Programa para Excelência em Microeletrônica
Matéria: SystemVerilog

Exercício 02

Henrique Martins Miranda

SONY make.believe
FUNDAÇÃO DE APOIO AO IFPB
FUNETEC-PB Fundação de Educação Tecnológica e Cultural da Paraíba
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PARAÍBA

Sumário

Questão 01

```
module Registrador(
        input logic clock,
        input logic [3:0] entrada,
        output logic[3:0] saida
);

        always_ff @ (posedge clock)
                saida <= entrada;
endmodule

module Questao01 (
        input swap, enable, clock,
        output logic[3:0] UpSaida, DownSaida
);

logic[3:0] UpEntrada, DownEntrada;
Registrador Upcount (.clock(clock), .entrada(UpEntrada), .saida(UpSaida)),
                Downcount (.clock(clock), .entrada(DownEntrada), .saida(DownSaida));

always_comb
begin
       if(enable)
         if(swap)
                UpEntrada = DownSaida;
         else
                UpEntrada = UpSaida + 4'd1;
       else
         UpEntrada = UpSaida + 4'd1;
end

always_comb
begin
       if(enable)
         if(swap)
                DownEntrada = UpSaida;
         else
                DownEntrada = DownSaida - 4'd1;
       else
         DownEntrada = DownSaida - 4'd1;
end

endmodule
```

Questão 02

```
module Count(
	input logic clock,
	input logic [8:0] entrada,
	output logic[8:0] saida
);

	always_ff @ (posedge clock)
		saida <= entrada;
endmodule

module Questao02(
	input clock,
	output logic f
);

	logic[8:0] CountEntrada, CountSainda;

	Count count(.clock(clock), .entrada(CountEntrada), .saida(CountSainda));

	always_comb
		begin
			if(CountSainda == 8'd499)
				CountEntrada = 8'd0;
			else
				CountEntrada = CountSainda + 8'd1;
		end

	always_comb
		begin
			if((CountSainda > 8'd19) && (CountSainda < 8'd90))
				f = 1'd0;
			else
				f = 1'd1;
		end

endmodule
```

- “f” em 0 passa aproximadamente 700 ns;
- “f” em 1 passa aproximadamente 4 300 ns.

Questão 03

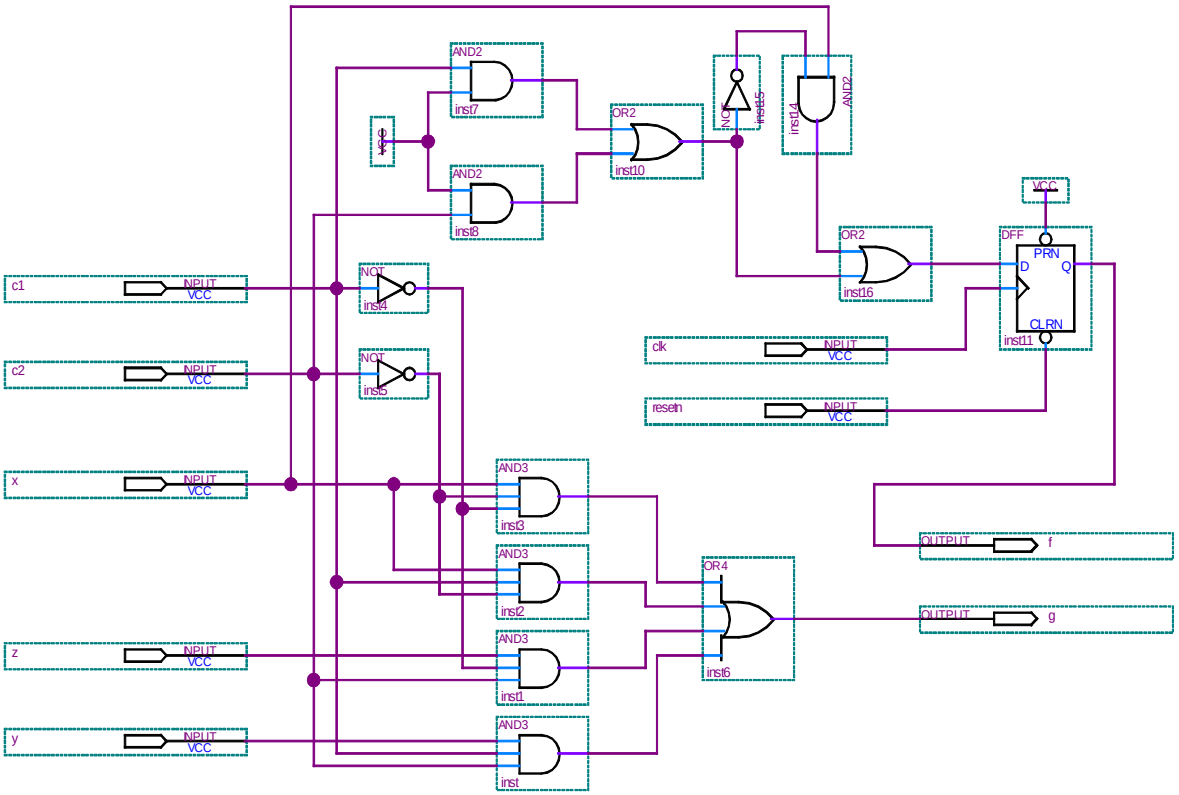

Figura 2 – Questão 03.

