UVVM Common Methods – Quick Reference

await_completion (vvc_target, vvc_instance_idx, [vvc_channel,] [wanted_idx,] [timeout, [msg]])

Example: await_completion(SBI_VVCT, 1, 100 ns, “Waiting for all SBI commands to complete”);

enable_log_msg (vvc_target, vvc_instance_idx, [vvc_channel,] msg_id, [msg])

Example: enable_log_msg(UART_VVCT, 1, RX, ID_BFM);

disable_log_msg (vvc_target, vvc_instance_idx, [vvc_channel,] msg_id, [msg])

Example: disable_log_msg(SBI_VVCT, 1, ID_BFM);

fetch_result (vvc_target, vvc_instance_idx, [vvc_channel,] wanted_idx, result, [fetch_is_accepted, [value_is_new,]] [msg, [alert_level]])

Example: fetch_result(SBI_VVCT, 1, v_idx, v_result, v_fetch_is_accepted, v_value_is_new);

flush_command_queue (vvc_target, vvc_instance_idx, [vvc_channel,] [msg])

Example: flush_command_queue(AXILITE_VVCT, 1);

terminate_current_command (vvc_target, vvc_instance_idx, [vvc_channel, [msg]])

Example: terminate_current_command(SBI_VVCT, 1);

terminate_all_commands (vvc_target, vvc_instance_idx, [vvc_channel, [msg]])

Example: terminate_all_commands(UART_VVCT, 1, RX);

insert_delay (vvc_target, vvc_instance_idx, delay, [vvc_channel,] [msg])

Example: insert_delay(SBI_VVCT, 1,100 ns);
Example: insert_delay(SBI_VVCT, 1,10); -- 10 Clock cycles delay using the VVC clk

UVVM™

VHDL 2008 only

UVVM methods package - target parameters

Name	Type	Example(s)	Description
vvc_target	t_vvc_target_record	UART_VVCT	VVC target type compiled into each VVC in order to differentiate between VVCs.
vvc_instance_idx	Integer	1	Instance number of the VVC used in this method
vvc_channel	t_channel	TX, RX or ALL_CHANNELS	The VVC channel of the VVC instance used in this method

UVVM methods package - functional parameters

Name	Type	Example(s)	Description
wanted_idx	natural	50	The index to be fetched or awaited
timeout	time	100 ns	The maximum time to await completion of a specified command, or all pending commands. An alert of severity ERROR will be triggered if the awaited time is equal to the specified timeout.
msg	string	“Awaiting CR from UART”	A message parameter to be appended to the log when the method is executed.
msg_id	t_msg_id	ID_SEQUENCER	The ID to enable/disable with enable/disable_log_msg(). For more info, see the UVVM-Util documentation.
result	std_logic_vector	v_result	The output where the fetched data is to be placed with fetch_result()
fetch_is_accepted	boolean	v_fetch_is_accepted	Output containing a Boolean that states if the fetch command was accepted or not. Will be false if the specified command index has not been stored.
value_is_new	boolean	v_value_is_new	Output containing a Boolean which indicates whether or not the value has been read before (if it is old or new). This value is set false after the first fetch result on this command index.
alert_level	t_alert_level	TB_WARNING	The alert level used for the alert which occurs when a fetch_result() command is not accepted
delay	time or natural	100 ns or 10	Delay to be inserted in the insert_delay() procedure, either as time or number of clock cycles

UVVM VVC Framework Common Methods details

All VVC procedures are defined in the UVVM VVC framework common methods package, td_vvc_framework_common_methods_pkg.vhdp

