Revisar envio do teste: Quiz Primeiro Bimestre

Usuário	RICARDO ISRAEL
Curso	DESIGN DE COMPUTADORES - 6ENGCOMPA 2023/1
Teste	Quiz Primeiro Bimestre
Iniciado	04/04/23 18:08
Enviado	04/04/23 18:16
Data de vencimento	
Status	Completada
Resultado da tentativ	a 8 em 10 pontos
Tempo decorrido	8 minutos de 2 horas
	Todas as respostas, Respostas enviadas, Respostas corretas, Comentários, Perguntas respondidas incorretamente

Pergunta 1 1,75 em 1,75 pontos

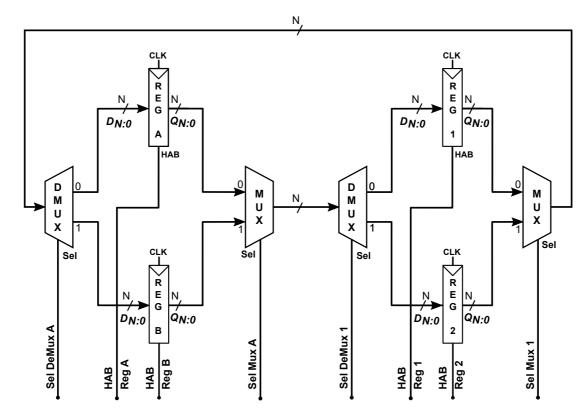
(Z)

Para o fluxo de dados abaixo, considere que:

- O sinal de "clock" está distribuído adequadamente;
- Os pontos de controle dos registradores (tipo D) fazem a habilitação em nível ALTO;
- Os registradores, quando habilitados, alteram seu valor na próxima borda ascendente do "clock";
- Não existem problemas de temporização;
- A seleção dos multiplexadores segue os níveis que estão indicados no desenho.

Crie uma sequência de palavras de controle, uma para cada "clock" consecutivo, que faça a troca (swap) entre os conteúdos dos registradores A e B.

Sem modificar o projeto, existe alguma forma de diminuir a quantidade de ciclos utilizados?



Copie e cole a tabela abaixo para fazer a sua resposta:

	Ponto de Controle	clk 1	clk 2	clk 3	clk 4	clk 5	clk 6	clk 7	clk 8
	Sel DeMux A								
	HabReg A								
	HabReg B								



Sel Mux A				
Sel DeMux 1				
HabReg 1				
HabReg 2				
Sel Mux 1				

Resposta Selecionada: Ponto de Controle clk 1 clk 2 clk 3 clk 4

Sel DeMux A Χ 0 HabReg A 0 1 0 X X 0 0 0 HabReg B 0 1 1 Sel Mux A Χ Sel DeMux 1 Χ HabReg 1 0 0 HabReg 2 0 0 1 Sel Mux 1 Χ 0

Existe sim uma forma de diminuir a quantidade de ciclos:

Ponto de Controle	clk 1	clk 2	clk 3		
Sel DeMux A		Χ		1	0
HabReg A		0		0	1
HabReg B		0		1	0
Sel Mux A		0		1	Χ
Sel DeMux 1		0		1	Χ
HabReg 1		1		0	0
HabReg 2		0		1	0
Sel Mux 1		Χ		0	1

Resposta Correta:



Versão básica:

Ponto de Controle clk 1 clk 2 clk 3 clk 4

X X Sel DeMux A 0 1 HabReg A 0 0 0 1 HabReg B 0 0 0 1 0 1 Sel Mux A Χ Χ Sel DeMux 1 HabReg 1 HabReg 2 Sel Mux 1 0 1 Χ Χ 0 1 0 0 0 0 1 0 0 1

Versão otimizada:

É possível ler e escrever no mesmo registrador durante o mesmo pulso de *clock*. Portanto, podemos economizar um *clock*.

