Arrays en Verilog

Verilog Arrays

- **Vectors**

  reg [7:0] R; // 8-bit register variable

- **Arrays of vectors**

  reg [7:0] M[9:0]; // Array of ten 8-bit registers

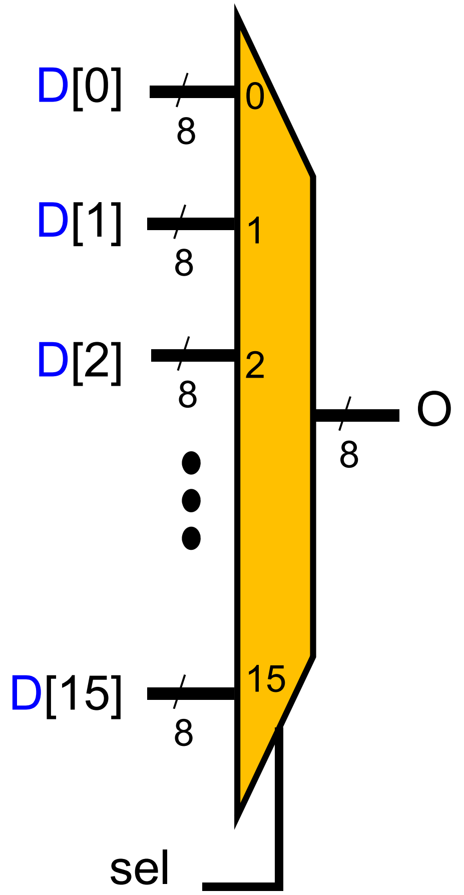
- **Examples:**

  M[0] is the 8-bit vector with index 0

  M[3][7] is the 7th bit of the vector M[3]

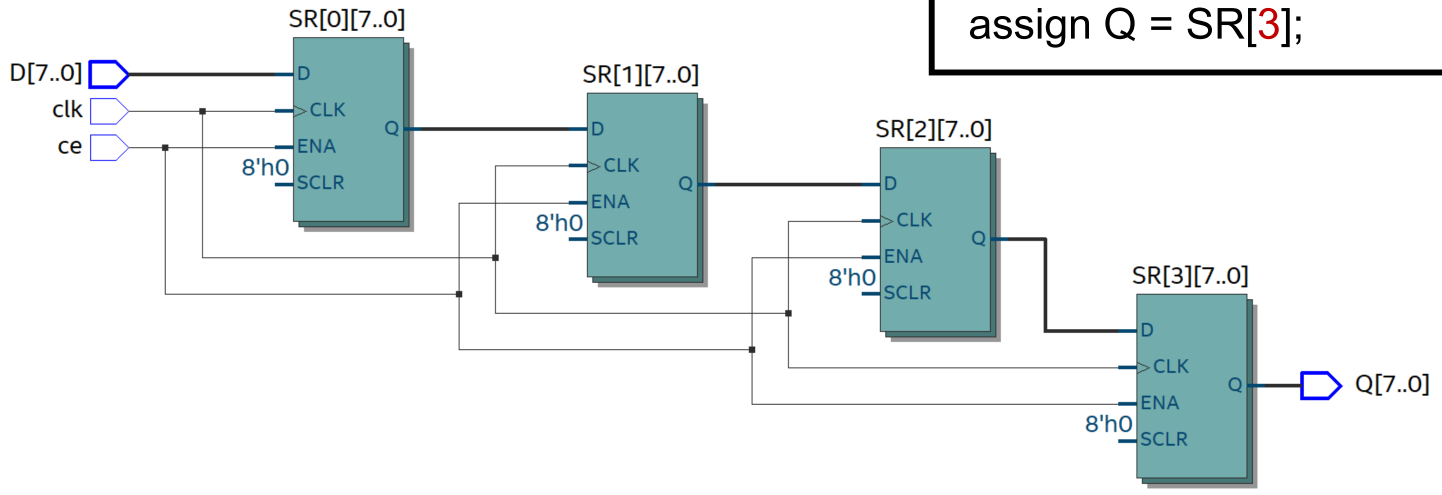
Multiplexer

```
wire [7:0] D[15:0];

wire [3:0] sel;
wire [7:0] O;

assign O = D[sel];
```



