Synchronous FIFO

1, Description

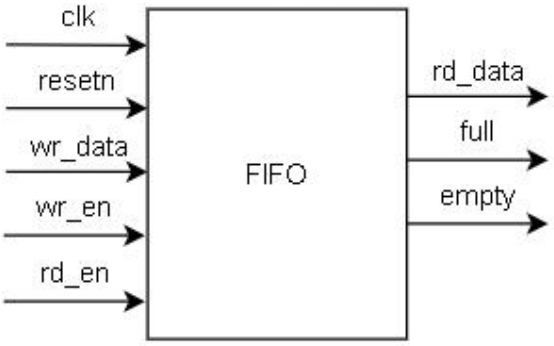
· FIFO (First In, First Out) is a buffer in which data written first will be read out first.

· FIFO receives signals including write data, read/write control signals, and returns read data along with full/empty memory status signals.

2, Parameter, input and output signals

Parameter	Value	Description
DATAWIDTH	Default: 8	Data width in bit
DEPTH	Default: 16	The maximum number of data elements that the FIFO can store.

Signal	Width	I/O	Description
clk	1	Input	System clock signal
resetn	1	Input	System asynchronous reset, active low
wr_data	DATAWIDTH	Input	Data write to FIFO
wr_en	1	Input	Request to write
rd_en	1	Input	Request to read
rd_data	DATAWIDTH	Output	Data read from FIFO Init value: 0
empty	1	Output	Set to indicate that FIFO is empty Init value: 1
full	1	Output	Set to indicate that FIFO is full Init value: 0



FIFO block diagram

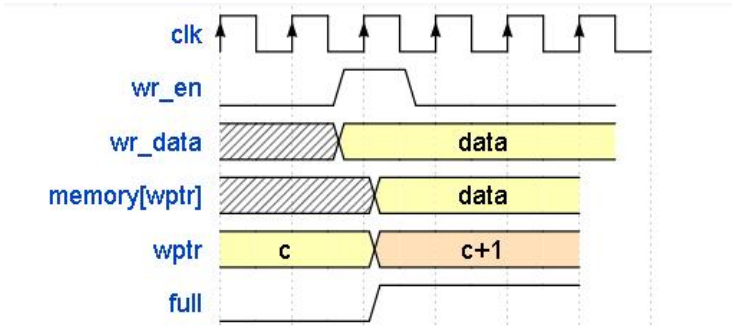
3, Design

Local signal	Width	Reset value	Description
memory	DEPTH * DATAWIDTH	0	Internal memory of FIFO
data_cnt	$clog(DEPTH) +1	0	Number of register in use
wptr	$clog(DEPTH)	0	Write pointer
rptr	$clog(DEPTH)	0	Read pointer

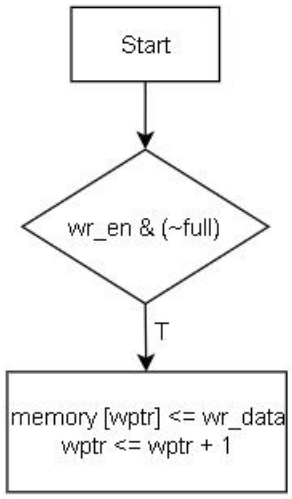
- FIFO has a internal memory array (memory).

- FIFO uses two pointers, rptr (read pointer) and wptr (write pointer), to store the read and write locations in memory, and a register data_cnt to keep track of the number of registers in use.

- FIFO write data from wr_data to memory when wr_en is set and FIFO is not full. When data is writen to memory, wptr increases by 1, data_cnt increases by 1.

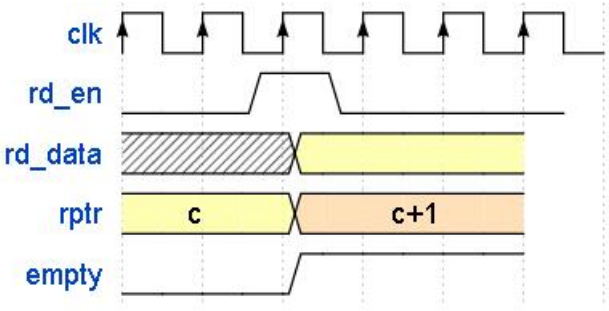


Write data time diagram

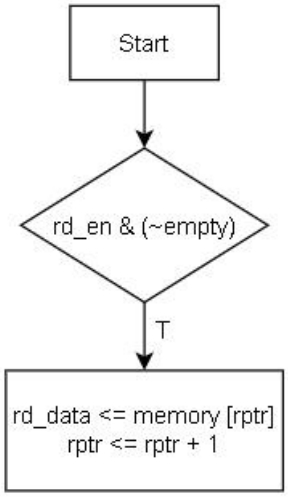


Write control flow

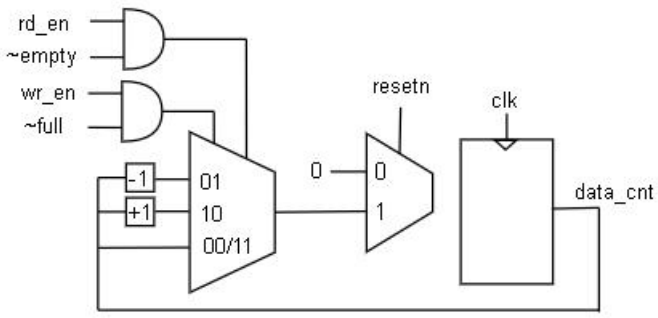
- FIFO read data from memory to rd_data when rd_en is set and FIFO is not empty. When data is read, rptr increases by 1, data_cnt decreases by 1.



Read data time diagram



Read control flow



Logic update data_cnt

- Empty flag is set when data_cnt is equal to 0, full flag is set when data_cnt is equal to DATAWIDTH

```
assign full = (data_cnt == DEPTH)

assign empty = (data_cnt == 0)
```