Versão Otimizada da Ativação

dos Pontos

 Ponto de Controle
 clk 1
 clk 2
 clk 3

 Sel DeMux A
 X
 1
 0

 HabReg A
 0
 0
 1

 Sel Mux A
 0
 1
 X

 Sel DeMux A
 0
 1
 X

 Sel DeMux 1
 0
 1
 X

 HabReg 1
 1
 0
 0

 HabReg 2
 0
 1
 0

Comentário da

Rubrica:

resposta:

Versão simples: 0,75 pts Versão otimizada: 1 pts

Sel Mux 1 X 0 1

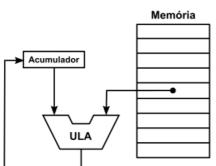
Pergunta 2 0,75 em 0,75 pontos



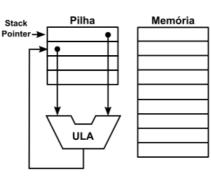
Para as arquiteturas abaixo, quais são os tipos de instruções executáveis?



Arquitetura: Acumulador

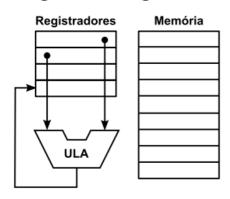


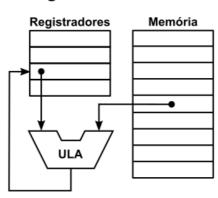
Arquitetura: Pilha



Arquitetura: Registrador-Registrador

Arquitetura: Registrador-Memória





Pergunta Correspondência Correta Correspondência Selecionada Add R1, 0x1FF Arquitetura: Registrador-Memória Arquitetura: Registrador-Memória **⊘** B. 👩 A. Arquitetura: Acumulador 👩 A. Arquitetura: Acumulador Add [0x1FF] Add R1, R2, R3 👩 C. Arquitetura : Registrador-Registrador 👩 C. Arquitetura : Registrador-Registrador Arquitetura: Pilha push a; push a; push b; push b; add; add; O. pop c; O. pop c;

Escolhas com todas as respostas

A. Arquitetura: Acumulador

Arquitetura: Registrador-Memória

В.

C. Arquitetura : Registrador-Registrador

push a; push b; add; D. pop c;

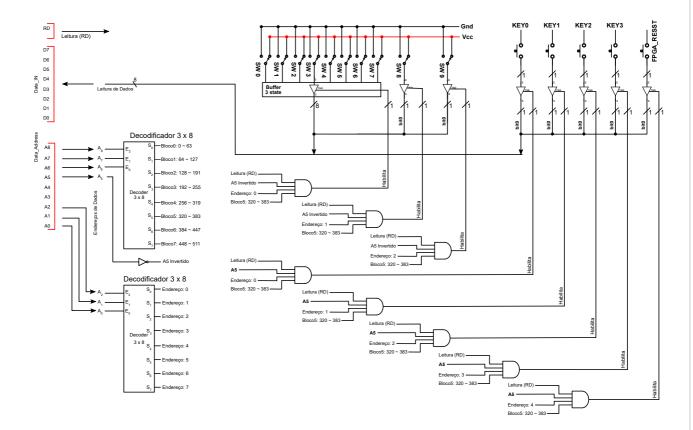
Pergunta 3 0 em 1,5 pontos



Considerando as informações abaixo:

- RD=1;
- Data_Address=0x158;
- Data_IN=0x09.

Qual das afirmações está correta?



Resposta Selecionada: 🔼 A. Estamos fazendo a leitura das chaves SW (9 downto0), com SW9 =1.

Respostas:

- A. Estamos fazendo a leitura das chaves SW (9 downto0), com SW9 =1.
- B. Estamos fazendo a leitura do Botão Key 0 com o valor de 1 e os demais valores podem ser ignorados.

C.

O endereçamento 0x09 insere de A8 até A0 os valores de 0b00001001, sendo que os três últimos bits habilitam a leitura do botão KEY1.

- 🗸 D. Estamos fazendo a leitura das chaves SW (7 downto 0) com o valor = 0b00001001.
 - E. O endereçamento não hablita nem chaves e nem botões deixando suas saídas em alta impedância (tristate).
 - F. Estamos fazendo a leitura das chaves SW (7 downto 0) com o valor = 0b10010000.
 - G. Nenhuma das outras afirmações está correta.

Comentário da resposta: Estamos fazendo a leitura das chaves SW (7 downto 0) com o valor = 0b00001001.

