gmii\_read() procedure reads data from the DUT, using the GMII protocol.  Example:  gmii\_write(v\_receive\_data\_bytes, “Read data from DUT”, gmii\_from\_dut\_if); |
|  |  |

# BFM Configuration record

Type name: t\_gmii\_bfm\_config

|  |  |  |  |
| --- | --- | --- | --- |
| **Record element** | **Type** | **C\_GMII\_BFM\_CONFIG\_DEFAULT** | **Description** |
| clock\_period | time | 8 ns | Period of the clock signal. |
| setup\_time | time | 2 ns | Generated signals setup time. Suggested value is clock\_period/4.  An alert is reported if setup\_time exceed clock\_period/2. |
| hold\_time | time | 2 ns | Generated signals hold time. Suggested value is clock\_period/4.  An alert is reported if hold\_time exceed clock\_period/2. |
| timeout | time | 1 us | The maximum time allowed to wait for DUT. |
| timeout\_severity | t\_alert\_level | TB\_ERROR | Severity of alert when timeout. |
| id\_for\_bfm | t\_msg\_id | ID\_BFM | The message ID used as a general message ID in the GMII BFM |
| id\_for\_bfm\_wait | t\_msg\_id | ID\_BFM\_WAIT | The message ID used for logging waits in the GMII BFM |

# Compilation

The GMII BFM may only be compiled with VHDL 2008. It is dependent on the UVVM Utility Library (UVVM-Util), which is only compatible with VHDL 2008.

See the separate UVVM-Util documentation for more info. After UVVM-Util has been compiled gmii\_bfm\_pkg.vhd can be compiled into any desired library.

See UVVM Essential Mechanisms located in uvvm\_vvc\_framework/doc for information about compile scripts.

## Simulator compatibility and setup

This BFM has been compiled and tested with Modelsim version 10.5b and Riviera-PRO version 2018.02.111.6909

For required simulator setup see UVVM-Util Quick reference.

# Local BFM overloads

A good approach for better readability and maintainability is to make simple, local overloads for the BFM procedures in the TB process.

This allows calling the BFM procedures with the key parameters only

e.g.

gmii\_write(v\_write\_data\_bytes, “Write data to DUT”);

rather than

gmii\_write(v\_write\_data\_bytes, “Write data to DUT”, gmii\_to\_dut\_if, C\_SCOPE,

shared\_msg\_id\_panel, C\_GMII\_BFM\_CONFIG\_DEFAULT);

By defining the local overload as e.g.:

procedure gmii\_write(

constant data : in t\_byte\_array;

constant msg : in string) is

begin

gmii\_write(data, -- keep as is

msg, -- keep as is

gmii\_to\_dut\_if, -- Signal must be visible in local process scope

C\_SCOPE, -- Just use the default

shared\_msg\_id\_panel, -- Use global, shared msg\_id\_panel

C\_GMII\_BFM\_CONFIG\_LOCAL); -- Use locally defined configuration or C\_GMII\_BFM\_CONFIG\_DEFAULT

end;

Using a local overload like this also allows the following – if wanted:

* Set up defaults for constants. May be different for two overloads of the same BFM
* Apply dedicated message ID panel to allow dedicated verbosity control

IMPORTANT   
This is a simplified Bus Functional Model (BFM) for GMII.  
The given BFM complies with the basic GMII protocol and thus allows a normal access towards a GMII interface. This BFM is not a GMII protocol checker.   
For a more advanced BFM please contact Bitvis AS at [support@bitvis.no](mailto:support@bitvis.no)

Disclaimer: This IP and any part thereof are provided "as is", without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose and noninfringement.  
In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of or in connection with this IP.

**INTELLECTUAL**

**PROPERTY**