Questão 04

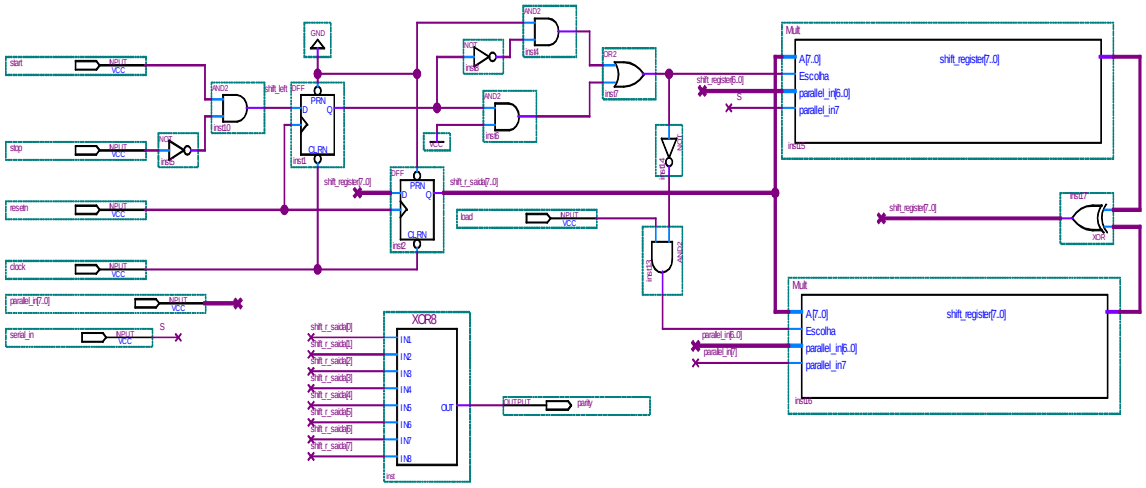

Figura 3 – Questão 04.

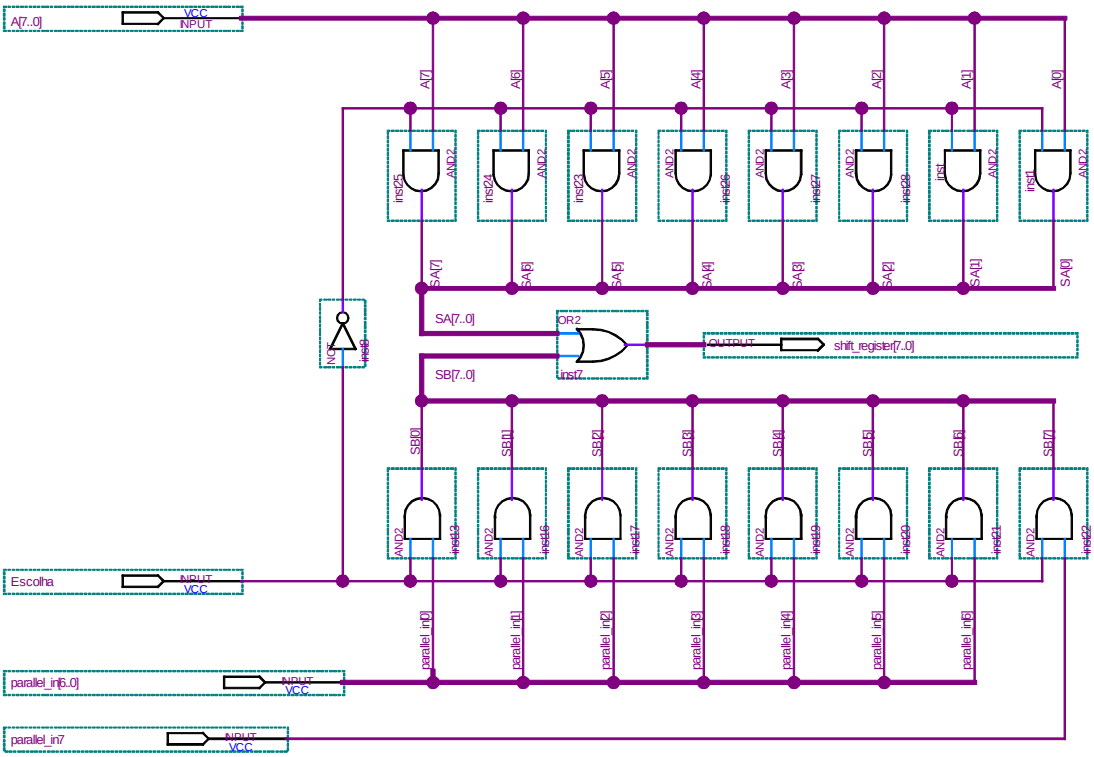

Figura 4 – Multiplex 2-1 de 8 bits.