Shift register

```
wire clk,ce;
wire [7:0] D;
wire [7:0] Q;

reg [7:0] SR [3:0];

integer i;

always @ (posedge clk)
    if (ce)
        begin
            SR[0] <= D0;
            for (i = 3; i>0; i = i-1)
                SR[i] <= SR[i-1];
       end

assign Q = SR[3];
```

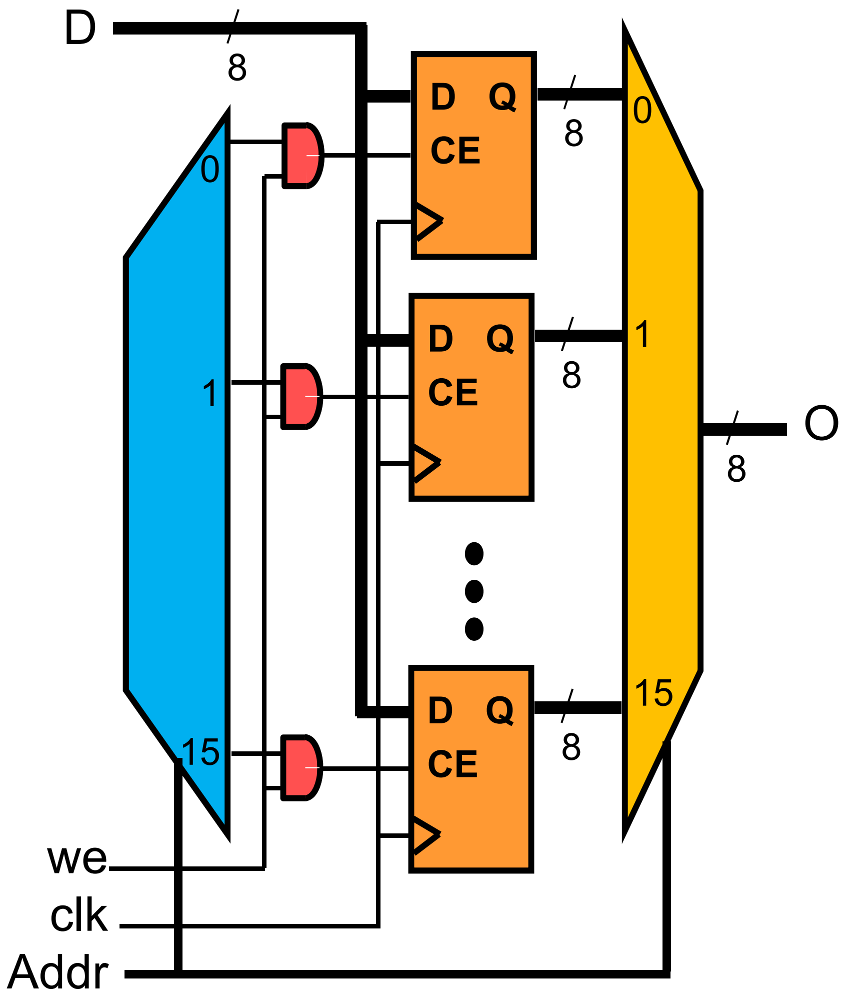


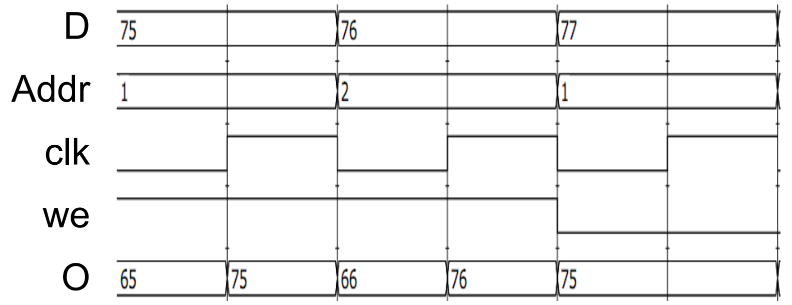
RAM

- **Synchronous write & asynchronous read**

```
wire [7:0] D,
wire [3:0] Addr,
wire clk, we,
wire [7:0] O;

reg [7:0] RAM[15:0];

always@(posedge clk)
    if (we)
        RAM[Addr] <= D;

assign O = RAM[Addr];
```

