S8_15-09-2021

15 September 2021 08:51

⊛ Assignment : A.3

⊛ Synthesize, Simulate & Download VHDL Model for FIFO :-

⇒ F.I.F.O = First - In - First - Out

⇒ It is a Memory

⇒ Memory Organisation :-

⇒ Consider a Memory which can store 'P' no. of bits

⇒ These 'P' bits which be distributed in 'P' No. of memory cells

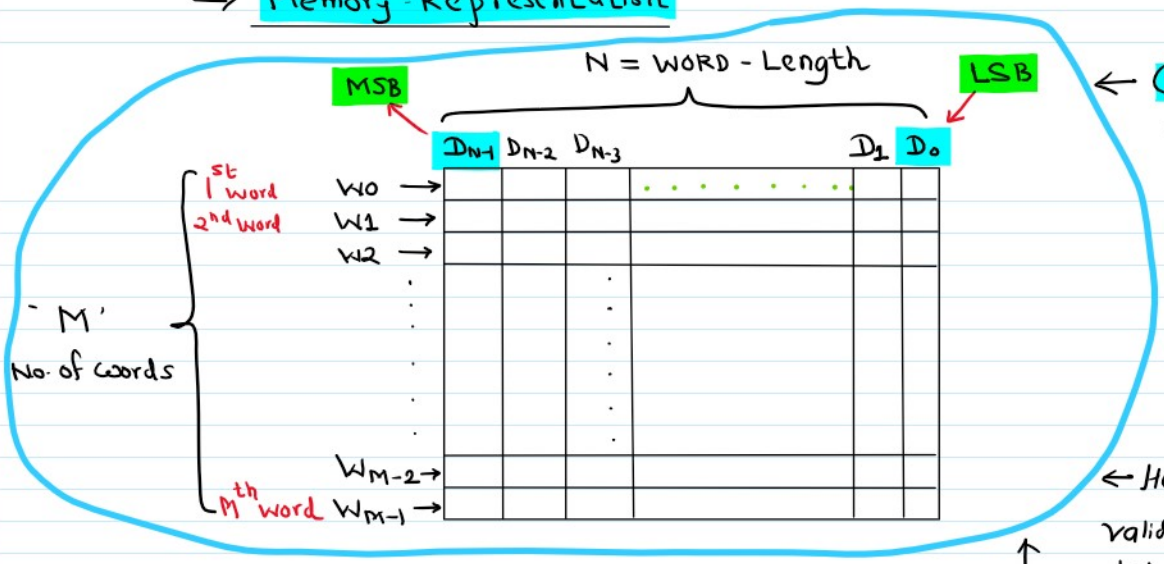
⇒ These 'P' cells will be organised as (M ROWS) X (N COLUMNS)

⇒ (M X N) is known as Memory-Layout / Memory organisation

⇒ M = No. of WORDS (Vertical-spread of Memory)

⇒ N = WORD-Length (Horizontal-spread of Memory)

⇒ **Memory-Representation**



← (M X N) Structure

To be created because there is No Predefined (M X N) Structure in VHDL

← Has to be given a valid-name

Let 'mem' be the name

⇒ Thus MEMORY is a 2D-element

⇒ This MEMORY (M X N) can thus be visualised as a VERTICAL-STACKING of 'M', N-bit Registers

⇒ We know the VHDL-statement for creating N-bit Register as :-

Signal_name : <MODE> STD_LOGIC_VECTOR (N-1 DOWNTO 0);

⇒ If we want to Create 'M' such N-bit Registers ; the VHDL Statement is :-

⊛ To Create an (M X N) 2D-Structure in VHDL ; Syntax is

TYPE Structure_name IS ARRAY (0 TO M-1) OF STD_LOGIC_VECTOR (N-1 DOWNTO 0);

↓

TYPE Structure_name IS ARRAY (0 to M-1) OF STD_LOGIC_VECTOR(N-1 DOWNTO 0);

- Structure_name: ↓ User-defined valid-name
- ARRAY (0 to M-1): Creates Vertical Spread of size 'M'
- STD_LOGIC_VECTOR(N-1 DOWNTO 0): Creates Horizontal Spread of size 'N'

⇒ So, the Syntax for creating 32-bit Memory-structure organised as (4 x 8) will be as follows:-

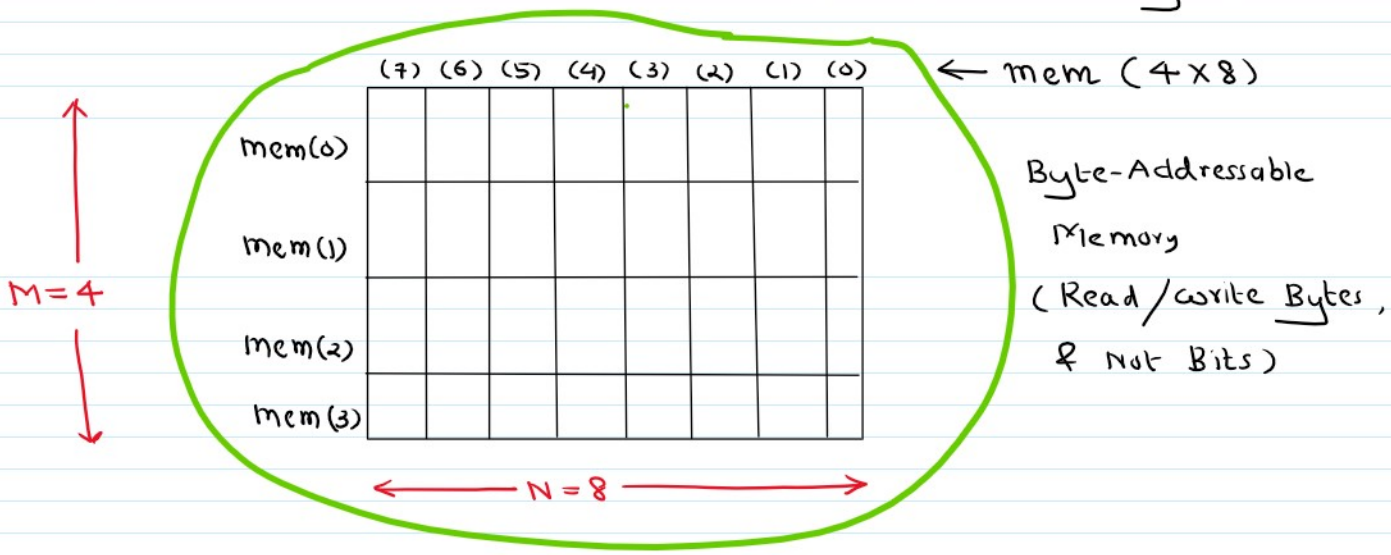
```
TYPE mem IS ARRAY (0 TO 3) OF STD_LOGIC_VECTOR (7 DOWNTO 0);
```