Questão 05

Sim, pode substituir o laço “for” por um “while”, esses dois laços tem a mesma função, contudo, com sintax diferente, por exemplo a incrementarão da variável “i” tem que ser feita dentro do “while”. O código ficaria assim:

```
[...]
        i =0;

        while (i < 8)
        begin
            if(data_in[i]) msb <= i;

            i = i+1;
        end
[...]
```

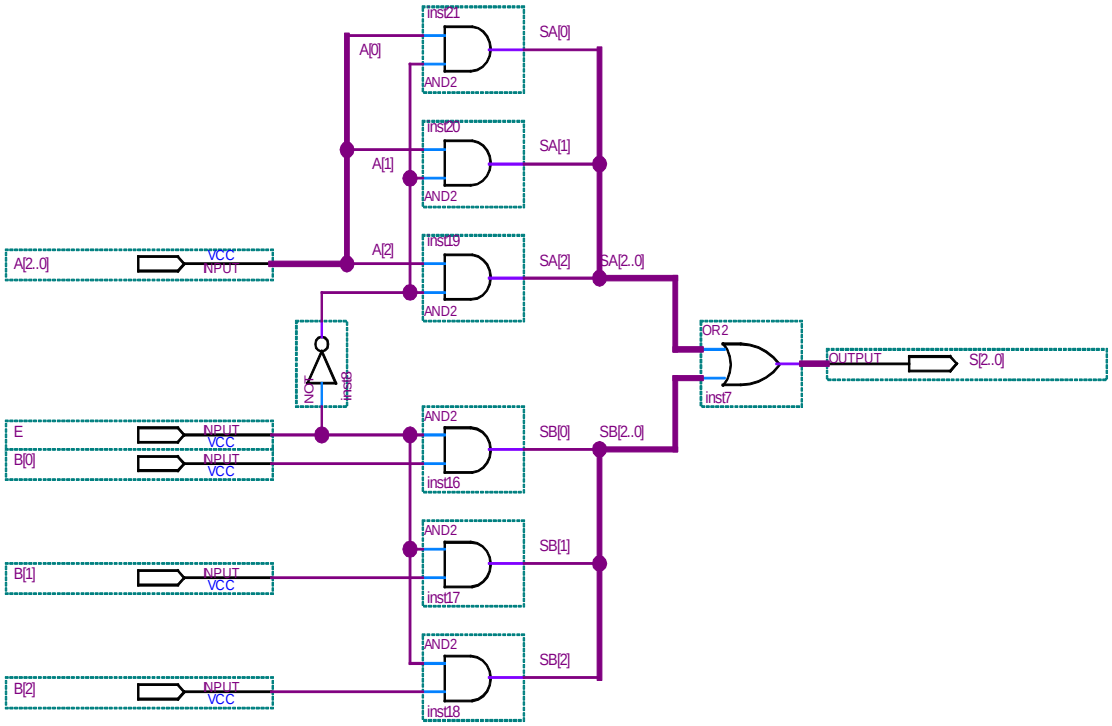

Figura 5 – Multiplex 2-1 3bits.

