GMII BFM – Quick Reference

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.

BFM

gmii_bfm_pkg.vhd

gmii_write (data_array, msg, gmii_tx_if, [scope, [msg_id_panel, [config]]])

Example: gmii_write(v_data_array(0 to v_numBytes-1), “Write v_numBytes bytes”, gmii_tx_if, C_SCOPE, shared_msg_id_panel, gmii_bfm_config);
Example: gmii_write((x”01”, x”02”, x”03”, x”04”), “Write 4 bytes”, gmii_tx_if);

gmii_read (data_array, data_len, msg, gmii_rx_if, [scope, [msg_id_panel, [config, [ext_proc_call]]]])

Example: gmii_read(v_data_array, v_numBytes, “Read v_numBytes bytes”, gmii_rx_if, C_SCOPE, shared_msg_id_panel, gmii_bfm_config, “gmii_expect()”);
Example: gmii_read(v_data_array, v_numBytes, “Read v_numBytes bytes”, gmii_rx_if);

gmii_expect (data_exp, msg, gmii_rx_if, alert_level, [scope, [msg_id_panel, [config]]])

Example: gmii_expect(v_data_array(0 to v_numBytes-1), “Expect v_numBytes bytes”, gmii_rx_if, ERROR, C_SCOPE, shared_msg_id_panel, gmii_bfm_config);
Example: gmii_expect((x”01”, x”02”, x”03”, x”04”), “Expect 4 bytes”, gmii_rx_if, ERROR);

init_gmii_if_signals (VOID)

Example: gmii_tx_if <= init_gmii_if_signals(VOID);

UVVM™

VHDL 2008 only

BFM Configuration record ´**t_gmii_bfm_config**´

Record element	Type	C_GMII_BFM_CONFIG_DEFAULT
max_wait_cycles	integer	10
max_wait_cycles_severity	t_alert_level	ERROR
clock_period	time	-1 ns
clock_period_margin	time	0 ns
clock_margin_severity	t_alert_level	TB_ERROR
setup_time	time	-1 ns
hold_time	time	-1 ns
bfm_sync	t_bfm_sync	SYNC_ON_CLOCK_ONLY
id_for_bfm	t_msg_id	ID_BFM

Signal record '**t_gmii_tx_if**'

Record element	Type
gtxclk	std_logic
txd	std_logic_vector(7 downto 0)
txen	std_logic

Signal record '**t_gmii_rx_if**'

Record element	Type
rxclk	std_logic
rxd	std_logic_vector(7 downto 0)
rxdv	std_logic

BFM signal parameters

Name	Type	Description
gtxclk	std_logic	TX reference clock
txd	std_logic_vector	TX data lines (to DUT)
txen	std_logic	TX enable
rxclk	std_logic	RX reference clock
rxd	std_logic_vector	RX data lines (from DUT)
rxdv	std_logic	RX data valid

BFM non-signal parameters

Name	Type	Example(s)	Description
data_array data_exp	t_byte_array	(x"D0", x"D1", x"D2", x"D3")	An array of bytes containing the data to be written/read. data_array(0) is written/read first, while data_array(data_array'high) is written/read last. For clarity, data_array is required to be ascending, for example defined by the test sequencer as follows: `variable v_data_array : t_byte_array(0 to C_MAX_BYTES-1);`
data_len	natural	v_data_len	The number of valid bytes in the data_array. Note that the data_array can be bigger and that is why the length is returned.
alert_level	t_alert_level	ERROR or TB_WARNING	Set the severity for the alert that may be asserted by the procedure.
msg	string	"Write bytes"	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 message ID panel defined in the UVVM-Util adaptations package.
config	t_gmii_bfm_config	C_GMII_BFM_CONFIG_DEFAULT	Configuration of BFM behaviour and restrictions. See section 2 for details.
ext_proc_call	string	"gmii_expect()"	External procedure call. Only use when called from another BFM procedure.

BFM details

1 BFM procedure details and examples

Procedure	Description
gmii_write()	**gmii_write (data_array, msg, gmii_tx_if, [scope, [msg_id_panel, [config]]])** The gmii_write() procedure writes data to the DUT. The length and data are defined by the "data_array" argument, which is a t_byte_array. data_array(0) is written first, while data_array(data_array'high) is written last.
gmii_read()	**gmii_read (data_array, data_len, msg, gmii_rx_if, [scope, [msg_id_panel, [config, ext_proc_call]]]])** The gmii_read() procedure reads data from the DUT. The received data is stored in the data_array output, which is a t_byte_array. The number of valid bytes in the data_array is stored in data_len. data_array(0) is read first, while data_array(data_array'high) is read last.
gmii_expect()	**gmii_expect (data_exp, msg, gmii_rx_if, alert_level, [scope, [msg_id_panel, [config]]])** Calls the gmii_read() procedure, then compares the received data with data_exp.
init_gmii_if_signals()	**init_gmii_if_signals (VOID)** This function initializes the GMII interface. All the BFM outputs are set to zeros ('0')

2 BFM Configuration record

Type name: t_gmii_bfm_config

Record element	Type	C_GMII_BFM_CONFIG_DEFAULT	Description
max_wait_cycles	integer	10	Used for setting the maximum cycles to wait before an alert is issued when waiting for signals from the DUT.
max_wait_cycles_severity	t_alert_level	ERROR	Severity if max_wait_cycles expires.
clock_period	time	-1 ns	Period of the clock signal.
clock_period_margin	time	0 ns	Input clock period margin to specified clock_period.
clock_margin_severity	t_alert_level	TB_ERROR	The above margin will have this severity.
setup_time	time	-1 ns	Setup time for generated signals. Suggested value is clock_period/4. An alert is reported if setup_time exceed clock_period/2.
hold_time	time	-1 ns	Hold time for generated signals. Suggested value is clock_period/4. An alert is reported if hold_time exceed clock_period/2.
bfm_sync	t_bfm_sync	SYNC_ON_CLOCK_ONLY	Sets the start and exit synchronisation of the BFM.
id_for_bfm	t_msg_id	ID_BFM	The message ID used as a general message ID in the BFM.

3 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.

3.1 Simulator compatibility and setup

See README.md for a list of supported simulators. For required simulator setup see UVVM-Util Quick reference.

4 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_data_array(0 to 1), "msg");
```

rather than

```
gmii_write(v_data_array(0 to 1), "msg", gmii_tx_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_array : in t_byte_array;
  constant msg        : in string) is
begin
  gmii_write(data_array,                    -- keep as is
             msg,                           -- keep as is
             gmii_tx_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