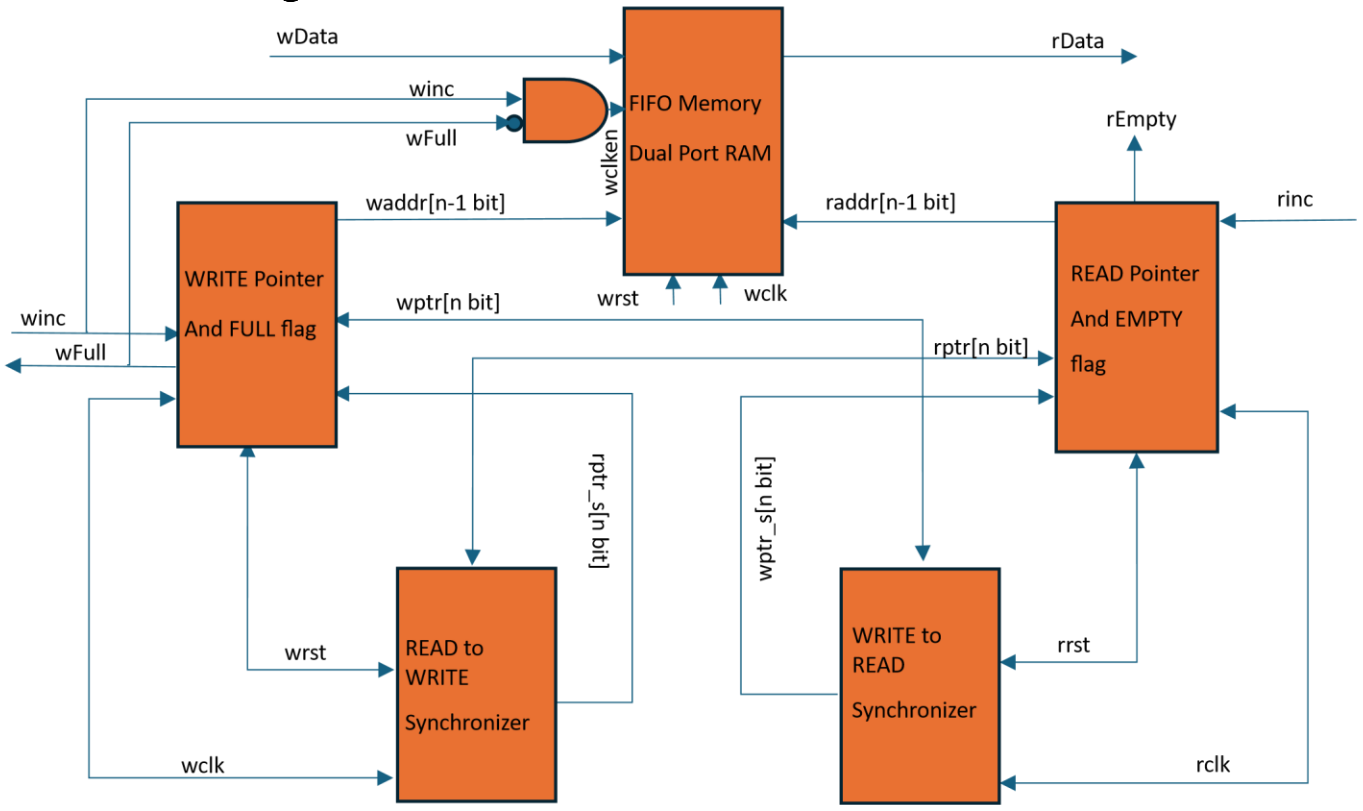
Design Specification

Block diagram of ASYNCHRONOUS FIFO:

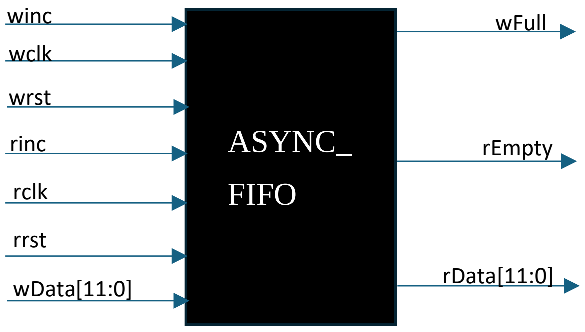


Detailed Block diagram of Asynchronous FIFO

Module description:

