bitvis

AXI4-Stream VVC – Quick Reference

VVC

1 Methods

The following methods are available (in addition to UVVM common methods defined in VVC_Framework_common_methods_QuickRef)

1.1 When GC_VVC_IS_MASTER = true:

When GC_VVC_IS_MASTER is true, the VVC transmits data, and the following methods are available:

axistream_transmit (VVCT, vvc_instance_idx, data_array, [user_array], msg)

Example (tdata'length = 16) : axistream_transmit (AXISTREAM_VVCT, 0, (x"D0", x"D1", x"D2", x"D3"), (x"00", x"0A"), "Send a 4 byte packet with tuser=A at the 2nd (last) word", clk, axistream_if);
Example (tdata'length = 8) : axistream_transmit (AXISTREAM_VVCT, 0, (x"D0", x"D1", x"D2", x"D3"), (x"00", x"00", x"00", x"0A"), "Send a 4 byte packet with tuser=A at the 4th (last) word", clk, axistream_if);
Example: axistream_transmit(AXISTREAM_VVCT, 0, v_data_array(0 to 1), "Send a 2 byte packet to DUT, tuser=0 each word / clock cycle");
Example: axistream_transmit(AXISTREAM_VVCT, 0, v_data_array(0 to v_numBytes-1), v_user_array(0 to v_numWords-1), " Send a 'v_numBytes' byte packet to DUT");

1.2 When GC_VVC_IS_MASTER = false

When GC_VVC_IS_MASTER is false, the VVC receives data, and the following methods are available:

axistream_expect (VVCT, vvc_instance_idx, exp_data_array, [exp_user_array], msg, [alert_level])

Example (tdata'length = 16) : axistream_expect(AXISTREAM_VVCT, 0, (x"D0", x"D1", x"D2", x"D3"), (x"00", x"0A"), "Expect a 4 byte packet with tuser=A at the 2nd (last) word", clk, axistream_if);
Example (tdata'length = 8) : axistream_expect(AXISTREAM_VVCT, 0, (x"D0", x"D1", x"D2", x"D3"), (x"00", x"00", x"00", x"0A"), "Expect a 4 byte packet with tuser=A at the 4th (last) word", clk, axistream_if);
Example: axistream_expect(AXISTREAM_VVCT, 0, v_data_array(0 to 1), "Expect a 2 byte packet, ignoring the tuser bits");
Example: axistream_expect(AXISTREAM_VVCT, 0, v_data_array(0 to v_numBytes-1), v_user_array(0 to v_numWords-1), "Expect a packet, checking the tuser bits");

axistream_receive (VVCT, vvc_instance_idx, msg)

Example: axistream_receive (AXISTREAM_VVCT, 1, "Receive packet, which is stored in VVC and will be fetched later using fetch_result() ");

AXI4-Stream VVC Configuration record ´vvc_config´ -- accessible via shared_axistream_vvc_config

Parameter name	Type	C_AXISTREAM_VVC_CONFIG_DEFAULT
inter_bfm_delay	t_inter_bfm_delay	C_AXISTREAM_INTER_BFM_DELAY_DEFAULT
cmd_queue_count_max	natural	C_CMD_QUEUE_COUNT_MAX
cmd_queue_count_threshold	natural	C_CMD_QUEUE_COUNT_THRESHOLD
cmd_queue_count_threshold_severity	t_alert_level	C_CMD_QUEUE_COUNT_THRESHOLD_SEVERITY
bfm_config	t_axistream_bfm_config	C_AXISTREAM_BFM_CONFIG_DEFAULT
msg_id_panel	t_msg_id_panel	C_VVC_MSG_ID_PANEL_DEFAULT

AXI4-Stream VVC Status record signal ´vvc_status´ -- accessible via shared_axistream_vvc_status

Common VVC procedures applicable for this VVC

- See UVVM Methods QuickRef for details.

