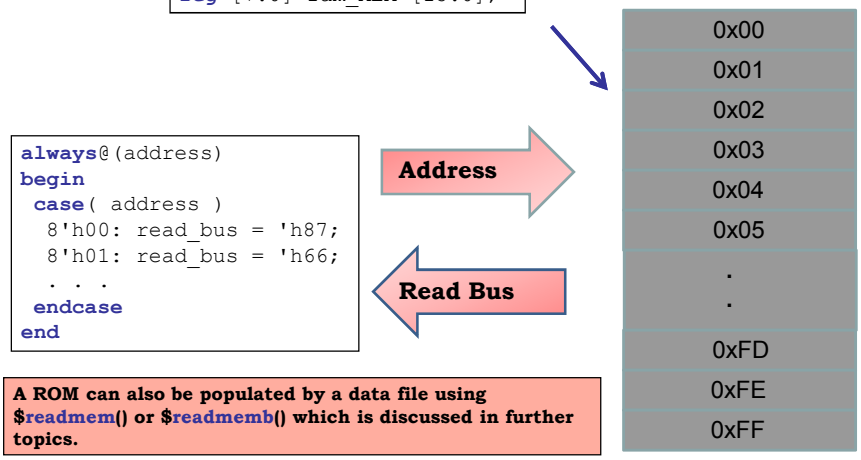
Session #8

Modeling memories in Verilog

- Memories in Verilog are modeled using 2 dimensional arrays.
- Read operation accesses a particular element in the array and puts the data in that location on the read data bus.
- Write operation writes into the data on the write data bus into the given location in the array

Read Only Memories

```
reg [7:0] ram_MEM [15:0];
```

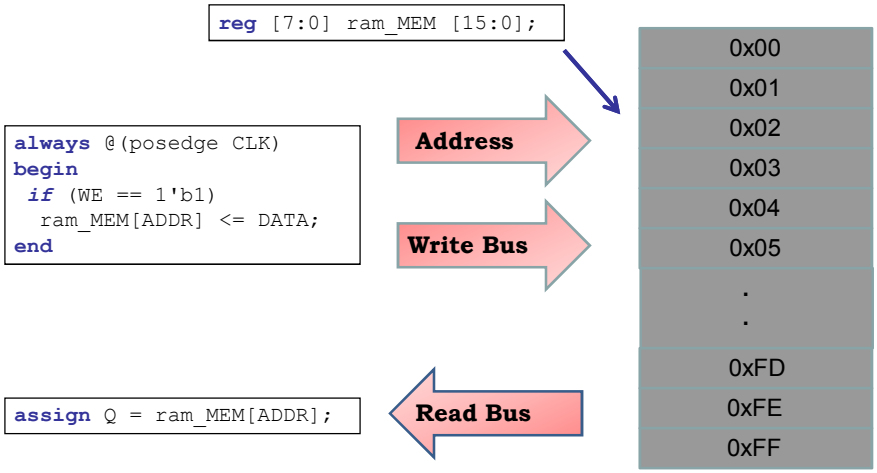



```
always@(address)
begin
 case( address )
  8'h00: read_bus = 'h87;
  8'h01: read_bus = 'h66;
  . . .
 endcase
end
```

A ROM can also be populated by a data file using $readmem() or $readmemb() which is discussed in further topics.

Random Access Memories

```
reg [7:0] ram_MEM [15:0];
```




```
always @(posedge CLK)
begin
 if (WE == 1'b1)
  ram_MEM[ADDR] <= DATA;
end
```

```
assign Q = ram_MEM[ADDR];
```

FIFO

```
always@(posedge clock)
…
else if (WR == 1) FIFO_MEM[WR_PTR] <= DATA;
```

```
if (WREN == 1 && N < 16)
begin
 if (WR_PTR < 15) WR_PTR <= WR_PTR + 1;
 else if (WR_PTR == 15) WR_PTR <= 0;
 N = N + 1;
end
```

If FIFO not full

```
if (RDEN == 1 && N > 0)
begin
 if (RD_PTR < 15) RD_PTR <= RD_PTR+1;
 else if (RD_PTR == 15) RD_PTR <= 0;
 N = N - 1;
end
```