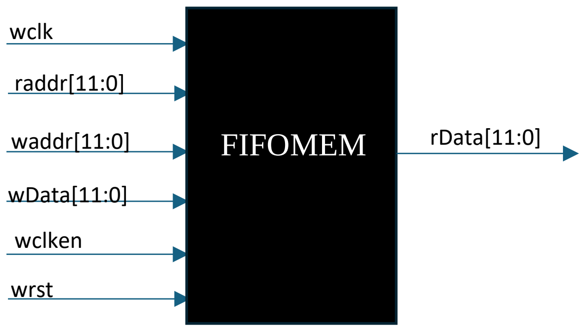
1) **ASYNC_FIFO**: The top-level module instantiates all the other modules used in the complete asynchronous FIFO design. The instance includes fifomem, synchronizer_r2w, synchronizer_w2r, rptr_handler, and wptr_handler modules.

   Block diagram (Black box/PIN level diagram) of ASYNC_FIFO:



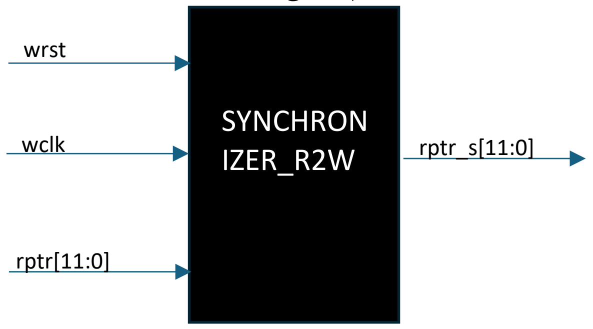
2) **FIFOMEM:** This module acts as the FIFO memory buffer which is accessed by both the write and read clock domains. This buffer is instantiated synchronous dual-port RAM.

Block diagram (Black box/PIN level diagram) of FIFOMEM:



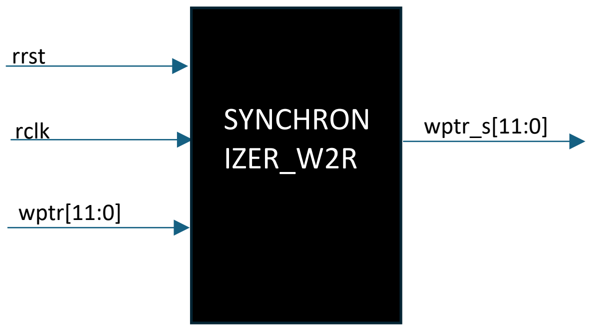
3) **SYNCHRONIZER_R2W**: This is a simple synchronizer module, used to pass an n-bit pointer from the read clock domain to the write clock domain, through a pair of registers that are clocked by the FIFO write clock.

Block diagram (Black box/PIN level diagram) of SYNCHRONIZER_R2W:



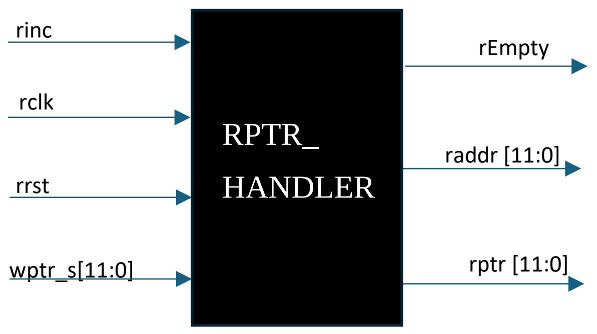
4) **SYNCHRONIZER_W2R**: This is a simple synchronizer module, used to pass an n-bit pointer from the write clock domain to the read clock domain, through a pair of registers that are clocked by the FIFO read clock. The synchronized write pointer will be used by the rptr_handler module to generate the FIFO empty condition.

Block diagram (Black box/PIN level diagram) of SYNCHRONIZER_W2R:



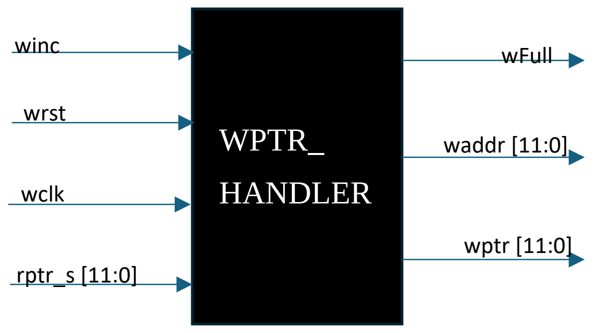
5) **RPTR_HANDLER**: This module has all the FIFO logic that is generated within the read clock domain. The read pointer is a dual n-bit Gray code counter. The rptr is passed to the write clock domain through the SYNCHRONIZER_R2W module. The raddr is used to address the FIFO memory buffer. The FIFO empty output is registered and is asserted on the next rising rclk edge when the next rptr value equals the synchronized wptr value. This module is synchronous with the rclk.