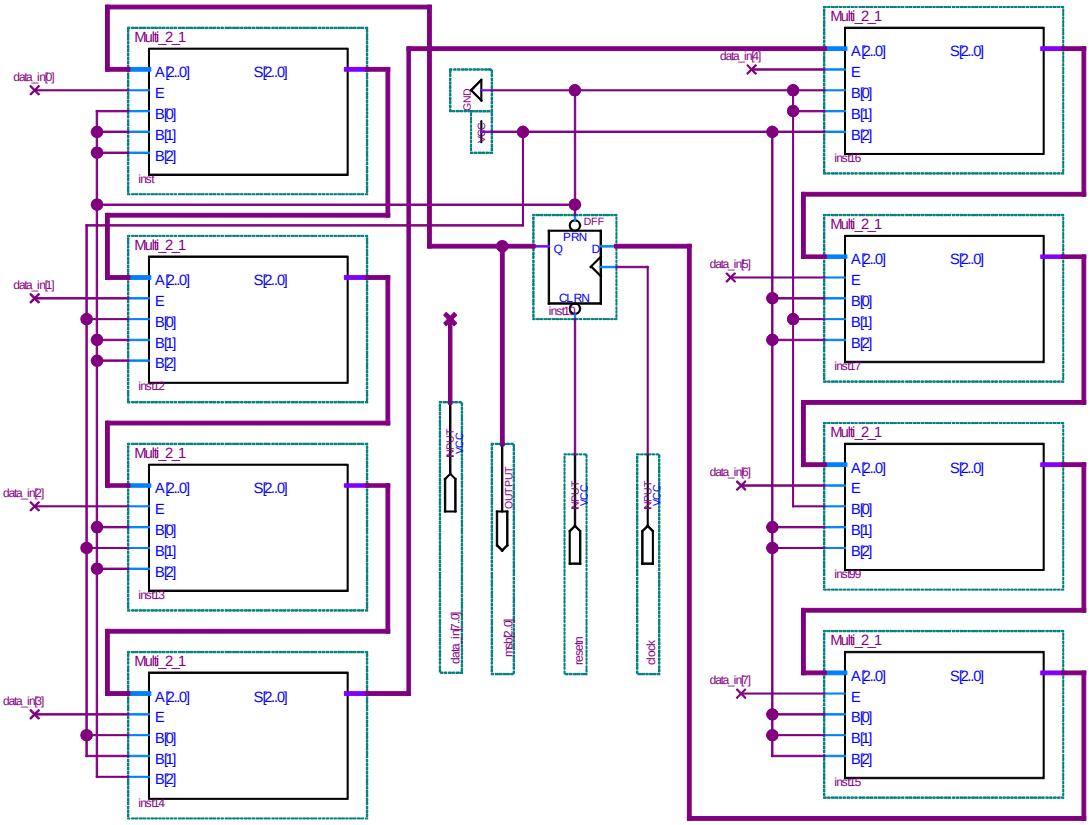

Figura 6 – Questão 05.

Questão 06

- “f” em 0 passa 810 ns;
- “f” em 1 passa 3 690 ns;
- “g’ em 1 passa 1 012,5 ns;
- “g” em 0 passa 3 487,5 ns.

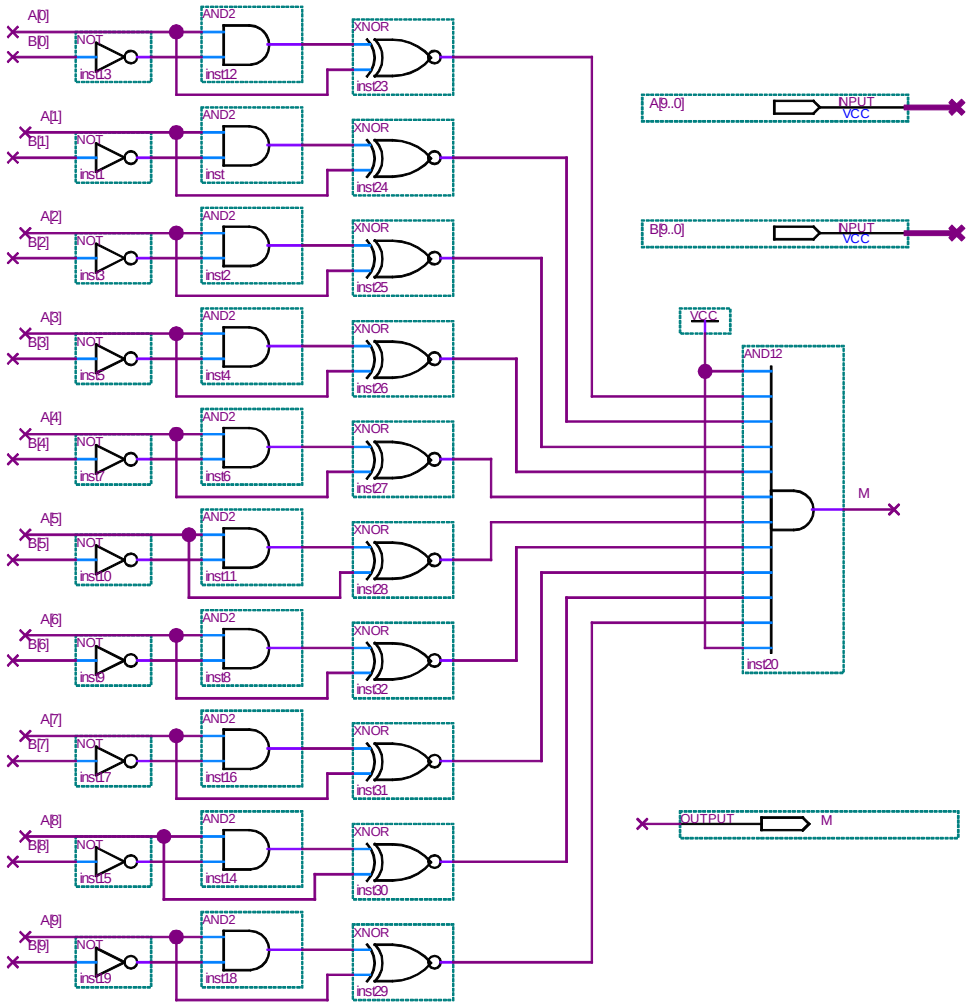

Figura 7 – Comparador “>” de 10 bits.

