**UART MONITOR** –Quick Reference

**MONITOR**

**NOTE: As of UVVM v3.x, all shared variables have been made protected. This means that any access to shared variables must be done**

**using get- and set-methods. This documentation has not yet been updated with the methods for accessing these variables, but will be very soon.**

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

UART Monitor Configuration record **´t\_uart\_monitor\_config´ **is accessible via shared variable array **shared\_uart\_monitor\_config(channel, instance)**.

*uart\_monitor.vhd*

The UART transaction information is located in **‘t\_transaction\_group’** accessible via shared variable array **shared\_uart\_monitor\_transaction\_info(channel, instance)**.

**t\_uart\_monitor\_config**

Record hierarchy of **‘t\_transaction\_group’ –** accessible via shared\_uart\_monitor\_transaction\_info

|  |  |
| --- | --- |
| **Record element** | **Type** |
| 🡪 bt | t\_transaction |
| 🡪 operation | t\_operation |
| 🡪 data | std\_logic\_vector |
| 🡪 vvc\_meta1 | t\_vvc\_meta |
| 🡪 transaction\_status | t\_transaction\_status |
| 🡪 error\_info | t\_error\_info |
| 🡪 parity\_bit\_error | boolean |
| 🡪 stop\_bit\_error | boolean |
| 🡪 ct | t\_transaction |
|  |  |

Message IDs for UART Monitor

|  |  |
| --- | --- |
| **Message ID** | **Description** |
| ID\_FRAME\_INITIATE | Logs start of UART frame. |
| ID\_MONITOR | Logs information about monitored transaction. |

|  |  |  |
| --- | --- | --- |
| **Record element** | **Type** | **C\_UART\_MONITOR\_CONFIG\_DEFAULT** |
| scope\_name | string |  |
| msg\_id\_panel | t\_msg\_id\_panel | C\_UART\_MONITOR\_MSG\_ID\_PANEL\_DEFAULT |
| interface\_config | t\_uart\_interface\_config | C\_UART\_MONITOR\_INTERFACE\_CONFIG\_DEFAULT |
| transaction\_display\_time | time | 0 ns |

**t\_uart\_interface\_config**

|  |  |  |
| --- | --- | --- |
| **Record element** | **Type** | **Description** |
| bit\_time | time | The time used to transfer one bit. | |
| num\_data\_bits | positive range 7 to 8 | Number of data bits. |
| parity | t\_parity | The parity used, PARITY\_ODD or PARITY\_EVEN. |
| num\_stop\_bits | t\_stop\_bits | Number of stop bits, STOP\_BITS\_ONE, STOP\_BIT\_ONE\_AND\_HALF or STOP\_BITS\_TWO. |

**t\_base\_transaction**

|  |  |  |
| --- | --- | --- |
| **Record element** | **Type** | **Description** |
| operation | t\_operation | Operation on UART line, TRANSMIT, RECEIVE or NO\_OPERATION. | |
| data | std\_logic\_vector | UART data. | |
| vvc\_meta1 | t\_vvc\_meta | Only used by VVC. |
| transaction\_status | t\_transaction\_status | Status of transaction, SUCCEEDED, FAILED, INACTIVE or IN\_PROGRESS. |
| error\_info | t\_error\_info | Error information when failed transaction. |

**t\_error\_info**

|  |  |  |
| --- | --- | --- |
| **Record element** | **Type** | **Description** |
| parity\_bit\_error | boolean | True if parity error detected. |
| stop\_bit\_error | boolean | True if stop bit error detected. |



1vvc\_meta only applies for the VVC

Monitor entity signals

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Direction** | **Description** |
| uart\_dut\_rx | std\_logic | Input | Input of DUTs UART RX signal. |
| uart\_dut\_tx | std\_logic | Input | Input of DUTs UART TX signal. |

Monitor entity generic constants

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Default** | **Description** |
| GC\_INSTANCE\_IDX | natural | 1 | Instance number to assign the monitor. |
| GC\_MONITOR\_CONFIG | t\_uart\_monitor\_config | C\_UART\_MONITOR\_CONFIG\_DEFAULT | Configuration of the UART monitor, both channels get initiated with this configuration. |