1 UVVM VVC Framework Common Methods details and examples

Method	Description
await_completion()	Tells the VVC to await the completion of either all pending commands or a specified command index. A message with log ID ID_IMMEDIATE_CMD_WAIT will be logged before waiting, and a message with log ID ID_IMMEDIATE_CMD will be logged at the end of the wait. The procedure will report an alert if not all commands have completed within the specified time, *timeout*. The severity of this alert will be TB_ERROR. **await_completion(vvc_target, vvc_instance, timeout, msg)** **await_completion(vvc_target, vvc_instance, wanted_idx, timeout, msg)** **await_completion(vvc_target, vvc_instance, vvc_channel, timeout, msg)** **await_completion(vvc_target, vvc_instance, vvc_channel, wanted_idx, timeout, msg)** e.g.: - await_completion(SBI_VVC, 1, 16 ns, "Await execution. For single entry queue"); - await_completion(SBI_VVC, 1, v_cmd_idx, 100 ns, "Wait for sbi_read to finish");
disable_log_msg()	Instruct the VVC to disable a given log ID. This call will be forwarded to the UVVM Utility Library disable_log_msg function. For more information about the disable_log_msg() method, please refer to the UVVM-Util QuickRef. **disable_log_msg(vvc_target, vvc_instance, msg_id, msg)** **disable_log_msg(vvc_target, vvc_instance, vvc_channel, msg_id, msg)** e.g. - disable_log_msg(VVC_SBI, 1, ID_LOG_BFM, “Disabling SBI BFM logging”); - disable_log_msg(VVC_UART, TX, 1, ID_LOG_BFM, “Disabling UART TX BFM logging”);
enable_log_msg()	Instruct the VVC to enable a given log ID. This call will be forwarded to the UVVM Utility Library enable_log_msg function. For more information about the enable_log_msg() method, please refer to the UVVM-Util QuickRef. **enable_log_msg(vvc_target, vvc_instance, msg_id, msg)** **enable_log_msg(vvc_target, vvc_instance, vvc_channel, msg_id, msg)** e.g. - enable_log_msg(VVC_SBI, 1, ID_LOG_BFM, “Enabling SBI BFM logging”); - enable_log_msg(VVC_UART, TX, 1, ID_LOG_BFM, “Enabling UART TX BFM logging”);

flush_command_queue()

Flushes the VVC command queue for the specified VVC target/channel. The procedure will log information with log ID ID_IMMEDIATE_CMD.

flush_command_queue(vvc_target, vvc_instance, msg)
flush_command_queue(vvc_target, vvc_instance, vvc_channel,msg)

e.g.
- flush_command_queue(VVC_SBI, 1, “Flushing command queue”);

fetch_result()

Fetches a stored result using the command index. A result is stored when using e.g. the read or receive commands in a VVC. The fetched result is available on the ‘result’ output. Boolean output ‘value_is_new’ is used to indicate if the returned result is a new value. The Boolean output ‘fetch_is_accepted’ is used to indicate if the fetch was successful or not. A fetch can fail if e.g. the wanted_id did not have a result to store, or the wanted_id read has not yet been executed. Omitting the ‘fetch_is_accepted’ and ‘value_is_new’ parameters causes the parameters to be checked automatically in the procedure. On successful fetch, a message with log ID ID_UVVM_CMD_RESULT is logged.

fetch_result(vvc_target, vvc_instance, wanted_id, result, msg, alert_level)
fetch_result(vvc_target, vvc_instance, vvc_channel, wanted_id, result, msg, alert_level)
fetch_result(vvc_target, vvc_instance, wanted_id, result, fetch_is_accepted, value_is_new, msg, alert_level)
fetch_result(vvc_target, vvc_instance, vvc_channel, wanted_id, result, fetch_is_accepted, value_is_new, msg, alert_level)

e.g.
- fetch_result(SBI_VVC,1, v_cmd_idx, v_data, v_is_ok, v_is_new, "Fetching read-result");

Full example:
```
sbi_read(SBI_VVCT, 1, C_ADDR_FIFO_GET, "Read from FIFO");
v_cmd_idx := shared_cmd_idx; -- Retrieve the command index
await_completion(SBI_VVCT, 1, v_cmd_idx, 100 ns, "Wait for sbi_read to finish");
fetch_result(SBI_VVCT, 1, v_cmd_idx, v_data, v_is_ok, v_is_new, "Fetching read-result");
check_value(v_is_ok, ERROR, "Readback OK via fetch_result()");
```

insert_delay()

This method inserts a delay of ‘delay’ clock cycles or ‘delay’ seconds in the VVC.

insert_delay(vvc_target, vvc_instance, vvc_channel, delay, msg)

e.g.
- insert_delay(SBI_VVC,1, 100, "100T delay");
- insert_delay(SBI_VVC,1, 50 ns, "50 ns delay");

terminate_current_command()

This method terminates the current command in the VVC, if the currently running BFM command supports the terminate signal.

terminate_current_command(vvc_target, vvc_instance, msg)
terminate_current_command(vvc_target, vvc_instance, vvc_channel, msg)

e.g.
- terminate_current_command(VVC_SBI, 1, “Terminating current command”);

terminate_all_commands()

This method terminates the current command in the VVC, if the currently running BFM command supports the terminate signal. The terminate_all_commands() procedure also flushes the VVC command queue, removing all pending commands.

terminate_all_commands(vvc_target, vvc_instance, msg)
terminate_all_commands(vvc_target, vvc_instance, vvc_channel, msg)

e.g.

- terminate_all_commands(VVC_SBI, 1, “Terminating all commands”);