If FIFO not empty

```
if (N == 0) EMPTY = 1;
 else EMPTY = 0;
if (N == 16) FULL = 1;
 else FULL = 0;
```

'N' represents the data depth in FIFO

```
assign Q = (RD == 1) ? FIFO_MEM[RD_PTR] : {8{1'bz}};
```

Two dimensional array

Read

0x00
0x01
0x02
0x03
0x04
0x05
.
.
0xFD
0xFE
0xFF

Front of the data list

Read Pointer

Write Pointer

Last of the data list

Write

Exercise 1

- Design and verify a synchronous FIFO of 8 bit width and 32 bit depth. The FIFO module should indicate the following conditions:
  - FIFO full
  - FIFO empty
  - FIFO almost full
  - FIFO almost empty
  - All of the above should be parametrizable

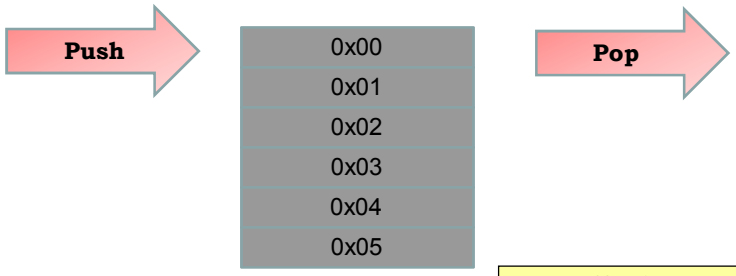
What changes should be made in order that FIFO works in asynchronous mode? Make the changes and verify the same with a test bench.

Learn the differences between asynchronous FIFO and synchronous FIFO.

120 Minutes

Exercise 2

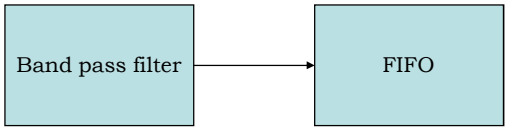
- Design and verify a stack with 8 bit width and 32 bit depth.
  - It should indicate stack almost full and empty conditions



60 Minutes

Exercise 3

The given band pass filter should be connected to a FIFO, write a Verilog top level module which instantiates the FIFO and the band pass filter. The data output of the band pass filter should be stored in the FIFO. Also write a test bench for the module to test for the correctness of the specs

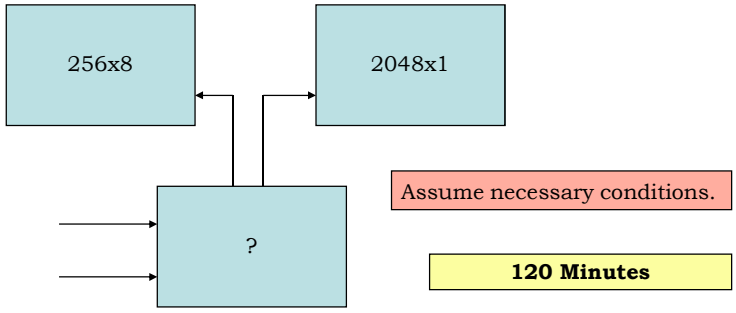


Use the Verilog models which you have designed. If the write rate is 100 MHz and read out rate is 75 Mhz what should be the size of the FIFO so that the data is not lost because of FIFO over flow?

60 Minutes

Exercise 4

- Two memories one 256x8 and 2048x1 are given. The following operations are to be made when appropriate commands are given:
  - Transfer the entire contents of 256x8 memory to 2048x1 memory
  - Transfer the entire contents of 2048x1 memory to 256x8 memory
- Write Verilog models and a test bench to prove the design

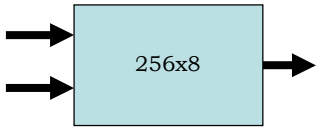


Assume necessary conditions.

120 Minutes

Exercise 5

- Design, implement and verify a dual port memory with 128 depth and 8 bits width.
  - Independent read can be done on both ports i.e. asynchronous read
  - Independent write also can be done on both ports, but, if a write is attempted on both ports at the same time for the same address, port A gets the priority.



Assume necessary signals for the ports

60 Minutes