**Ethernet BFM** –Quick Reference

**BFM**

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

|  |
| --- |
| encode\_ethernet\_frame (mac\_dest, mac\_src, payload, [scope, [msg\_id\_panel]]) |
| Example: v\_encoded\_ethernet\_frame := encode\_ethernet\_frame(v\_mac\_dest, v\_mac\_src, v\_payload); |

*ethernet\_bfm\_pkg.vhd*

|  |
| --- |
| decode\_ethernet\_frame (data, [scope, [msg\_id\_panel]]) |
| Example: v\_decoded\_ethernet\_frame := decode\_ethernet\_frame(v\_data); |



|  |
| --- |
| encode\_ethernet\_packet (data, [scope, [msg\_id\_panel]]) |
| Example: v\_encoded\_ethernet\_packet := encode\_ethernet\_packet(v\_encoded\_ethernet\_frame); |

|  |
| --- |
| decode\_ethernet\_packet (data, [scope, [msg\_id\_panel]]) |
| Example: v\_decoded\_ethernet\_packet := decode\_ethernet\_packet(v\_data); |

|  |
| --- |
| send\_to\_sub\_vvc (interface, data, [scope, [msg\_id\_panel]]) |
| Example: send\_to\_sub\_vvc(GMII, v\_encoded\_ethernet\_packet); |

|  |
| --- |
| receive\_from\_sub\_vvc (interface, data\_received, [scope, [msg\_id\_panel]]) |
| Example: receive\_from\_sub\_vvc(GMII, v\_data); |

|  |
| --- |
| expect\_from\_sub\_vvc (interface, data\_exp, [scope, [msg\_id\_panel]]) |
| Example: receive\_from\_sub\_vvc(GMII, v\_data); |

|  |  |  |
| --- | --- | --- |
| BFM Configuration record ´**t\_ethernet\_bfm\_config´** | | |
| **Record element** | **Type** | **C\_ETHERNET\_BFM\_CONFIG\_DEFAULT** |
| mac\_destination | t\_byte\_array(0 to 5) | x”000000000000” |
| mac\_source | t\_byte\_array(0 to 5) | x”000000000000” |
| fcs\_error\_severity | t\_alert\_level | TB\_ERROR |
| interpacket\_gap\_time | time | 96 ns |

|  |  |
| --- | --- |
| Record ´**t\_ethernet\_frame´** | |
| **Record element** | **Type** |
| mac\_destination | t\_byte\_array(0 to 5) |
| mac\_source | t\_byte\_array(0 to 5) |
| length | t\_byte\_array(0 to 1) |
| payload | t\_byte\_array |
| fcs | t\_byte\_array(0 to 3) |

|  |  |
| --- | --- |
| Record ´**t\_ethernet\_packet´** | |
| **Record element** | **Type** |
| preamble | t\_byte\_array(0 to 6) |
| sfd | t\_byte? |
| ethernet\_frame | t\_ethernet\_frame |
|  |  |

BFM non-signal parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Example(s)** | **Description** |
| mac\_dest | t\_byte\_array |  | The MAC address of destination. |
| mac\_src | t\_byte\_array |  | The MAC address of source. |
| payload | t\_byte\_array |  | The payload containing data. |
| data | t\_byte\_array |  | The raw data of the whole ethernet frame or ethernet packet. |
| data\_received | t\_byte\_array |  | The received data. |
| data\_exp | t\_byte\_array |  | The expected data. |
| interface | t\_interface | GMII, UART | The interface of the sub-VVC used. |
| scope | string | "ETHERNET BFM” | A string describing the scope from which the log/alert originates. In a simple single sequencer typically "ETHERNET BFM". In a verification component typically " ETHERNET 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. |

BFM details

# BFM procedure details and examples

|  |  |
| --- | --- |
| **Procedure** | **Description** |
| **encode\_ethernet\_frame()** | **encode\_ethernet\_frame(mac\_dest, mac\_src, payload, [scope, [msg\_id\_panel]])**  The encode\_ethernet\_frame() function returns a an encoded ethernet frame in t\_byte\_array format.  Example:  v\_encoded\_ethernet\_frame := encode\_ethernet\_frame(v\_mac\_dest, v\_mac\_src, v\_payload); |
| **decode\_ethernet\_frame()** | **decode\_ethernet\_frame(data, [scope, [msg\_id\_panel]])**  The decode\_ethernet\_frame() functions returns a t\_ethernet\_frame record with all ethernet frame fields decoded.  Example:  v\_decoded\_ethernet\_frame := decode\_ethernet\_frame(v\_received\_ethernet\_frame); |
| **encode\_ethernet\_packet()** | **encode\_ethernet\_packet(data, [scope, [msg\_id\_panel]])**  The encode\_ethernet\_packet() function returns an encoded ethernet packet in t\_byte\_array format.  Example:  v\_encoded\_ethernet\_packet := encode\_ethernet\_packet(v\_encoded\_ethernet\_frame); |
| **decode\_ethernet\_packet()** | **decode\_ethernet\_packet(data, [scope, [msg\_id\_panel]])**  The decode\_ethernet\_packet() function returns a ethernet frame in t\_byte\_array format.  Example:  v\_decoded\_ethernet\_packet := decode\_ethernet\_packet(v\_received\_ethernet\_packet); |
| **send\_to\_sub\_vvc()** | **send\_to\_sub\_vvc(interface, data, msg, [scope, [msg\_id\_panel]])**  The send\_to\_sub\_vvc() procedure sends data to the sub-VVC specified in the interface parameter.  Example:  send\_to\_sub\_vvc(GMII, v\_encoded\_ethernet\_packet, “Sending ethernet packet over GMII”); |
| **receive\_from\_sub\_vvc()** | **receive\_from\_sub\_vvc(interface, data\_received, msg, [scope, [msg\_id\_panel]])**  The receive\_from\_sub\_vvc() procedure receive data from the sub-VVC specified in the interface parameter.  Example:  receive\_from\_sub\_vvc(GMII, v\_received\_ethernet\_packet, “Receiving ethernet packet over GMII”); |
| **expect\_from\_sub\_vvc()** | **expect\_from\_sub\_vvc(interface, data\_exp, msg, [scope, [msg\_id\_panel]])**  The expect\_from\_sub\_vvc() procedure receive data from the sub-VVC specified in the interface parameter and check against expected data.  Example:  expect\_from\_sub\_vvc(GMII, v\_expected\_ethernet\_packet, “Expecting ethernet packet over GMII”); |

# Compilation

The Ethernet 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, the ethernet\_bfm\_pkg.vhd BFM 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 10.3d and Riviera-PRO version 2015.10.85.

For required simulator setup see UVVM-Util Quick reference.

\*1 <https://www.aldec.com/en/support/resources/multimedia/webinars/1673>

# 

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**