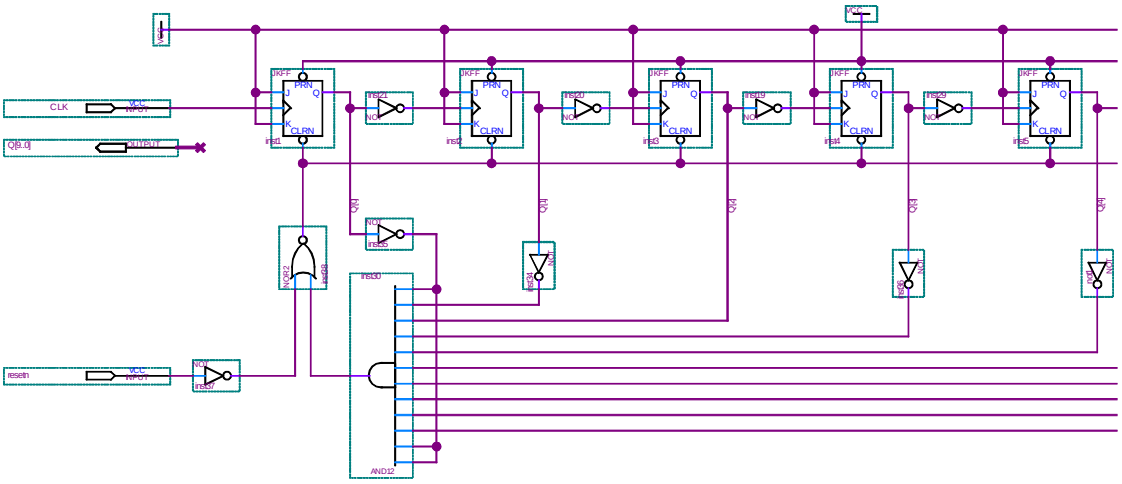

Figura 8 – Contador de 10 bits parte 1.

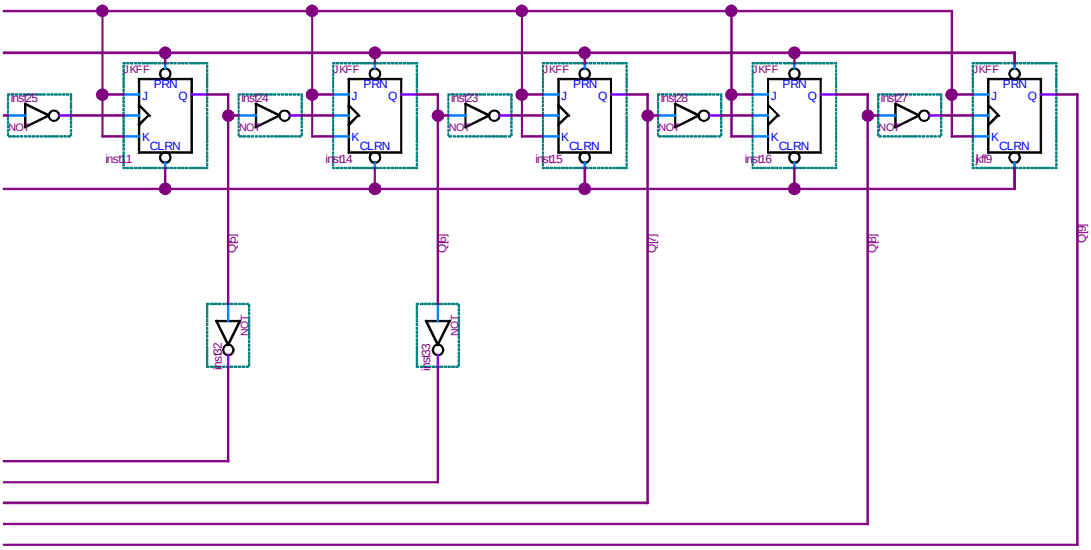

Figura 9 – Contador de 10 bits parte 2.