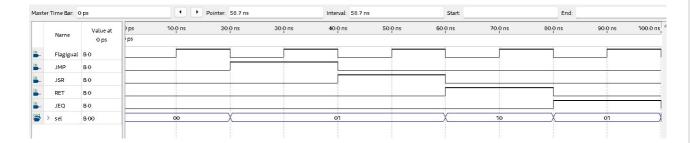
Pergunta 4 1,25 em 1,25 pontos

Ao desenvolver a lógica de desvio do diagrama abaixo, obtivemos a carta de tempo que não corresponde ao funcionamento correto.

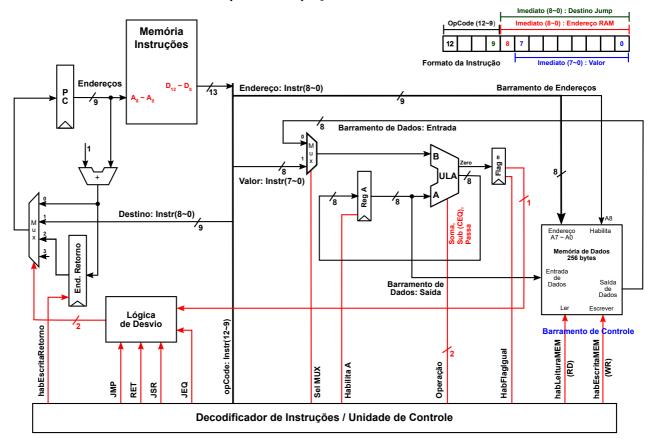
Considere que os sinais de entrada não correspondem ao funcionamento do processador e são somente os vetores de teste para o bloco lógico de desvio

Ou seja, considere como se fossem os valores de entrada de uma tabela da verdade e a saída o sinal de controle desejado (sel).

Onde está o erro na carta de tempo e qual deveria ser a expressão correta?



Computador - Equação de Primeiro Grau



Resposta Selecionada:

Respostas:

O erro está no intervalo de tempo entre 80ns e 90ns, pois com a flag em zero, a seleção deveria ser 00.

E as expressões corretas são:

sel(1) <= not(JMP) and RET and not(JSR) and not (JEQ);

o G

sel(0) <= JMP or (JEQ and FlagIgual) or JSR;

O erro está no intervalo de tempo entre 80ns e 90ns, pois com a flag em zero, a seleção deveria ser 10. E as expressões corretas são:

sel(1) <= not(FlagIgual) and JEQ;

sel(0) <= JMP or (JEQ and FlagIgual) or JSR;

A.

Não há erro na carta de tempo e ela corresponde ao funcionamento correto.

B. A afirmação de que há um erro na carta é devido a uma má interpretação dos resultados feita pelo projetista.

O erro está no intervalo de tempo entre 40ns e 60ns, pois com o JSR em 1, a seleção deveria ser 10.

E as expressões corretas são:

sel(1) <= JSR and not(JMP);

c. $sel(0) \le JMP \text{ or (JEQ and FlagIgual) or JSR;}$

O erro está no intervalo de tempo entre 60ns e 80ns, pois com o RET em 1, a seleção deveria ser 01.

E as expressões corretas são:

sel(1) <= JSR and not (RET);

D. $sel(0) \le RET or not(JSR);$

O erro está no intervalo de tempo entre 40ns e 60ns, pois com o JSR em 1, a seleção deveria ser 00.

E as expressões corretas são:

 $sel(1) \le not(JMP)$ and RET and not(JSR) and not(JEQ);

E. $sel(0) \le (JMP \text{ or } JEQ) \text{ and } JSR;$

O erro está no intervalo de tempo entre 60ns e 80ns, pois com o RET em 1, a seleção deveria ser 00.

E as expressões corretas são:

sel(1) <= ISR and not (RET):

F. sel(0) <= JMP or (JEQ and FlagIgual) or JSR;

O erro está no intervalo de tempo entre 80ns e 90ns, pois com a flag em zero, a seleção deveria ser 00.

E as expressões corretas são:

sel(1) <= not(JMP) and RET and not(JSR) and not (JEQ);

✓ G. sel(0) <= JMP or (JEQ and FlagIgual) or JSR;
</p>

Comentário da resposta: O erro está no intervalo de tempo entre 80ns e 90ns, pois com a flag em zero, a seleção deveria ser 00.