await_completion()
enable_log_msg()
disable_log_msg()
fetch_result()
flush_command_queue()
terminate_current_command()
terminate_all_commands()
insert_delay()
get_last_received_cmd_idx()

VHDL 2008 only

UVVM™

Parameter name	Type
current_cmd_idx	natural
previous_cmd_idx	natural
pending_cmd_cnt	natural

VVC target parameters

Name	Type	Example(s)	Description
VVCT	t_vvc_target_record	AXISTREAM_VVCT	VVC target type compiled into each VVC in order to differentiate between VVCs.
vvc_instance_idx	integer	1	Instance number of the VVC

VVC functional parameters

Name	Type	Example(s)	Description
data_array	t_slv8_array	x"D0" & x"D1"	A byte array containing the packet data to be sent or the data received. t_slv8_array is defined in axistream_bfm_pkg. Refer to the AXI4-Stream BFM documentation
user_array	t_user_array	x"1" & x"2"	Sideband data to send or has been received via the tuser signal. t_user_array is defined in axistream_bfm_pkg. Refer to the AXI4-Stream BFM documentation
msg	string	"Send data"	A custom message to be appended in the log/alert
alert-level	t_alert_level	ERROR or TB_WARNING	Set the severity for the alert that may be asserted by the method.

VVC entity signals

Name	Type	Description
clk	std_logic	VVC Clock signal
axistream_vvc_master_if	t_axistream_if	See AXI4-Stream BFM documentation

VVC entity generic constants

Name	Type	Default	Description
GC_VVC_IS_MASTER	boolean	-	Set to true when this VVC instance is an AXI4 Stream master (data is output from BFM). Set to false when this VVC is an AXI4 Stream slave (data is input to BFM.)
GC_DATA_WIDTH	integer	-	Width of the AXI4-Stream data bus
GC_USER_WIDTH	integer	-	Width of the AXI4-Stream tuser bus. Note1: if GC_USER_WIDTH wider than 8 is required, increase the value of the constant c_maxTUserBits in axistream_bfm_pkg. Note2: If the tuser signal doesn't exist in DUT's interface, refer to description in Section 5
GC_INSTANCE_IDX	natural	-	Instance number to assign the VVC
GC_AXISTREAM_CONFIG	t_axistream_bfm_config	C_AXISTREAM_BFM_CONFIG_DEFAULT	Configuration for the AXI4-Stream BFM, see AXI4-Stream BFM documentation.
GC_CMD_QUEUE_COUNT_MAX	natural	1000	Absolute maximum number of commands in the VVC command queue
GC_CMD_QUEUE_COUNT_THRESHOLD	natural	950	An alert will be generated when reaching this threshold to indicate

			that the command queue is almost full. The queue will still accept new commands until it reaches C_CMD_QUEUE_COUNT_MAX.
GC_CMD_QUEUE_COUNT_THRESHOLD_SEVERITY	t_alert_level	WARNING	Alert severity which will be used when command queue reaches GC_CMD_QUEUE_COUNT_THRESHOLD.
GC_RESULT_QUEUE_COUNT_MAX	natural	1000	Maximum number of unfetched results before result_queue is full.
GC_RESULT_QUEUE_COUNT_THRESHOLD	natural	950	An alert with severity 'result_queue_count_threshold_severity' will be issued if command queue exceeds this count. Used for early warning if result queue is almost full. Will be ignored if set to 0.
GC_RESULT_QUEUE_COUNT_THRESHOLD_SEVERITY	t_alert_level	WARNING	Severity of alert to be initiated if exceeding result_queue_count_threshold

VVC details

All VVC procedures are defined in vvc_methods_pkg (dedicated this VVC), and uvvm_vvc_framework.uvvm_methods_pkg and uvvm_vvc_framework.uvvm_support_pkg (common VVC procedures)