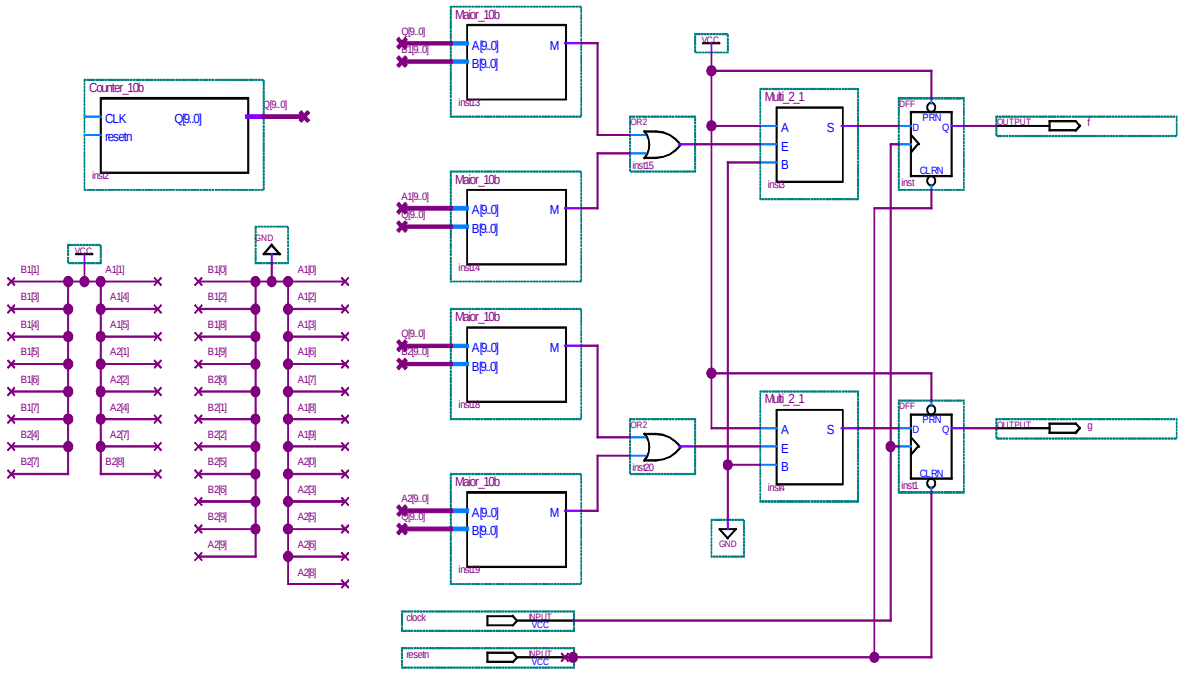

Figura 10 – Questão 06.

Questão 07

A saída “f” receber o valor de “c” antes de “c” receber “a + b”, pós é bloqueante, contudo, “a” e “b” recebe os valores não bloqueantes para que quando o “c” for receber os valores deles, já receber o “valor atualizado” e assim poder ir contando. O “f” é sempre o valor antigo de “c”.

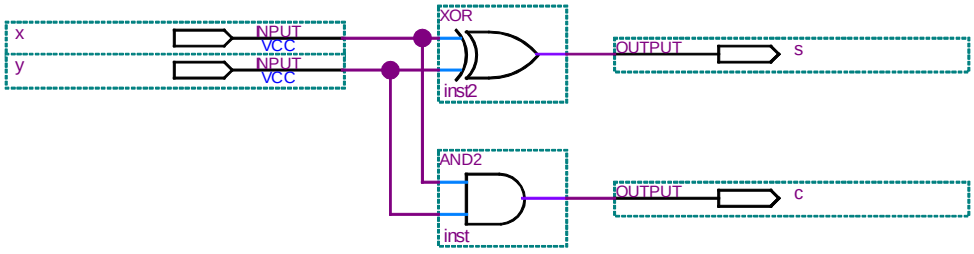

Figura 11 – Meio Somador usado nas questões 7,8 e 9.

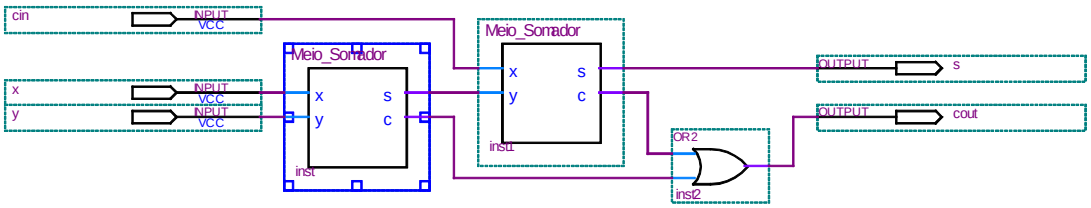

Figura 12 – Somador Completo usado nas questões 7,8 e 9.

