FIFO Collection - Quick Reference

UVVM Support Component

The FIFO Collection is a memory buffer that can be used to hold one or more FIFOs. Each FIFO will be allocated a chosen size and ID number. This allows a selectable number of FIFOs to operate individually and be independently accessed.

uvvm_fifo_init ([buffer_idx,] buffer_size_in_bits)

Example: v_buffer_idx := uvvm_fifo_init(C_BUFFER_SIZE-1); -- returns a buffer index on initialization
Example: uvvm_fifo_init(C_BUFFER_IDX_1, C_BUFFER_SIZE-1); -- buffer index is selected

uvvm_fifo_init ([buffer_idx,] buffer_size_in_bits)

Example: v_buffer_idx := uvvm_fifo_init(C_BUFFER_SIZE-1); -- returns a buffer index on initialization
Example: uvvm_fifo_init(C_BUFFER_IDX_1, C_BUFFER_SIZE-1); -- buffer index is selected

uvvm_fifo_get (buffer_idx, entry_size_in_bits)

Example: v_rx_data := uvvm_fifo_get (C_BUFFER_IDX_1, C_ENTRY_SIZE_1);

uvvm_fifo_flush (buffer_idx)

Example: uvvm_fifo_flush(C_BUFFER_IDX_1);

uvvm_fifo_peek (buffer_idx, entry_size_in_bits)

Example: v_rx_data := uvvm_fifo_peek(C_BUFFER_IDX_1, C_ENTRY_SIZE_1);

uvvm_fifo_get_count (buffer_idx)

Example: v_num_elements := uvvm_fifo_get_count(C_BUFFER_IDX_1);

uvvm_fifo_get_max_count (buffer_idx)

Example: v_max_fifo_elements := uvvm_fifo_get_max_count(C_BUFFER_IDX_1);

uvvm_fifo_is_full (buffer_idx)

Example: v_fifo_is_full := uvvm_fifo_is_full (C_BUFFER_IDX_1);

uvvm_fifo_deallocate (VOID)

Example: uvvm_fifo_deallocate(VOID);

UVVM™

VHDL 2008 only

FIFO Collection – Functional parameters

Name	Type	Example(s)	Description
buffer_idx	natural	1	The index of the FIFO that shall be initialized.
buffer_size_in_bits	natural	1024	The size of the FIFO.
data	SLV	v_rx_data	The data that shall be pushed to the FIFO.

FIFO Collection details

All FIFO functions and procedures are defined in the UVVM Data FIFO package, ti_data_fifo_pkg.vhd

1 FIFO Collection details and examples

Method	Description

uvvm_fifo init()	**uvvm_fifo_init([buffer_idx,] buffer_size_in_bits)**
	
	This UVVM FIFO call will allocate space in the FIFO buffer. If no buffer_idx is given, the call will return a buffer index for use when addressing the FIFO. Note that 0 will be returned on error. If a buffer_idx is given, the FIFO is initialized with this index.
	
	Example:
	`uvvm_fifo_init(C_BUFFER_IDX_1, C_BUFFER_SIZE-1); -- initialize buffer with index C_BUFFER_IDX_1`
	`v_fifo_idx := uvvm_fifo_init(C_BUFFER_SIZE-1);`

uvvm_fifo_put()	**uvvm_fifo_put(buffer_idx, data)** This procedure puts data into a FIFO with index buffer_idx. The size of the data is unconstrained, meaning that it can be any size. Pushing data with a size that is larger than the FIFO size results in wrapping, i.e., that when reaching the end that data remaining will overwrite the data that was first written. Example: `uvvm_fifo_put(C_BUFFER_IDX_1, v_rx_data);`
uvvm_fifo_get()	**uvvm_fifo_get(buffer_idx, entry_size_in_bits)** This function returns the data from the FIFO and removes the returned data from the FIFO. Note that buffer_idx is the index of the FIFO that shall be read, and that entry_size_in_bits is the size of the returned data as SLV. Attempting to get data from an empty FIFO is allowed but triggers a TB_WARNING and returns garbage data. Attempting to get a larger value than the FIFO size is allowed but triggers a TB_WARNING. Example: `v_rx_data := uvvm_fifo_get(C_BUFFER_IDX_1, C_ENTRY_SIZE-1);`

uvvm_fifo_flush()

uvvm_fifo_flush(buffer_idx)

This procedure empties the FIFO given by buffer_idx.

Example:
```
uvvm_fifo_flush(C_BUFFER_IDX_1);
```

uvvm_fifo_peek()

uvvm_fifo_peek(buffer_idx, entry_size_in_bits)

This function returns the data from the FIFO without removing it.
Note that, apart from not removing the data, this function will behave in the same way as the uvvm_fifo_get() function.

Example:
```
v_rx_data := uvvm_fifo_peek(C_BUFFER_IDX_1, C_ENTRY_SIZE-1);
```

uvvm_fifo_get_count()

uvvm_fifo_get_count(buffer_idx)

This function returns a natural indicating the number of elements currently occupying the FIFO given by buffer_idx.

Example:
```
v_num_elements := uvvm_fifo_get_count(C_BUFFER_IDX);
```

uvvm_fifo_get_max_count()

uvvm_fifo_get_max_count(buffer_idx)

This function returns a natural indicating the maximum number of elements that can occupy the FIFO given by buffer_idx.

Example:
```
v_max_elements := uvvm_fifo_get_max_count(C_BUFFER_IDX);
```

uvvm_fifo_is_full()

uvvm_fifo_is_full(buffer_idx)

This function returns a boolean indicating if the FIFO is full or not.

Example:
```
v_fifo_is_full := uvvm_fifo_is_full(C_BUFFER_IDX);
```

uvvm_fifo_deallocate()

uvvm_fifo_deallocate(VOID)

This function deallocates the FIFO buffer, all the FIFO pointers are reset.

Example:
```
uvvm_fifo_deallocate(VOID);
```