# Use of monitor transaction info

All transaction information from the UART Monitor is located in the shared variable **shared\_uart\_monitor\_transaction\_info(channel, instance).**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Example(s)** | **Description** |
| global\_uart\_monitor\_transaction\_trigger(channel, instance\_idx) | std\_logic | ‘1’ | Global trigger pulsed when UART Monitor transaction info is available. |
| shared\_uart\_monitor\_transaction\_info(channel, instance\_idx) | t\_uart\_transaction\_array | global\_uart\_monitor\_transaction(RX, 1) | Shared variable containing all UART Monitor transaction information. |
| channel | t\_channel | TX, RX | The interface channel of the monitor instance |
| instance\_idx | natural | 1,2, etc. | Instance number of the monitor |

An example of use of the global\_uart\_monitor\_transaction and shared\_uart\_monitor\_transaction\_info is seen below. A process extracts the transaction info from the shared variable when the global signal is triggered.

p\_monitor\_tx : process

variable v\_transaction : t\_uart\_transaction;

begin

wait until global\_uart\_monitor\_transaction\_trigger(TX, 1) = ‘1’;

if (shared\_uart\_monitor\_transaction\_info(TX, 1).bt.transaction\_status = SUCCEEDED or

shared\_uart\_monitor\_transaction\_info(TX, 1).bt.transaction\_status = FAILED) then

v\_transaction := shared\_uart\_monitor\_transaction\_info(TX, 1).bt;

end if;

-- Processing received transaction

...

end process p\_monitor\_tx;

Monitor details

# Monitor Configuration

|  |  |  |  |
| --- | --- | --- | --- |
| **Record element** | **Type** | **C\_UART\_MONITOR\_CONFIG\_DEFAULT** | **Description** |
| scope\_name | string |  | A string describing the scope from which the log/alert originates. |
| msg\_id\_panel | t\_msg\_id\_panel | C\_UART\_MONITOR\_MSG\_ID\_PANEL\_DEFAULT | The message id panel used by the monitor instance. |
| interface\_config | t\_uart\_interface\_config | C\_UART\_MONITOR\_INTERFACE\_CONFIG\_DEFAULT | The configuration for the interface. |
| transaction\_display\_time | time | 0 ns | After this amount of time operation is set to NO\_OPERATION and transaction\_status is set to INACTIVE if not a new transaction is received. If set to 0 ns operation and transaction\_status will be unchanged until the next transfer is started. |

The configuration record can be accessed from the Central Testbench Sequencer through the shared variable array, e.g.:

shared\_uart\_monitor\_config(TX, 1).msg\_id\_panel := new\_msg\_id\_panel;

shared\_uart\_monitor\_config(TX, 1).interface\_config.num\_data\_bits := 8;

# Additional Documentation

Additional documentation about UVVM and its features can be found under “/uvvm\_vvc\_framework/doc/”.

For additional documentation on the UART protocol, please see the UART specification.

# Compilation

The UART Monitor must be compiled with VHDL 2008.   
It is dependent on the following libraries

* ***UVVM Utility Library (UVVM-Util), version 2.13.0 and up***
* ***UVVM VVC Framework, version 2.8.0 and up***
* ***UART BFM***

Before compiling the UART Monitor, make sure that uvvm\_vvc\_framework and uvvm\_util have been compiled.

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

**Compile order for the UART Monitor:**

|  |  |  |
| --- | --- | --- |
| **Compile to library** | **File** | **Comment** |
| bitvis\_vip\_uart | transaction\_pkg.vhd | UART transaction types |
| bitvis\_vip\_uart | uart\_bfm\_pkg.vhd | UART BFM |
| bitvis\_vip\_uart | vvc\_cmd\_pkg.vhd | UART VVC command types and operations |
| bitvis\_vip\_uart | monitor\_cmd\_pkg.vhd | UART Monitor command types and operations |
| bitvis\_vip\_uart | ../uvvm\_vvc\_framework/src\_target\_dependent/td\_target\_support\_pkg.vhd | UVVM VVC target support package, compiled into the UART VIP library. |
| bitvis\_vip\_uart | vvc\_methods\_pkg.vhd | UART VVC methods |
| bitvis\_vip\_uart | uart\_monitor.vhd | UART Monitor |

# Simulator compatibility and setup

This Monitor has been compiled and tested with Modelsim version 10.5b and Riviera-PRO version 2015.10.85.

For required simulator setup see ***UVVM-Util*** Quick reference.

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**