E as expressões corretas são:

sel(1) <= not(JMP) and RET and not(JSR) and not (JEQ);

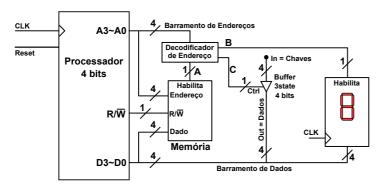
sel(0) <= JMP or (JEQ and FlagIgual) or JSR;



Considerando um processador de 4 bits, tanto de endereços quanto de dados, queremos que:

- A sua memória esteja alocada entre a posição 0 (0b0000) e posição 13 (0b1101) do mapa de endereçamento. O sinal de ativação será chamado de A;
- Um periférico, de escrita (display 7 segmentos), formado por um registrador de 4 bits, esteja na posição 14 (0b1110). O sinal de ativação será chamado de B;
- Um periférico, de leitura, formado por um buffer tristate de 4 bits, esteja na posição 15 (0b1111). O sinal de ativação será chamado de C.

Qual projeto lógico que corresponde a esse decodificador?



Resposta Selecionada: A = not(A3 AND A2 AND A1);

B = (A3 AND A2 AND A1 AND (NOT A0));

⊗ B. C = (A3 AND A2 AND A1 AND A0).

Respostas:

A = (A3 AND A2 AND A1);

B = (A3 AND A2 AND A1 AND A0);

A. C = (A3 AND A2 AND A1 AND A0).

A = not(A3 AND A2 AND A1);

B = (A3 AND A2 AND A1 AND (NOT A0));

✓ B. C = (A3 AND A2 AND A1 AND A0).

A = (A3 AND A2);

B = (A3 AND A2 AND A1);

C. C = (A3 AND A2 AND A1 AND (NOT(A0)).

Não é possível implementar esse decodificador porque a memória ocuparia todo o espaço de endereçamento. D lsso é devido ao fato de a memória ser um dispositivo de bloco e ter 2^n posições.

A = not(A3 AND A2 AND A1);

B = (A3 AND A2 AND A1);

E. C = (A3 AND A2 AND A1 AND (NOT A0)).

A = (A3 AND A2 AND A1 AND A0);

B = (A3 AND A2 AND (NOT A1) AND A0);

 $_{\mathsf{F.}}$ C = (A3 AND A2 AND A1 AND A0).

Comentário da resposta: A = not(A3 AND A2 AND A1);

B = (A3 AND A2 AND A1 AND (NOT A0));

C = (A3 AND A2 AND A1 AND A0).

Pergunta 6 1,5 em 1,5 pontos



A imagem e o código em VHDL abaixo são correspondentes, porém ao tentar verificar o funcionamento, percebeu-se que existe um problema.

Considere que todos os componentes foram criados corretamentre e utilizam o generics.

Qual das alternativas abaixo corresponde ao problema deste código?

```
use ieee.std_logic_1164.all;
entity Aula1 is
generic ( larguraDados : natural := 4;
```

simulacao : boolean := TRUE

);

port (

library ieee;

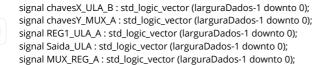
CLOCK_50 : in std_logic;

KEY: in std_logic_vector(3 downto 0);
SW: in std_logic_vector(9 downto 0);

LEDR: out std_logic_vector(9 downto 0)

end entity;