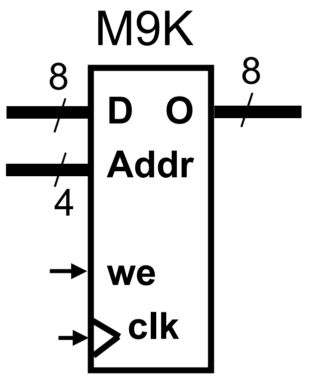
⇒ 16x8 register-based





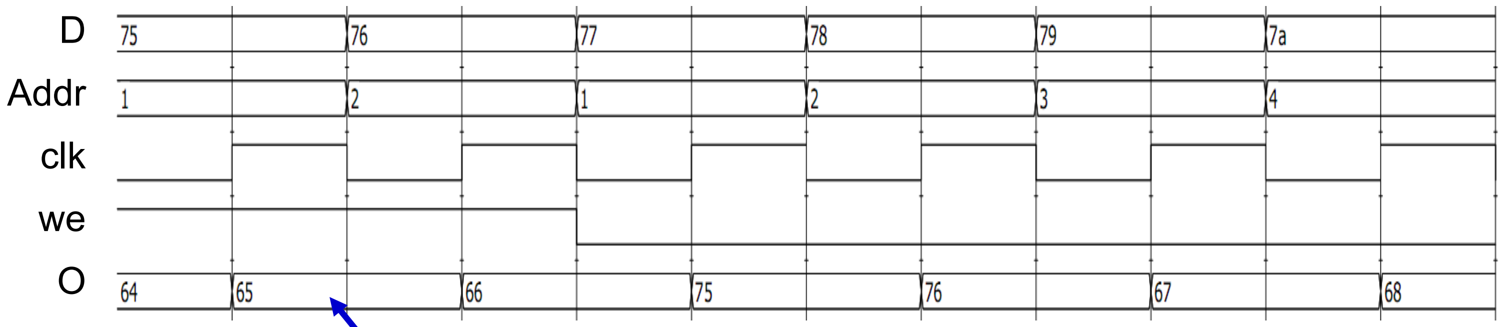
RAMs

- **Synchronous read: read first**



```
wire [7:0] D;
wire [3:0] Addr;
wire clk,we;
reg [7:0] O;

reg [7:0] RAM[15:0];

always@(posedge clk)
    begin
        if (we)
            RAM[Addr] <= D;
        O <= RAM[Addr];
    end
```

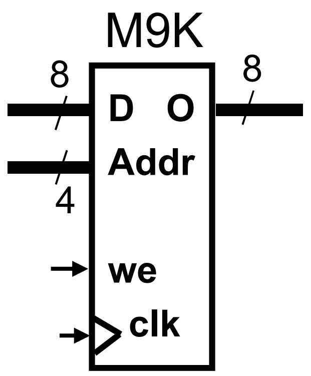
⇒ M9K as a 16x8-bit single port RAM



read the previous value stored in RAM[1]