⇒ So, the Syntax for creating 4096-bit Memory-structure organised as 512 words of 1-Byte each is as follows:-

```
TYPE mem IS ARRAY (0 TO 511) OF STD_LOGIC_VECTOR (7 DOWNTO 0);
```

```
TYPE mem IS ARRAY (0 TO 3) OF STD_LOGIC_VECTOR (7 DOWNTO 0);
```

⇒ SYNTHESIS of the above VHDL-statement infers the following H/w:-

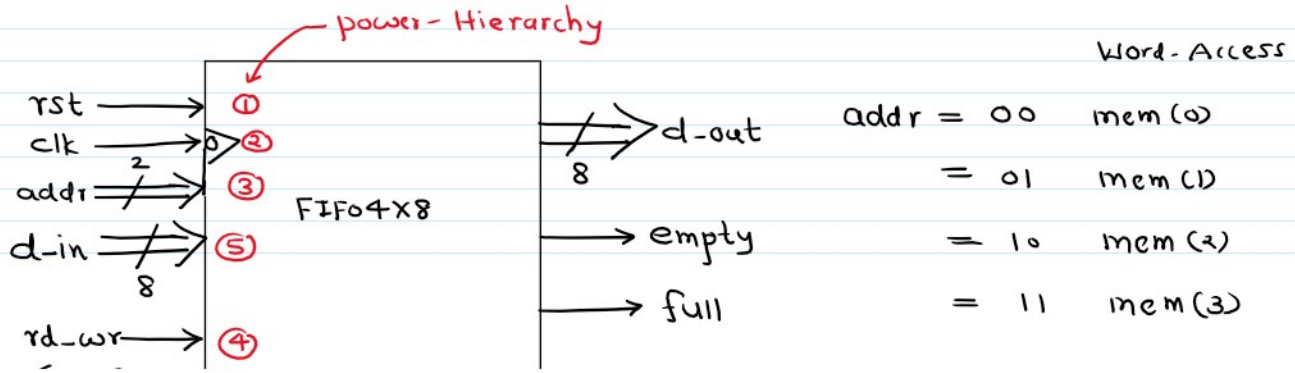


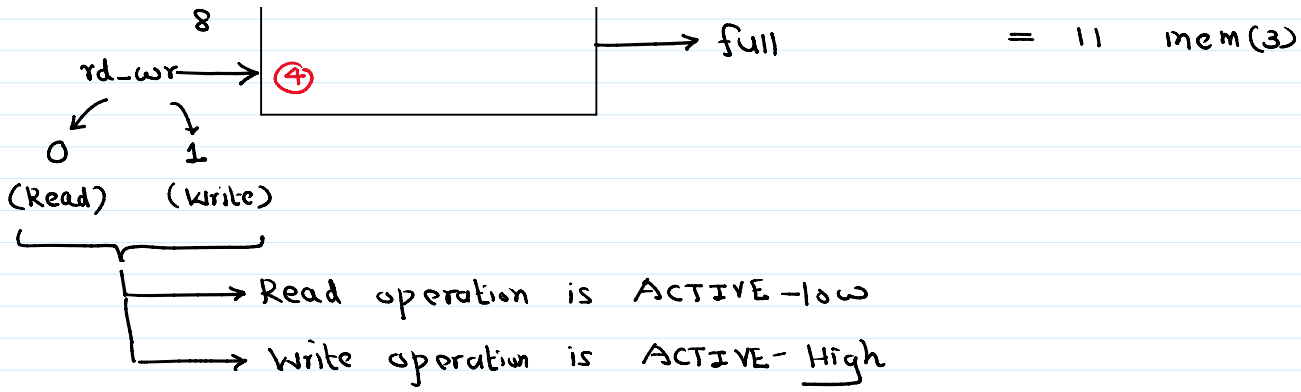
⇒ Real FIFo = (only Memory pointers, No-Address)

⇒ 2 Flags which indicate whether FIFo = Empty/Full.

⇒ It has data i/p, data o/p (8-bit)

⇒ Block-Diagram:-





Function-Table :-

rst	clk	addr	rd_wr	d_out	empty	full	
1	X	X	X	$(00)_{16}$	1	0	← Reset Action
0	↓	00	0	mem(0)	0	1	Read-operation
↓	:	↓	↓	:	:	:	
0	↓	11	0	mem(3)	0	1	
0	↓	00	1	↑	0	0	Write-operation
:	:	↓	↓	NA	0	0	
0	↓	11	1	↓	0	1	

TVM :- I/ps = rst, clk, addr, rd_wr.

Seperate process ⇒ ✓ ✓ ✓

↓

Created by Tool