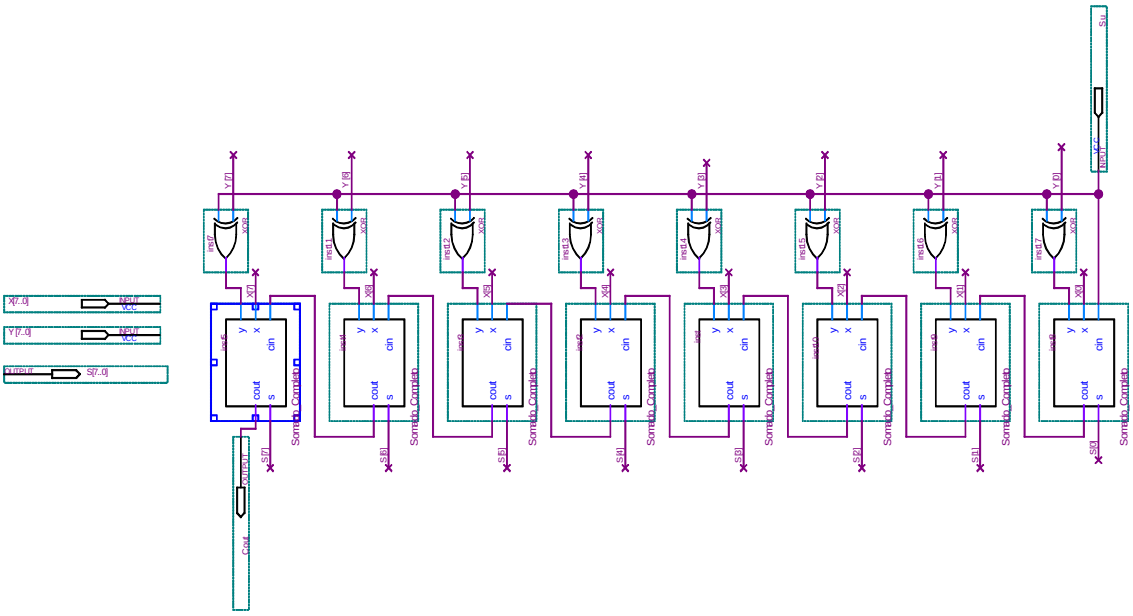

Figura 13 – Somador 8bits usado nas questões 7,8 e 9.

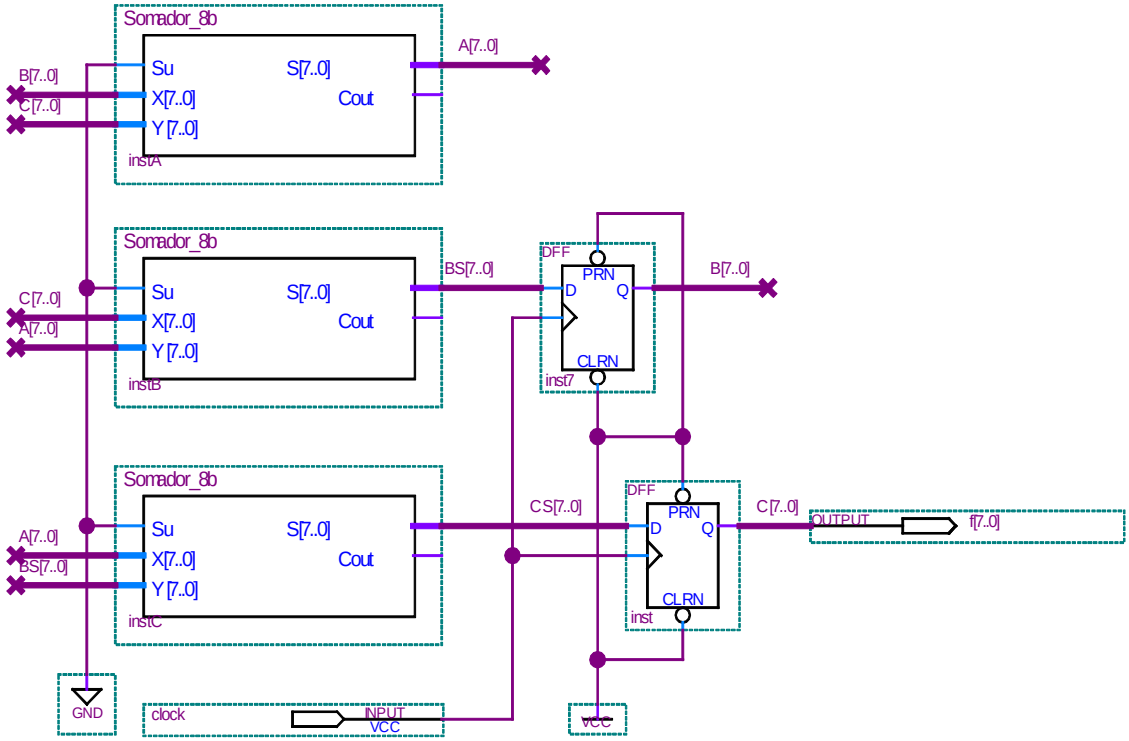

Figura 14 – Questão 07.

