

CLOCK GENERATOR VVC – Quick Reference

For general information see UVVM VVC Framework Essential Mechanisms located in `uvvm_vvc_framework/doc`.

start_clock (VVCT, vvc_instance_idx, msg, [scope])

Example: `start_clock(CLOCK_GENERATOR_VVCT, 1, "Start clock generator");`

stop_clock (VVCT, vvc_instance_idx, msg, [scope])

Example: `stop_clock(CLOCK_GENERATOR_VVCT, 1, "Stop clock generator");`

set_clock_period (VVCT, vvc_instance_idx, clock_period, msg, [scope])

Example: `set_clock_period(CLOCK_GENERATOR_VVCT, 1, 10 ns, "Change clock period to 10 ns");`

set_clock_high_time (VVCT, vvc_instance_idx, clock_high_time, msg, [scope])

Example: `set_clock_high_time(CLOCK_GENERATOR_VVCT, 1, 5 ns, "Change duty cycle to 50%");`



clock_generator_vvc.vhd

CLOCK GENERATOR VVC Configuration record `'vvc_config'` -- accessible via

shared_clock_generator_vvc_config

| Record element | Type | C_CLOCK_GENERATOR_VVC_CONFIG_DEFAULT |
|--|--------------------------------|--|
| <code>inter_bfm_delay</code> | <code>t_inter_bfm_delay</code> | <code>C_CLOCK_GENERATOR_INTER_BFM_DELAY_DEFAULT</code> |
| <code>cmd_queue_count_max</code> | <code>natural</code> | <code>C_CMD_QUEUE_COUNT_MAX</code> |
| <code>cmd_queue_count_threshold</code> | <code>natural</code> | <code>C_CMD_QUEUE_COUNT_THRESHOLD</code> |
| <code>cmd_queue_count_threshold_severity</code> | <code>t_alert_level</code> | <code>C_CMD_QUEUE_COUNT_THRESHOLD_SEVERITY</code> |
| <code>result_queue_count_max</code> | <code>natural</code> | <code>C_RESULT_QUEUE_COUNT_MAX</code> |
| <code>result_queue_count_threshold</code> | <code>natural</code> | <code>C_RESULT_QUEUE_COUNT_THRESHOLD</code> |
| <code>result_queue_count_threshold_severity</code> | <code>t_alert_level</code> | <code>C_RESULT_QUEUE_COUNT_THRESHOLD_SEVERITY</code> |
| <code>bfm_config</code> | <code>t_bfm_config</code> | <code>C_CLOCK_GENERATOR_CONFIG_DEFAULT</code> |
| <code>msg_id_panel</code> | <code>t_msg_id_panel</code> | <code>C_VVC_MSG_ID_PANEL_DEFAULT</code> |

Clock Generator VVC Status record signal `'vvc_status'` -- accessible via

shared_clock_generator_vvc_status

| Record element | Type |
|-------------------------------|----------------------|
| <code>current_cmd_idx</code> | <code>natural</code> |
| <code>previous_cmd_idx</code> | <code>natural</code> |
| <code>pending_cmd_cnt</code> | <code>natural</code> |

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()`



UVVM™
VHDL 2008 only

VVC target parameters

| Name | Type | Example(s) | Description |
|------------------|---------------------|----------------------|--|
| VVCT | t_vvc_target_record | CLOCK_GENERATOR_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 |
|-----------------|--------|-----------------------|---|
| clock_period | time | 10 ns | Clock period |
| clock_high_time | time | 5 ns | Time of the clock cycle that is '1'. Value have to be less than clock_period. |
| msg | string | "Read from DUT" | A custom message to be appended in the log/alert |
| scope | string | "CLOCK GENERATOR VVC" | A string describing the scope from which the log/alert originates. |

VVC entity signals

| Name | Type | Direction | Description |
|------|-----------|-----------|------------------|
| clk | std_logic | Output | VVC Clock signal |

VVC entity generic constants

| Name | Type | Default | Description |
|--|---------------|----------------------------------|---|
| GC_INSTANCE_IDX | natural | 1 | Instance number to assign the VVC |
| GC_CLOCK_GENERATOR_CONFIG | t_bfm_config | C_CLOCK_GENERATOR_CONFIG_DEFAULT | Configuration for the Clock Generator. |
| 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 GC_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 result 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.td_vvc_framework_common_methods_pkg` (common VVC procedures)

It is also possible to send a multicast to all instances of a VVC with `ALL_INSTANCES` as parameter for `vvc_instance_idx`.

Note: Every procedure here can be called without the optional parameters enclosed in [].