architecture arquitetura of Aula1 is

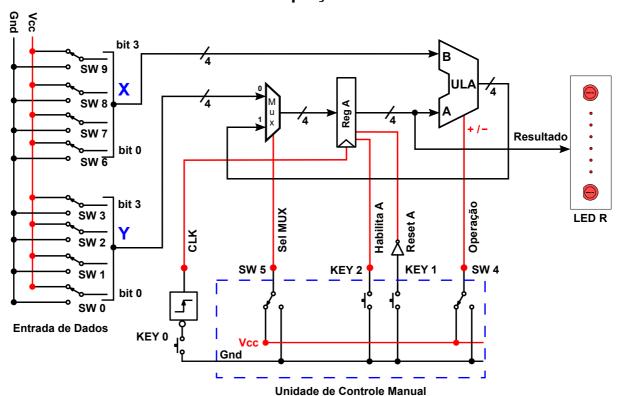




```
signal CLK: std_logic;
 signal SelMUX : std_logic;
 signal Habilita_A: std_logic;
 signal Reset_A: std_logic;
 signal Operacao_ULA: std_logic;
begin
gravar: if simulacao generate
CLK <= KEY(0);
else generate
detectorSub0: work.edgeDetector(bordaSubida)
    port map (clk => CLOCK_50, entrada => (not KEY(0)), saida => CLK);
end generate;
MUX1 : entity work.muxGenerico2x1 generic map (larguraDados => larguraDados)
    port map( entradaA_MUX => chavesY_MUX_A,
         entradaB_MUX => Saida_ULA,
         seletor_MUX => SelMUX,
         saida_MUX => MUX_REG_A);
REG1 : entity work.registradorGenerico generic map (larguraDados => larguraDados)
     port map (DIN => REG1_ULA_A, DOUT => MUX_REG_A, ENABLE => Habilita_A, CLK => CLK, RST => Reset_A);
ULA1 : entity work.ULASomaSub generic map(larguraDados => larguraDados)
     port map (entradaA => REG1_ULA_A, entradaB => chavesX_ULA_B, saida => Saida_ULA, seletor => SW(4));
chavesX_ULA_B <= SW(9 downto 6);
chavesY_MUX_A <= SW(3 downto 0);
SelMUX <= SW(5);
Operacao_ULA <= SW(4);
Reset_A <= not KEY(1);
Habilita_A <= KEY(2);
LEDR (9) <= SelMUX;
LEDR (8) <= Habilita_A;
LEDR (7) <= Reset_A;
LEDR (6) <= Operacao_ULA;
LEDR (5) <= '0'; -- Apagado.
LEDR (4) <= '0'; -- Apagado.
LEDR (3 downto 0) <= REG1_ULA_A;
end architecture;
```

signal Chave_Operacao_ULA: std_logic;

Fluxo de Dados - Equação de Primeiro Grau





Resposta Selecionada: 👩 F. A entrada e saída do registrador estão invertidas.

Respostas: A. A saída da ULA deveria estar conectada ao LEDR(3 downto 0).

B. As entradas SW (9 downto 6) deveriam estar conectadas na entrada B da ULA.

C. As entradas SW (3 downto 0) deveriam estar conectadas na entrada A do MUX.

D. As conexões dos SW estão invertidos.

E. O problema de funcionamento não é referente ao VHDL e pode ter sido um erro de interpretação da simulação.

🗸 F. A entrada e saída do registrador estão invertidas.

G. Não foram declarados como signal: SW e LEDR.

Comentário da resposta: A entrada e saída do registrador estão invertidas.

Pergunta 7 0 em 0 pontos

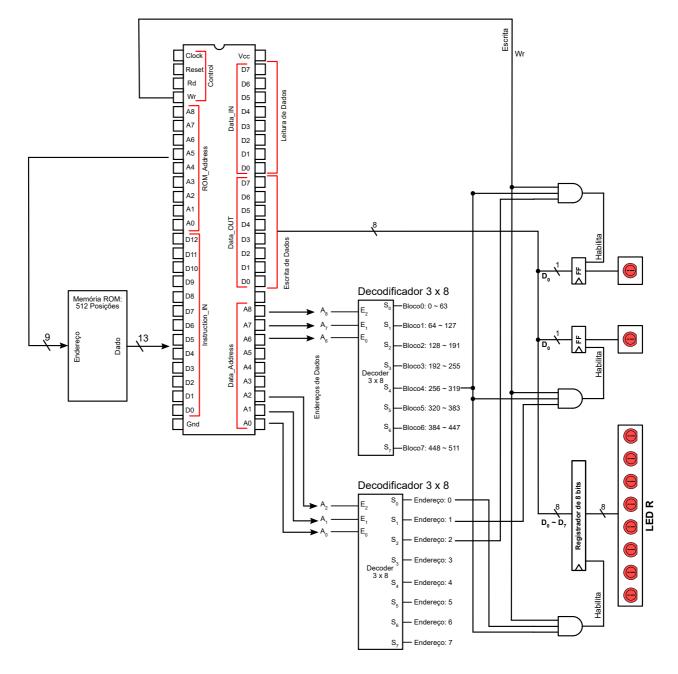


Analisando o programa de teste aplicado ao diagrama abaixo, qual resposta está correta?

Considere que a memória RAM está localizada no bloco 0.