2 VVC procedure details

Procedure	Applicable when VVC is…	Description
axistream_transmit()	Master	The axistream_transmit() VVC procedure adds a transmit command to the AXI4-Stream VVC executor queue, which will run as soon as all preceding commands have completed. When the command is scheduled to run, the executor calls the AXI4-Stream BFM axistream_transmit() procedure, described in the AXI4-Stream BFM QuickRef.
axistream_expect()	Slave	The axistream_expect() VVC procedure adds an expect command to the AXI4-Stream VVC executor queue, which will run as soon as all preceding commands have completed. When the command is scheduled to run, the executor calls the AXI4-Stream BFM axistream_expect() procedure, described in the AXI4-Stream BFM QuickRef.

axistream_receive()	Slave	The axistream_receive() VVC procedure adds a receive command to the AXISTREAM VVC executor queue, which will run as soon as all preceding commands have completed. When the receive command is scheduled to run, the executor calls the AXISTREAM BFM axistream_receive() procedure, described in the AXISTREAM BFM QuickRef.
		The value receive from DUT will not be returned in this procedure call since it is non-blocking for the sequencer/caller, but the received data and metadata will be stored in the VVC for a potential future fetch (see example with **fetch_result** below).
		
		axistream_receive (VVCT, vvc_instance_idx, addr, msg)
		e.g.
		- axistream_receive(AXISTREAM_VVCT, 1, “Receive data to VVC”);
		
		Example with fetch_result() call: Result is placed in **v_result**
		`    variable v_cmd_idx          : natural;                    -- Command index for the last receive`
		`    variable v_result           : work.vvc_cmd_pkg.t_vvc_result; -- Result from receive (data and metadata)`
		(…)
		`    axistream_receive(AXISTREAM_VVCT, 1, “Receive data to VVC”);`
		`    v_cmd_idx := shared_cmd_idx;`
		`    await_completion(AXISTREAM_VVCT,1, 1 ms, "Wait for receive to finish");`
		`    fetch_result(AXISTREAM_VVCT,1, v_cmd_idx, v_result, "Fetching result from receive operation");`

3 VVC Configuration

Name	Type	C_AXISTREAM_BFM_CONFIG_DEFAULT	Description
inter_bfm_delay	t_inter_bfm_delay	C_AXISTREAM_INTER_BFM_DELAY_DEFAULT	Minimum delay between BFM accesses from the VVC. If parameter delay_type is set to NO_DELAY, BFM accesses will be back to back, i.e. no delay.
cmd_queue_count_max	natural	C_CMD_QUEUE_COUNT_MAX	Maximum pending number in command queue before queue is full. Adding additional commands will result in an ERROR.
cmd_queue_count_threshold	natural	C_CMD_QUEUE_COUNT_THRESHOLD	An alert with severity “cmd_queue_count_threshold_severity” will be issued if command queue exceeds this count. Used for early warning if command queue is almost full. Will be ignored if set to 0.
cmd_queue_count_threshold_severity	t_alert_level	C_CMD_QUEUE_COUNT_THRESHOLD_SEVERITY	Severity of alert to be initiated if exceeding cmd_queue_count_threshold
result_queue_count_max	natural	C_RESULT_QUEUE_COUNT_MAX	Maximum number of unfetched results before result_queue is full.
result _queue_count_threshold	natural	C_RESULT_QUEUE_COUNT_THRESHOLD	An alert with severity 'result_queue_count_threshold_severity' will be issued if command queue exceeds this count. Used for early warning if result queue is almost full. Will be ignored if set to 0.
result _queue_count_threshold_severity	t_alert_level	C_ RESULT_QUEUE_COUNT_THRESHOLD_SEVERITY	Severity of alert to be initiated if exceeding result_queue_count_threshold
bfm_config	t_axistream_bfm_config	C_AXISTREAM_BFM_CONFIG_DEFAULT	Configuration for AXI4-Stream BFM. See quick reference for AXI4-Stream BFM
msg_id_panel	t_msg_id_panel	C_VVC_MSG_ID_PANEL_DEFAULT	VVC dedicated message ID panel

