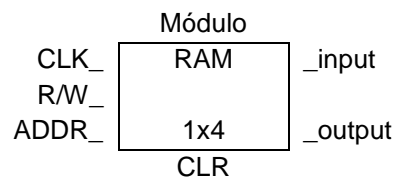


Tema: Introdução à linguagem Verilog
Atividade: Circuitos sequenciais – Memórias
Todos os circuitos deverão ser simulados no Logisim.

- 01.) Projetar e descrever em Verilog e Logisim um módulo para implementar uma memória RAM 1x4 (1 endereço para 4 bits) usando flip-flops do tipo JK como registradores de dados. Ver sugestão abaixo.



DICA:

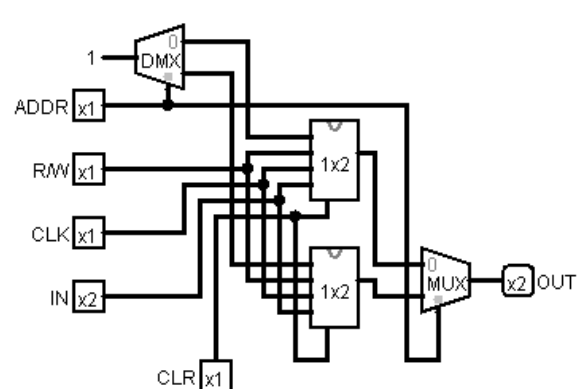
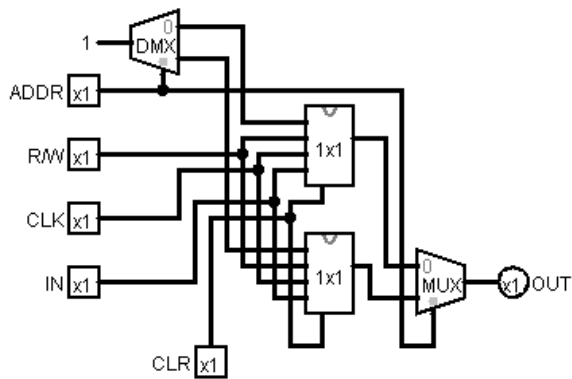
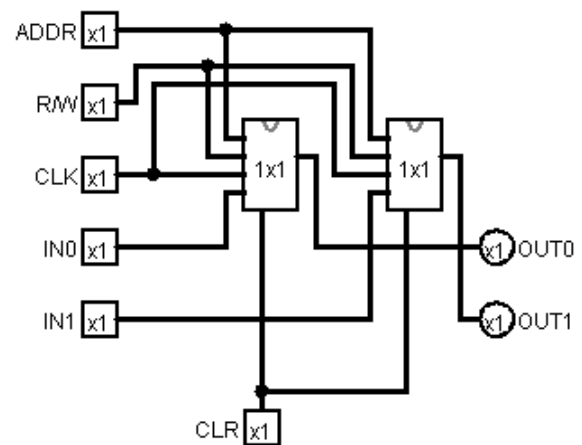
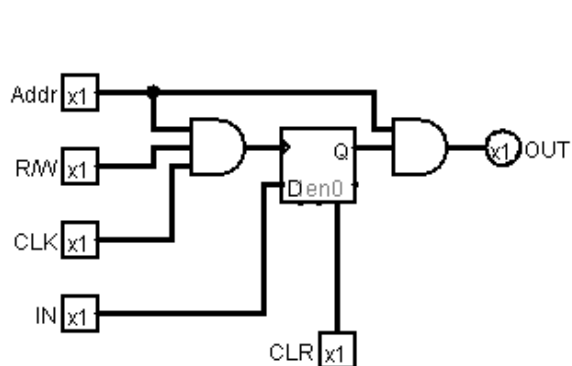
Supor que a escrita ocorrerá na borda de subida do **clock**, enquanto a leitura poderá ocorrer a partir da borda de descida do mesmo.

- 02.) Projetar e descrever em Verilog e Logisim um módulo para implementar uma memória RAM 2x4 (2 endereços para 4 bits cada) usando duas memórias RAM 1x4.
- 03.) Projetar e descrever em Verilog e Logisim um módulo para implementar uma memória RAM 1x8 (1 endereço para 8 bits) usando memórias RAM 1x4.
- 04.) Projetar e descrever em Verilog e Logisim um módulo para implementar uma memória RAM 2x8 (4 endereços para 8 bits) usando memórias RAM 2x4.
- 05.) Projetar e descrever em Verilog e Logisim um módulo para implementar uma memória RAM 8x8 (8 endereços para 8 bits) usando memórias RAM 2x8 do modelo acima (04).

Extras

06.) Projetar e descrever em Verilog e Logisim um módulo para implementar uma memória RAM 1x16 (1 endereço para 16 bits) usando memórias RAM 1x8.

07.) Projetar e descrever em Verilog e Logisim um módulo para implementar uma memória RAM 4x8 (4 endereços para 8 bits) usando memórias RAM 1x8.



```

module dff ( output q, output qnot,
             input  d, input clk );
reg q, qnot;

always @( posedge clk )
begin
    q <= d;      qnot <= ~d;
end

endmodule // dff

module tff ( output q, output qnot,
             input  t, input  clk,
             input  preset, input clear );

reg q, qnot;

always @( posedge clk )
begin
    if ( ~clear )
    begin
        q <= 0;      qnot <= 1;
    end
    else
    if ( ~preset )
    begin
        q <= 1;      qnot <= 0;
    end
    else
    begin
        if ( t )
        begin
            q <= ~q;  qnot <= ~qnot;
        end
    end
end

endmodule // tff

```

```

module srff ( output q, output qnot,
              input  s, input  r, input clk );
reg q, qnot;

always @( posedge clk )
begin
    if ( s & ~r )
    begin
        q <= 1;      qnot <= 0;
    end
    else
    if ( ~s & r )
    begin
        q <= 0;      qnot <= 1;
    end
    else
    if ( s & r )
    begin
        q <= 0;      qnot <= 0; // arbitrary
    end
end

endmodule // srff

module jkff ( output q, output qnot,
              input  j, input  k, input clk );
reg q, qnot;

always @( posedge clk )
begin
    if ( j & ~k )
    begin
        q <= 1;      qnot <= 0;
    end
    else
    if ( ~j & k )
    begin
        q <= 0;      qnot <= 1;
    end
    else
    if ( j & k )
    begin
        q <= ~q;      qnot <= ~qnot;
    end
end

endmodule // jkff

```