Linha	Instrução
0	JMP @3
1	JEQ @9
2	NOP
3	LDI \$170
4	STA @25
5	STA @256
6	CEQ @25
7	JMP @1
8	NOP
9	LDI \$85
10	CEQ @25
11	JEQ @3
12	IMP @12





Respostas Selecionadas: 🔼 G. Na linha de instrução 9, fará com que o conjunto LED R tenha o valor de sua entrada igual a 0b01010101.

Respostas:

A.

O programa nunca executará a linha de instrução 12, pois existe na linha de instrução 11 um jump condicional para linha de instrução 3.

В.

A instrução STA @256 habilitará os dois Flip-Flop's (FF), pois no endereçamento do decoder mais acima, a saída S4 estará com nível lógico alto.

C

O Program Counter executará as linhas de instrução sequencialmente, pois existe um incrementador de 1 e isso faz com que após a execução da linha de instrução 12 retorne para alinha de instrução 0.

- 👩 D. O conjunto LED R, terá o valor de sua entrada igual a 0b10101010
 - E. A linha de instrução 10 ativará o flag de igual, pois o valor na linha de instrução 5 é o mesmo.
 - F. Nenhuma das outras afirmações está correta.
 - G. Na linha de instrução 9, fará com que o conjunto LED R tenha o valor de sua entrada igual a 0b01010101.

Comentário da resposta:	Linha	Instrução	Observação
	0	JMP @3	Deve desviar para a posição 3
	1	JEQ @9	Deve desviar para a posição 9
	2	NOP	
	3	LDI \$170	Carrega acumulador com valor 170(10101010)
	4	STA @25	Armazena 170(10101010) na posição 25 da RAM
	5	STA @256	Armazena 170(10101010) na posição 256 da memória (habilita LED R)
	6	CEQ @25	A comparação deve fazer o flagigual ser 1
	7	JMP @1	Vai testar o flaglgual depois do jump
	8	NOP	
	9	LDI \$85	Carrega acumulador com valor 85(01010101)
	10	CEQ @25	Compara com valor 85(01010101), deve fazer o flaglgual ser 0
	11	JEQ @3	Não deve ocorrer o desvio
	12	JMP @12	Fim. Deve ficar neste laço

Pergunta 8 1,25 em 1,75 pontos

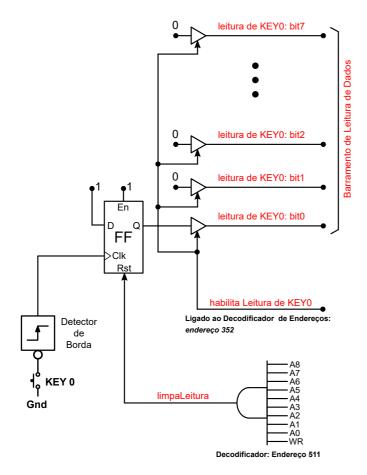
(Z)

abado o circuito abaixo, implemente, em VHDL, o top_level completo, com as entradas e saídas adequadas.

Considere que os componentes listados abaixo já existem e utilizam a configuração pelo generics, podendo fazer a instanciação dos mesmos:

- Edge Detector (detector de borda);
- Porta 3State configurável por generics;
- Flip-Flop.

Os outros componentes devem ser criados e interligados adequadamente.