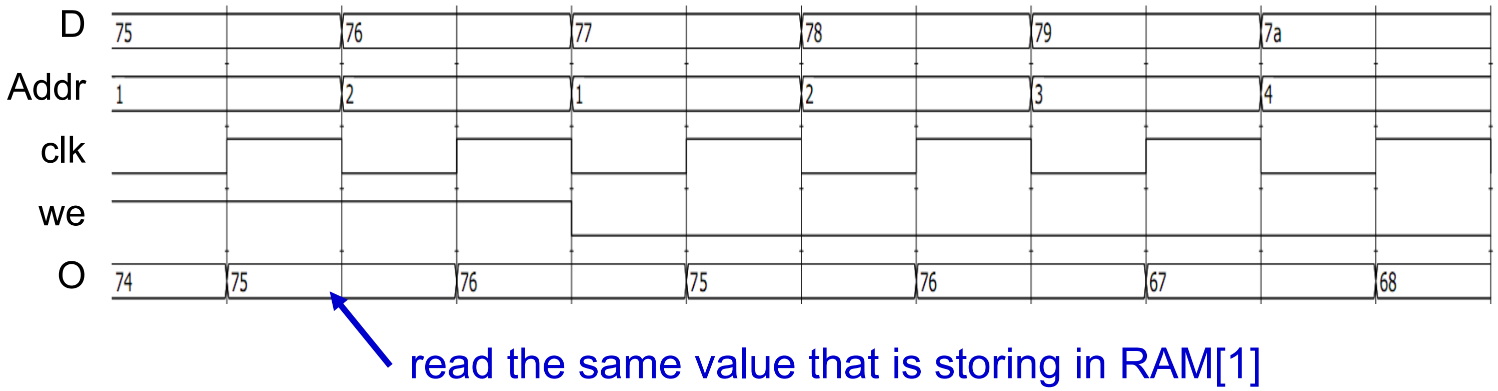
RAMs

- **Synchronous read: write first**



⇒ M9K as a 16x8-bit single port RAM

```
wire [7:0] D;
wire [3:0] Addr;
wire clk,we;
reg [7:0] O;

reg [7:0] RAM[15:0];

always@(posedge clk)
    if (we)
        begin
            RAM[Addr] <= D;
            O <= D;
        end
     else
          O <= RAM[Addr];
```

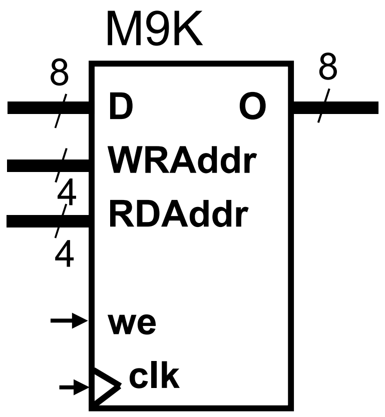


read the same value that is storing in RAM[1]

RAMs

- **Write and read in different addresses**

```
wire [7:0] D;
wire [3:0] RDAddr;
wire [3:0] WRAddr;
wire clk,we;
reg [7:0] O;

reg [7:0] RAM[15:0];

always@(posedge clk)
    begin
      if (we)
          RAM[WRAddr] <= D;
      O <= RAM[RDAddr];
    end
```

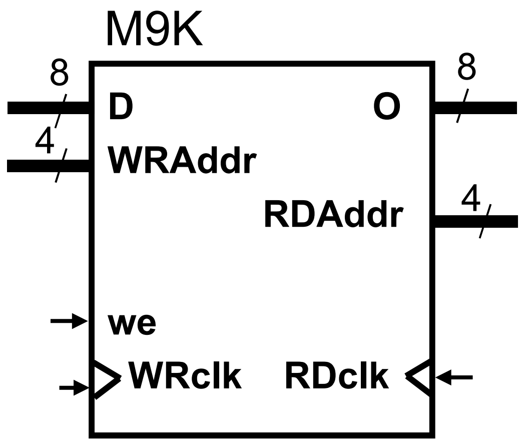


⇒ M9K as a 16x8-bit simple dual port RAM

RAMs

- **Separate read/write addresses and read/write clocks**

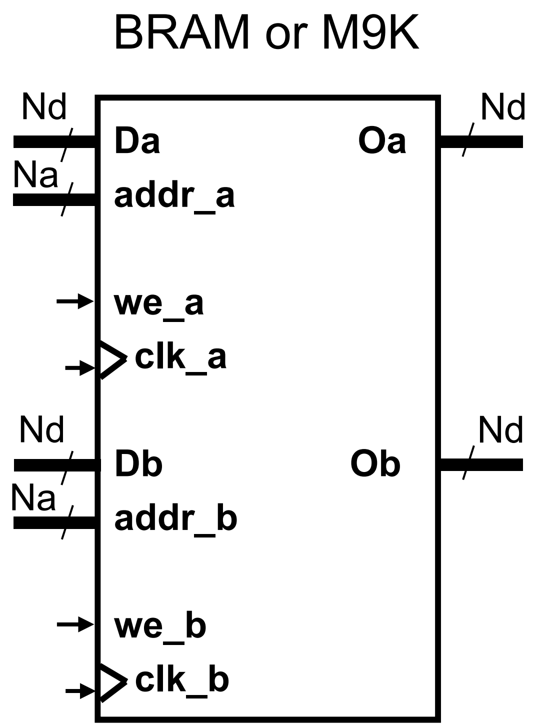
```
wire [7:0] D;
wire [3:0] RDAddr;
wire [3:0] WRAddr;
wire WRclk,RDclk,we;
reg [7:0] O;

reg [7:0] RAM[15:0];

always@(posedge WRclk)
     if (we)
        RAM[WRAddr] <= D;

always@(posedge RDclk)
     O <= RAM[RDAddr];
```