Block diagram (Black box/PIN level diagram) of RPTR_HANDLER:



6) **WPTR_HANDLER**: Similar to RPTR_HANDLER, this module has all the FIFO logic that is generated within the write clock domain. The write pointer is a dual n-bit Gray code counter. The wptr is passed to the read clock domain through the SYNCHRONIZER_W2R module. The waddr is used to address the FIFO buffer. The FIFO full output is registered and is asserted on the next rising wclk edge when the next modified wgraynext value equals the synchronized and modified rptr_s value. This module is synchronous to the wclk.

Block diagram (Black box/PIN level diagram) of WPTR_HANDLER:



Signals and its descriptions:

Signals	**Description**
wclk	Write clock: This clock is associated with the write operation and determines the timing for writing data into the FIFO.
rclk	Read clock: This clock is associated with the read operation and determines the timing for reading data from the FIFO.
rrst	Read Reset (active Low): Asynchronous reset signal for read operation. When asserted (low), it resets the read-related logic in the FIFO.
wrst	Write Reset (active low): Asynchronous reset signal for write operation. When asserted (low), it resets the write-related logic in the FIFO.
wData	Write Data: contains the data to be written into the FIFO when a write operation is initiated.
wFull	FIFO Is Full (write): Indicates whether the FIFO is full. Asserted when the FIFO has reached its maximum capacity and cannot accept more data.
rEmpty	FIFO Is Empty (read): Indicates whether the FIFO is empty. Asserted when there is no data available for reading from the FIFO.
rData	Read Data: Data read from the FIFO when a read operation is initiated.
wAddr	Write Address: Address signal specifying where the incoming written data should be stored in the FIFO.
wPtr	Write Pointer: Points to the location in the FIFO where the next write operation will occur.
wbinnext	Next write Address: Represents the next address in binary where incoming write data will be stored after the current write operation.
wgraynext	Write Pointer Next: Represents the next location in Gray code in the FIFO where the next write operation will occur after the current write operation.
rAddr	Read Address: Addresses the signal specifying where the next read data should be retrieved from in the FIFO.

rPtr	Read Pointer: Points to the location in the FIFO from which the next read operation will occur.
rbinnext	Next Read Address: Represents the next address in binary from which read data will be retrieved after the current read operation.
rgraynext	Read Pointer Next: Represents the next location in the FIFO in Gray code from which the next read operation will occur after the current read operation.
rPtr_s	Read Pointer Synchronized: Represents the delay between a write operation and the corresponding read operation. Ensures proper synchronization.
wPtr_s	Write Pointer Synchronized: Represents the delay between a read operation and the corresponding write operation. Ensures proper synchronization.

FIFO Depth Calculations:

Sample design for the above Asynchronous FIFO used as an interface circuitry.

Producer Clk frequency (clk 1): **120** MHz

Consumer Clk frequency (clk 2): **50** MHz

Maximum write burst size: **1024**.

Duty cycle: 50%

No. of Idle cycle between successive writes = **3**

No. of Idle cycle between successive reads = **2**

No. of idle cycles between successive writes is 3 clock cycles. It means that after writing one data, the Producer block is waiting for 3 clock cycles to initiate the next write. So, it can be understood that for every 4 clock cycles, one data is written.

The no. of idle cycles between two successive reads is 2 clock cycles, it means that after reading one data, the consumer block is waiting for 2 clock cycles to initiate the next read. So, it can be understood that for every 3 clock cycles, one data is read.

Time required to write one data item: 4 x (1/120MHz) = 33nsec

Time required to write all the data in the burst = 1024 x 33nsec= 34099nsec

Time required to read one data item = 3 x (1/50MHz) = 60nsec

So, for every 60 n, the consumer block is going to read one data in the burst. So, in a period of 34099nsec, a maximum 1024 number of data items can be written.

Therefore, the number of data items can be read in a period of 34099nsec = 34099nsec/60nsec=568

The remaining number of bytes to be stored in FIFO = 1024-568 = 456 so a depth of **9** is required.