Resposta Selecionada: cs Ctrl+M

```
use ieee.std_logic_1164.all;
entity top_level is
 port (
  CLK_50: in std_logic;
  KEY0: in std_logic;
  habilita_KEY0: in std_logic;
  leitura_KEY0: out std_logic;
  leitura_KEY1: out std_logic;
  leitura_KEY2: out std_logic;
  leitura_KEY3: out std_logic;
  leitura_KEY4: out std_logic;
  leitura_KEY5: out std_logic;
  leitura_KEY6: out std_logic;
  leitura_KEY7: out std_logic;
  entrada_decoder: in std_logic_vector(9 downto 0);
end entity;
architecture arquitetura of top_level is
 signal saida_edge : std_logic;
 signal saida_decoder: std_logic;
 signal saida_ff: std_logic;
 signal saida_3S0 : std_logic;
 signal saida_3S1: std_logic;
 signal saida_3S2: std_logic;
 signal saida_3S3: std_logic;
 signal saida_3S4 : std_logic;
 signal saida_3S5: std_logic;
 signal saida_3S6 : std_logic;
 signal saida_3S7: std_logic;
begin
CLK <= KEY0;
detectorSub0: work.edgeDetector(bordaSubida)
    port map (clk => CLK_50, entrada => (not KEY(0)), saida => CLK);
FF: entity work.flipFlop generic map (larguraDados => 1)
     port map (D =>'1', EN => '1', CLK => saida_edge, RST => saida_decoder, Q => saida_ff);
3S0: entity work.3State generic map(larguraDados =>1)
                       port map(DIN => saida_ff, EN => habilita_KEY0,
DOUT =>leitura_KEY0);
3S1: entity work.3State generic map(larguraDados =>1)
                       port map(DIN => '0', EN => habilita_KEY0,
DOUT =>leitura_KEY1);
3S2: entity work.3State generic map(larguraDados =>1)
                       port map(DIN => '0', EN => habilita_KEY0,
DOUT =>leitura_KEY2);
3S3: entity work.3State generic map(larguraDados =>1)
                       port map(DIN => '0', EN => habilita_KEY0,
DOUT =>leitura_KEY3);
3S4: entity work.3State generic map(larguraDados =>1)
                       port map(DIN => '0', EN => habilita_KEY0,
DOUT =>leitura_KEY4);
3S5: entity work.3State generic map(larguraDados =>1)
                       port map(DIN => '0', EN => habilita_KEY0,
DOUT =>leitura_KEY5);
3S6: entity work.3State generic map(larguraDados =>1)
                       port map(DIN => '0', EN => habilita_KEY0,
DOUT =>leitura_KEY6);
3S7: entity work.3State generic map(larguraDados =>1)
                       port map(DIN => '0', EN => habilita_KEY0,
DOUT =>leitura_KEY7);
DEC: entity work.decoderInstru port map(opcode => entrada_decoder,
saida => saida_decoder);
leitura_KEY0 <= saida_3S0;
leitura_KEY1 <= saida_3S1;
leitura_KEY2 <= saida_3S2;
leitura_KEY3 <= saida_3S3;
leitura_KEY4 <= saida_3S4;
leitura_KEY5 <= saida_3S5;
```

```
leitura_KEY6 <= saida_3S6;
leitura .KEY7 <= saida_3S0;
library ieee;
use ieee.std_logic_1164.all;
entity toplevel is
end architecture;
generic ( larguraDados : natural := 8);
 port (
  CLK_50: in std_logic;
  Addr: in std_logic_vector(8 downto 0);
  WR: in std_logic;
  KEY0: in std_logic;
  HabilitaLeituraKey0: in std_logic;
  leituraKey0 : out std_logic_vector(7 downto 0)
end toplevel;
architecture prova of toplevel is
signal saida_key0: std_logic_vector(7 downto 0);
signal saida_ff_Key0: std_logic;
signal limpaLeituraKey0: std_logic;
signal SaidaDetectorKey0: std_logic;
signal EntradaTristate: std_logic_vector(7 downto 0);
begin
detectorKey0: work.edgeDetector(bordaSubida)
        port map (clk => CLK_50, entrada => not KEY0, saida => SaidaDetectorKey0);
limpaLeituraKey0 <= WR and Addr(8) and Addr(7) and Addr(6) and Addr(5) and Addr(4) and Addr(3) and Addr(1)
FF_key: entity work.FlipFlop
     port map (DIN => '1',
          DOUT => saida_ff_Key0,
           ENABLE => '1',
           CLK => SaidaDetectorKey0,
           RST => limpaLeituraKey0);
EntradaTristate <= "0000000" & saida_ff_Key0;
TristateBuffer: entity work.BufferTristate generic map(larguraDados => larguraDados)
         port map(DIN => EntradaTristate, ENABLE => HabilitaLeituraKey0, DOUT => saida_key0);
leituraKey0 <= saida_key0;
end architecture;
Rubrica:
Não fez a entidade: descontar 30% da questão.
Não fez os sinais: descontar 10% da questão.
Não fez a arquitetura detalhada: descontar 10% da questão.
Outros erros: critério do avaliador.
```

Comentário da

Resposta Correta:

resposta:

Conexão leitura_KeY7 está errada

A saida do detector de borda e a entrada do clock do FF devertiam ser a mesma

não foi desenvolvido o decodificador

Sexta-feira, 21 de Abril de 2023 22h11min01s BRT

← 0K