Questão 08

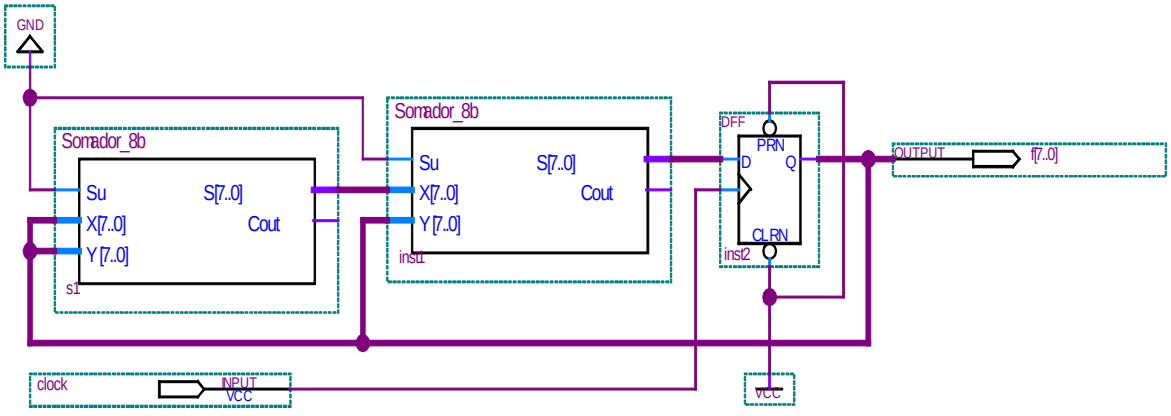

Figura 15 – Questão 08.

Questão 09

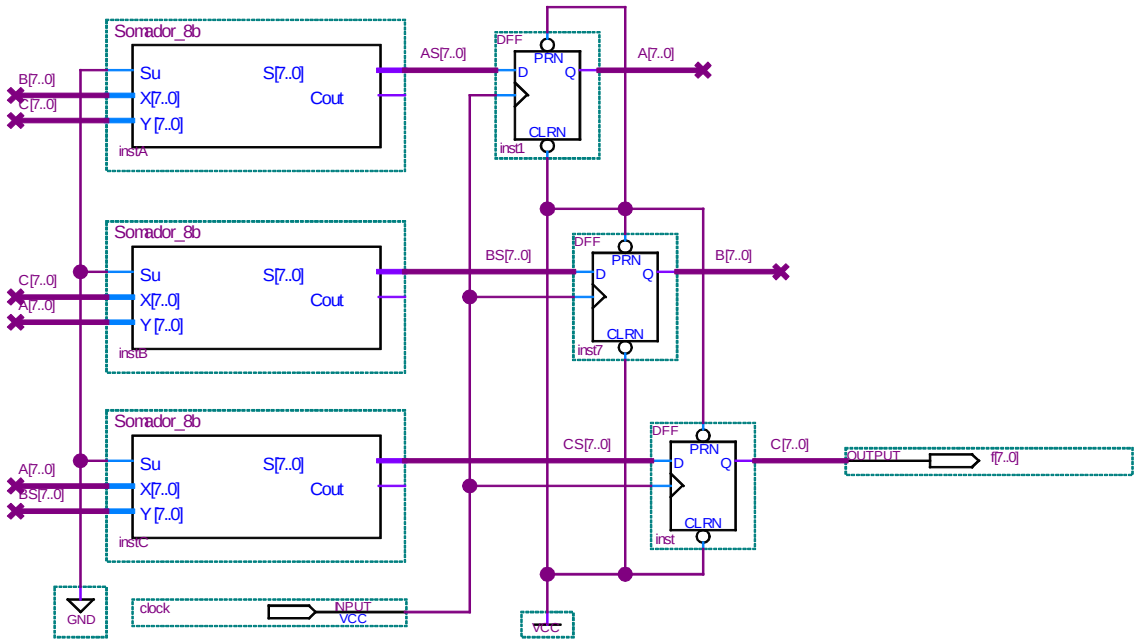

Figura 16 – Questão 08.

Questão 10

```
module Questao10 (
  input logic reset, clk,
  output logic f, g
);
logic [12:0] counter;
always_ff @(posedge clk or negedge reset )
begin
  if (!reset)
  begin
          counter <= 10'd0;
          f <= 1'b0;
          g <= 1'b0;
  end
  else
  begin
          f <= 1'b1;
          if (counter > 13'd3850 && counter < 13'd4150) f <= 1'b0;
          g <= 1'b1;
          if (counter > 13'd3200 && counter < 13'd3800) g <= 1'b0;
          if (counter < 13'd4600) counter <= counter + 13'd1;
          else counter <= 13'd0;

  end
end
endmodule
```