1 Method descriptions

Note 1: Arguments common for most methods (green text) are described in chapter 1.11.
Note 2: All methods are defined in uvvm_util.methods_pkg, unless otherwise noted.

Legend: bool=boolean, sl=std_logic, slv=std_logic_vector, u=unsigned, s=signed, int=integer
**IEEE=Method is native for VHDL2008 (Method is listed here for completeness.)*

1.1 Checks and awaits

Name	Parameters and examples	Description

[v_bool :=]
check_value()
	value(bool), [exp(bool)], alert_level, msg, [scope, [msg_id, [msg_id_panel]]]
	value(sl), exp(sl), [match_strictness], alert_level, msg, [scope, [msg_id, [msg_id_panel]]]
	value(slv), exp(slv), [match_strictness], alert_level, msg, [scope, [radix, [format, [msg_id, [msg_id_panel]]]]]
	value(u), exp(u), alert_level, msg, [scope, [radix, [format, [msg_id, [msg_id_panel]]]]]
	value(s), exp(s), alert_level, msg, [scope, [radix, [format, [msg_id, [msg_id_panel]]]]]
	value(int), exp(int), alert_level, msg, [scope, [msg_id, [msg_id_panel]]]
	value(real), exp(real), alert_level, msg, [scope, [msg_id, [msg_id_panel]]]
	value(time), exp(time), alert_level, msg, [scope, [msg_id, [msg_id_panel]]]
	
	Examples
	check_value(v_int_a, 42, WARNING, "Checking the integer");
	v_check := check_value(v_slv5_a, "11100", MATCH_EXACT, ERROR, "Checking the SLV", "My Scope", HEX, SKIP_LEADING_0, ID_SEQUENCER, shared_msg_id_panel);
		Checks if *val* equals *exp*, and alerts with severity *alert_level* if the values do not match.
		The result of the check is returned as a boolean if the method is called as a function.
		If *val* is of type *slv*, *unsigned* or *signed*, there are additional optional arguments:
		- *match_strictness:* Specifies if match needs to be exact or std_match, e.g. 'H' = '1'. (MATCH_EXACT, MATCH_STD)
		- *radix*: For the vector representation in the log: BIN, HEX, DEC or HEX_BIN_IF_INVALID. (HEX_BIN_IF_INVALID means hexadecimal, unless there are the vector contains any U, X, Z or W, - in which case it is also logged in binary radix.)
		- *format:* KEEP_LEADING_0 or SKIP_LEADING_0. Controls how the vector is formatted in the log.
		Defaults: *scope*<=C_TB_SCOPE_DEFAULT, *match_strictness*<=MATCH_STD, *radix*<=HEX_BIN_IF_INVALID, *format*<=SKIP_LEADING_0, *msg_id*<=ID_POS_ACK, *msg_id_panel*<=shared_msg_id_panel

[v_bool :=] check_value_in_range()	value(u), min_value(u), max_value(u), alert_level, msg, [scope, [msg_id, [msg_id_panel]]] value(s), min_value(s), max_value(s), alert_level, msg, [scope, [msg_id, [msg_id_panel]]] value(int), min_value(int), max_value(int), alert_level, msg, [scope, [msg_id, [msg_id_panel]]] value(time), min_value(time), max_value(time),alert_level, msg, [scope, [msg_id, [msg_id_panel]]] value(real), min_value(real), max_value(real), alert_level, msg, [scope, [msg_id, [msg_id_panel]]] **Example** check_value_in_range(v_int_a, 10, 100, ERROR, "Checking that integer is in range");	Checks if *min_value* ≤ *val* ≤ *max_value*, and alerts with severity *alert_level* if *val* is outside the range. The result of the check is returned as a boolean if the method is called as a function. **Defaults**: *scope*<=C_TB_SCOPE_DEFAULT, *msg_id*<=ID_POS_ACK, *msg_id_panel*<=shared_msg_id_panel
check_stable()	target(bool), stable_req(time), alert_level, msg, [scope, [msg_id, [msg_id_panel]]] target(sl), stable_req(time), alert_level, msg, [scope, [msg_id, [msg_id_panel]]] target(slv), stable_req(time), alert_level, msg, [scope, [msg_id, [msg_id_panel]]] target(u), stable_req(time), alert_level, msg, [scope, [msg_id, [msg_id_panel]]] target(s), stable_req(time), alert_level, msg, [scope, [msg_id, [msg_id_panel]]] target(int), stable_req(time), alert_level, msg, [scope, [msg_id, [msg_id_panel]]] target(real), stable_req(time), alert_level, msg, [scope, [msg_id, [msg_id_panel]]] **Example** check_stable(slv8, 9 ns, ERROR, "Checking if SLV is stable");	Checks if the *target* signal has been stable in *stable_req* time. If not, an alert is asserted. **Defaults**: *scope*<=C_TB_SCOPE_DEFAULT, *msg_id*<=ID_POS_ACK, *msg_id_panel*<=shared_msg_id_panel