FIFO buffer operates as follows:

Write Operation:

- When writeEn is asserted (set to 1) and the FIFO is not full (full is 0), the data present on writeData is written into the memory array in one clock cycle at the location indicated by wrPtr.
- The write pointer (wrPtr) is then incremented by one in the next clock cycle to point to the next write location.
- If the write pointer reaches the end of the memory array, it wraps around to the beginning, maintaining the circular nature of the FIFO.

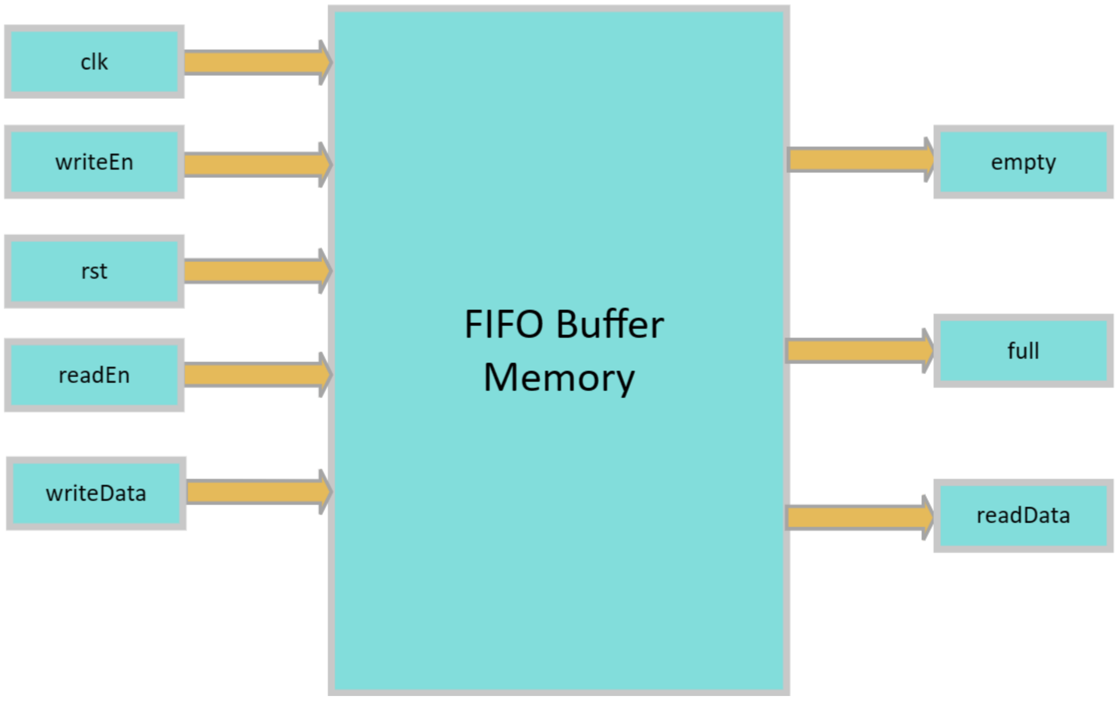
Read Operation:

- When readEn is asserted (set to 1) and the FIFO is not empty (empty is 0), the data at the location indicated by rdPtr is read out and presented on readData in a combinational way using continuous assignment.
- The read pointer (rdPtr) is then incremented by one in the next clock cycle to point to the next read location.
- Like the write pointer, if the read pointer reaches the end of the memory array, it wraps around to the beginning.

Full and Empty Flags:

- Empty Flag (empty): This is one when no write operation was performed and when the same amount of write and read operations was performed.
- Full Flag (full): If the difference between the write and read operations is equal to the memory size, the full flag is active.

Block Diagram:



Input Definitions

- clk: Clock signal.
- rst: Reset signal.
- writeEn: Write enable signal.
- writeData: Data input for writing.
- readEn: Read enable signal.

Output Definitions.