⇒ M9K as a 16x8-bit simple dual port RAM

Dual-port RAM

- Synchronous read: write first



⇒M9K as a 2^NaxNd-bit true dual port RAM

```
parameter Nd=8; //DATA WIDTH
parameter Na=6); //ADDR WIDTH

wire [(Nd-1):0] Da, Db;
wire [(Na-1):0] addr_a, addr_b;
wire we_a, we_b, clk_a, clk_b;
reg [(Nd-1):0] Oa, Ob;

reg [Nd-1:0] ram[2**Na-1:0];

always @ (posedge clk_a)  // Port A
       if (we_a)
          begin
              ram[addr_a] <= Da;
              Oa <= Da;
          end
       else
           Oa <= ram[addr_a];

always @ (posedge clk_b) // Port B
       if (we_b)
           begin
               ram[addr_b] <= Db;
               Ob <= Db;
           end
        else
           Ob <= ram[addr_b];
```

ROM

```
wire [3:0] addr;
wire clk;
reg [7:0] q;

reg [7:0] rom[2**4-1:0];


 initial
    begin
       $readmemh("rom_init.txt", rom);
   end

 always @ (posedge clk)
      q <= rom[addr];
```

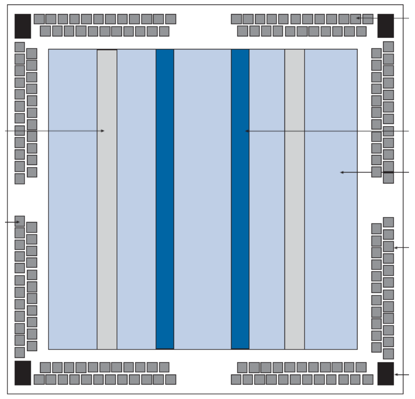
read data from the file "rom_init.txt" and store it in memory "rom"

rom_init.txt

```
0f
f3
2f
3a
d1
55
7f
7f
f2
92
af
bd
c1
d5
6f
ff
```

$readmemh read the file where data are in hex
$readmemb for binary

Memorias M9K en Cyclone IV



Comportamiento:
- Lectura/escritura síncrona

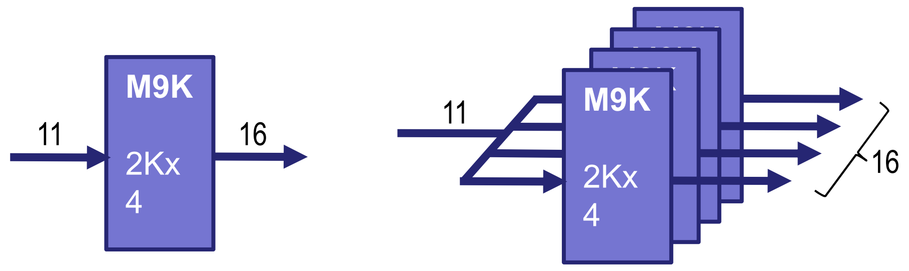
Modos de funcionamiento:
- Puerto simple
- Puerto doble sencillo (lee y escribe en distintas direcciones)
- Puerto doble
- Registro de desplazamiento
- ROM
- FIFO

Configuraciones modo ROM:
- 8Kx1 bits
- 4Kx2 bits
- 2Kx4 bits
- 1Kx(8+1) bits
- 512x(16+2) bits
- 256x(32+4) bits

Configuraciones modo puerto doble:
- 8Kx1 bits
- 4Kx2 bits
- 2Kx4 bits
- 1Kx(8+1) bits
- 512x(16+2) bits

Para implementar una memoria de cierta capacidad el sintetizador combina bloques M9K con la configuración más adecuada

Ej. RAM 2Kx16 bits


11
M9K
2Kx
4
16
11
M9K
2Kx
4
16