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

```
  shared_axistream_vvc_config(1).inter_bfm_delay.delay_in_time := 50 ns;
  shared_axistream_vvc_config(1).bfm_config.clock_period       := 10 ns;
```

4 VVC Status

The current status of the VVC can be retrieved during simulation. This is achieved by reading from the shared variable shared_axistream_vvc_status record from the test sequencer. The record contents can be seen below:

Name	Type	Description
current_cmd_idx	natural	Command index currently running
previous_cmd_idx	natural	Previous command index to run
pending_cmd_cnt	natural	Pending number of commands in the command queue

5 VVC Interface

In this VVC, the interface has been encapsulated in a signal record of type **t_axistream_if** in order to improve readability of the code. Since the AXI4-Stream interface busses can be of arbitrary size, the interface std_logic_vectors have been left unconstrained. These unconstrained SLVs needs to be constrained when the interface signals are instantiated. For this interface, the could look like:

```
    signal axistream_if : t_axistream_if(tdata(C_DATA_WIDTH -1 downto 0),
                                         tkeep((C_DATA_WIDTH/8)-1 downto 0),
                                         tuser(C_USER_WIDTH -1 downto 0)
                                         );
```

Note that the `tuser` element must be present even when it is not used (connected to DUT). In this case, it is recommended to set `C_USER_WIDTH = 1`.
If the record signal connects to a Slave BFM, where the tuser element is an input, assign a dummy value: `axistream_if.tuser <= (others => '0');`
Regardless, the `tuser` check will be skipped when the test sequencer calls `axistream_expect()` without providing the `user_array` argument.

6 Additional Documentation

Additional documentation about UVVM and its features can be found under “/uvvm_vvc_framework/doc/”.
For additional documentation on the AXI4-Stream standard, refer to “AMBA 4 AXI4-Stream Protocol Specification (ARM IHI 0051)”, available from ARM.

7 Compilation

AXI4-Stream VVC must be compiled with VHDL 2008.
It is dependent on the following libraries

- UVVM Utility Library (UVVM-Util), version 1.0.0 and up
- UVVM VVC Framework, version 1.0.0 and up
- AXI4-Stream BFM

Before compiling the AXI4-Stream VVC, assure that uvvm_vvc_framework and uvvm_util have been compiled.

Compile order for the AXI4-Stream VVC:

Compile to library	File	Comment
bitvis_vip_axistream	axistream_bfm_pkg.vhd	AXI4-Stream BFM
bitvis_vip_axistream	vvc_cmd_pkg.vhd	AXI4-Stream VVC command types and operations
bitvis_vip_axistream	../uvvm_vvc_framework/src_target_dependent/td_target_support_pkg.vhd	UVVM VVC target support package, compiled into the AXI4-Stream VVC library.
bitvis_vip_axistream	../uvvm_vvc_framework/src_target_dependent/td_vvc_framework_common_methods_pkg.vhd	UVVM framework common methods compiled into the AXI4-Stream VVC library
bitvis_vip_axistream	vvc_methods_pkg.vhd	AXI4-Stream VVC methods
bitvis_vip_axistream	../uvvm_vvc_framework/src_target_dependent/td_queue_pkg.vhd	UVVM queue package for the VVC
bitvis_vip_axistream	../uvvm_vvc_framework/src_target_dependent/td_vvc_entity_support_pkg.vhd	UVVM VVC entity support compiled into the AXI4-Stream VVC library
bitvis_vip_axistream	axistream_vvc.vhd	AXI4-Stream VVC

8 Simulator compatibility and setup

This VVC has been compiled and tested with Modelsim version 10.5b.
For required simulator setup see UVVM-Util Quick reference.

IMPORTANT
This is a simplified Verification IP (VIP) for AXI4-Stream. The given VIP complies with the basic AXI4-Stream protocol and thus allows a normal access towards an AXI4-Stream interface.
This VIP is not AXI4-Stream protocol checker. For a more advanced VIP please contact Bitvis AS at support@bitvis.no