- readData: Data output for reading.
- full: Full flag indicating if the FIFO is full.
- empty: Empty flag indicating if the FIFO is empty.

Internal Components:

- Memory Array: Stores the data elements.
- Write Pointer: Points to the current write location.
- Read Pointer: Points to the current read location.
- Full Flag: Indicates when the FIFO is full.

- Empty Flag: Indicates when the FIFO is empty.

Formal Verification:

Verify the correct write operation.

Properties defined to cover this spec:

Assumptions:

- Assume write enable is not active when full is active: writEnoff_full.
- Assume write enable is not active when rst is active: writeEnOff_rst_on.

Assertions:

- The property assures that if FIFO is full then write enable signal must not be active: full_notWriteEn.
- This property verifies writeData was written correctly when the writeEn is activated: write_correctly.
- The property assures that write pointer increments when a write operation happens: wrPtr_increm_writeEn_on.
- The property assures that wrPtrNext increments when a write operation happens: wrPtrNext_increm_writeEn_on.
- The property assures that when wrPtr reaches max value wraps around to 0: wrPtr_maxvalue_reset0.
- The property assures that when wrPtrNext reaches max value wraps around to 0: wrPtrNext_maxvalue_reset0.
- The property assures that FIFO memory value is stable if writeEn is not active: fifo_stable_when_writeEnoff.
- The property assures that wrPtr is stable if a write doesn't occur: wr_en_off_wr_ptr_stable.
- The property assures that after reset the read and write pointers must have the same value and be 0: rst_rdPtr_wrPtr_zero.

Covers:

- All the memory was written write_all_address.
- What if writeEn is active while FIFO is full: writeEn_fifo_full.

Verify the correct read operation:

Properties defined to cover this spec:

Assumptions:

- Assume write enable is not active when full is active: readEnoff_empty.
- Assume write enable is not active when rst is active: readEnOff_rst_on.

Assertions:

- The property assures that if FIFO is empty then read enable signal must not be active: empty_notReadEn.
- This property verifies readData was read correctly when the readEn is activated: read_correctly.
- The property assures that read pointer increments when a read operation happens: rdPtr_increm_ readEn_on.
- The property assures that rdPtrNext increments when a read operation happens: rdPtrNext_increm_ readEn_on.
- The property assures when rdPtr reaches max value wraps around to 0: rdPtr_maxvalue_reset0.
- The property assures that when rdPtrNext reaches max value wraps around to 0: rdPtrNext_maxvalue_reset0.
- The property assures that rdPtr is stable if a read doesn't occur: rd_en_off_rd_ptr_stable.
- After reset is active read enable is off until a write operation happens: This property is evaluated through other assertions, we make sure that reading never happens when the memory is empty, and also make sure that the empty flag is initialized high when resetting.

Covers

- All the memory was read: read_all_address.
- What if readEn is active while FIFO is empty: readEn_fifo_empty.

Verify the correct full signal:

Assertions:

- The property assures that full signal is off after reset: full_off_whenreset.
- The full and empty flags can never be active at the same time: never_full_and_empty.

Covers:

- Cover that the FIFO becomes full: fifo_full.
- Cover that the write enable signal is asserted when the FIFO is not full: fifo_notFull.
- Cover a sequence when FIFO becomes full and then no full: fifo_full_no_full.

Verify the correct empty signal:

Assertions:

- The property assures that empty signal is active after reset: empty_on_whenreset.

Covers:

- fifo_empty: Cover that the FIFO becomes empty.
- fifo_notEmpty: Cover that the read enable signal is asserted when the FIFO is not empty.
- fifo_empty_no_empty: Cover a sequence when FIFO becomes empty and then no empty.

Verify the correct simultaneous write and read operation:

Properties defined to cover this spec:

Covers:

- Read and write at the same time: write_and_read.
- Read and write at the same time while the memory is full: write_and_read_mem_full.
- Read and write at the same time while the memory is empty: write_and_read_mem_empty.