1 VVC procedure details and examples

| Procedure | Description |
|------------------------------|---|
| start_clock() | <p>start_clock(VVCT, vvc_instance_idx, msg, [scope])</p> <p>This procedure adds a <code>start_clock</code> command to the Clock Generator VVCs executor queue, which will run as soon as all preceding commands have completed. When the <code>start_clock</code> command is scheduled to run, the executor activates the clock generator process in the VVC.</p> <p>Example:</p> <pre>start_clock(CLOCK_GENERATOR_VVCT, 1, "Start clock generator", C_SCOPE);</pre> |
| stop_clock() | <p>stop_clock (VVCT, vvc_instance_idx, msg, [scope])</p> <p>This procedure adds a <code>stop_clock</code> command to the Clock Generator VVCs executor queue, which will run as soon as all preceding commands have completed. When the <code>stop_clock</code> command is scheduled to run, the executor deactivates the clock generator process in the VVC after finishing current clock cycle.</p> <p>Example:</p> <pre>stop_clock(CLOCK_GENERATOR_VVCT, 1 "Stop clock generator", C_SCOPE);</pre> |
| set_clock_period() | <p>set_clock_period (VVCT, vvc_instance_idx, clock_period, msg, [scope])</p> <p>This procedure adds a <code>set_clock_period</code> command to the CLOCK GENERATOR VVCs executor queue, which will run as soon as all preceding commands have completed. When the <code>set_clock_period</code> command is scheduled to run, the executor will change the clock period instantly.</p> <p>Examples:</p> <pre>set_clock_period(CLOCK_GENERATOR_VVCT, 1, 10 ns, "Change clock period to 10 ns", C_SCOPE);</pre> |
| set_clock_high_time() | <p>set_clock_high_time (VVCT, vvc_instance_idx, clock_high_time, msg, [scope])</p> <p>This procedure adds a <code>set_clock_high_time</code> command to the CLOCK_GENERATOR VVCs executor queue, which will run as soon as all preceding commands have completed. When the write command is scheduled to run, the executor changes the clock high time and the change will take effect from the next rising edge.</p> <p>Examples:</p> <pre>set_c_clock_high_time(CLOCK_GENERATOR_VVCT, 1, 6 ns, "Changing the duty cycle to 60%", C_SCOPE);</pre> |

2 VVC Configuration

| Record element | Type | C_CLOCK_GENERATOR_CONFIG_DEFAULT | Description |
|---------------------------------------|-------------------|---|---|
| inter_bfm_delay | t_inter_bfm_delay | C_CLOCK_GENERATOR_INTER_BFM_DELAY_DEFAULT | Delay between any requested BFM accesses towards the DUT. - TIME_START2START: Time from a BFM start to the next BFM start (A TB_WARNING will be issued if access takes longer than TIME_START2START). - TIME_FINISH2START: Time from a BFM end to the next BFM start. Any insert_delay() command will add to the above minimum delays, giving for instance the ability to skew the BFM starting time. |
| 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 result 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_bfm_config | C_CLOCK_GENERATOR_CONFIG_DEFAULT | Configuration for Clock Generator. |
| 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_clock_generator_vvc_config(1).inter_bfm_delay.delay_in_time := 50 ns;
```

3 VVC Status

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

| Record element | 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 |

4 Activity watchdog

The VVCs support an activity watchdog which monitors VVC activity and will alert if no VVC activity is registered within a selected timeout value. The VVCs will register their presence to the activity watchdog at start-up, and report when busy and not, using dedicated activity watchdog methods and triggering the global_trigger_activity_watchdog signal, during simulations.

Include activity_watchdog(num_exp_vvc, timeout, alert_level, msg) in the testbench to start using the activity watchdog.

Note that the clock generator VVC is included in the total registered VVCs in the activity watchdog register, but its activity is not included in the resetting of the inactivity timeout counter. More information can be found in UVVM Essential Mechanisms PDF in the UVVM VVC Framework doc folder.

5 Additional Documentation

Additional documentation about UVVM and its features can be found under “/uvvm_vvc_framework/doc/”.

6 Compilation

The Clock Generator VVC must be compiled with VHDL 2008.

It is dependent on the following libraries

- **UVVM Utility Library (UVVM-Util), version 2.11.0 and up**
- **UVVM VVC Framework, version 2.7.1 and up**

Before compiling the Clock Generator VVC, assure 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 Clock Generator VVC:

| Compile to library | File | Comment |
|----------------------------|--|---|
| bitvis_vip_clock_generator | vvc_cmd_pkg.vhd | Clock Generator VVC command types and operations |
| bitvis_vip_clock_generator | ../uvvm_vvc_framework/src_target_dependent/td_target_support_pkg.vhd | UVVM VVC target support package, compiled into the Clock Generator VVC library. |
| bitvis_vip_clock_generator | ../uvvm_vvc_framework/src_target_dependent/td_vvc_framework_common_methods_pkg.vhd | Common UVVM framework methods compiled into the Clock Generator VVC library |
| bitvis_vip_clock_generator | vvc_methods_pkg.vhd | Clock Generator VVC methods |
| bitvis_vip_clock_generator | ../uvvm_vvc_framework/src_target_dependent/td_queue_pkg.vhd | UVVM queue package for the VVC |
| bitvis_vip_clock_generator | ../uvvm_vvc_framework/src_target_dependent/td_vvc_entity_support_pkg.vhd | UVVM VVC entity support compiled into the Clock Generator VVC library |
| bitvis_vip_clock_generator | clock_generator_vvc.vhd | Clock Generator VVC |

7 Simulator compatibility and setup

See README.md for a list of supported simulators.

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