ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
Departamento de Engenharia de Computação e Sistemas Digitais

PCS3635 – LABORATÓRIO DIGITAL I

EXPERIÊNCIA 3 – Projeto de uma Unidade de Controle

Relato da Bancada A3 – Turma 2 – Prof. Reginaldo

Data de Emissão: 20 de Janeiro DE 2025.

Nome: Pedro Henrique Zanato da Costa	**Número USP: 13874761**
Nome: Enzo Koichi Jojima	**Número USP: 14568285**
Nome: Eduardo Ribeiro do Amparo Rodrigues de Souza	**Número USP: 14567346**

1 Introdução

O experimento tem como objetivo desenvolver uma unidade de controle para um sistema digital simples, utilizando a linguagem Verilog e técnicas de projeto hierárquico. Essa unidade de controle é responsável por gerenciar o fluxo de dados de um circuito sequencial que realiza a leitura, comparação e processamento de 16 dados de 4 bits armazenados em memória, exibindo os resultados em displays de 7 segmentos. O projeto busca integrar o fluxo de dados e a unidade de controle, permitindo a simulação, validação e síntese em uma FPGA, para compreender e aplicar conceitos fundamentais de sistemas digitais, como máquinas de estados, sinais de controle e depuração.

2 Descrição do Projeto

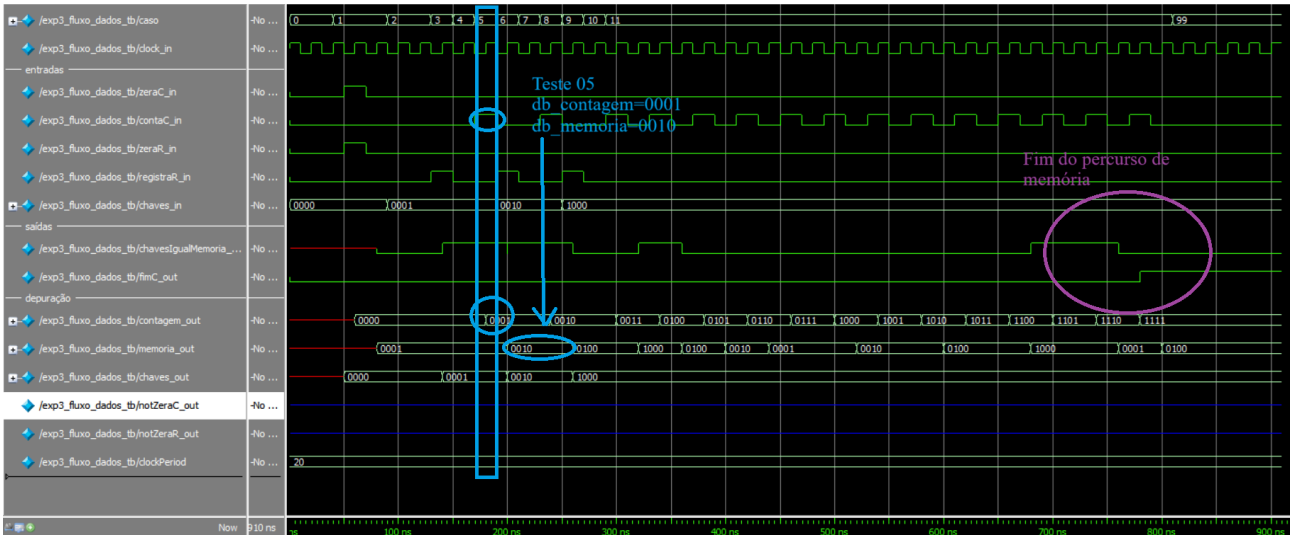
A visão conceitual do projeto descreve o desenvolvimento de uma unidade de controle para um sistema digital baseado em Verilog. A unidade de controle gerencia o

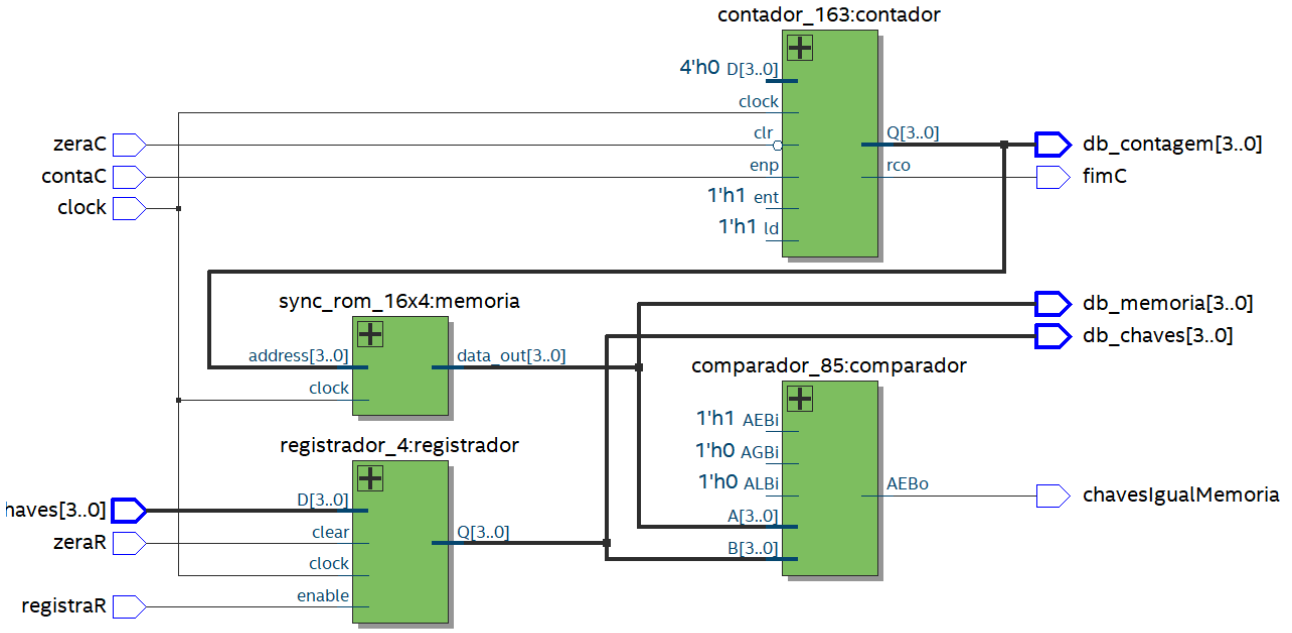
funcionamento de um circuito sequencial que armazena e compara 16 dados de 4 bits em uma memória interna. O sistema opera através de estados, ativados por sinais como reset e iniciar, executando ciclos de comparação e atualização de memória. Os resultados de debug, como chaves, estado atual e contagem, são exibidos nos displays de 7 segmentos da FPGA. Ao final, o sinal pronto indica a conclusão do processo.

O funcionamento é guiado pelo pseudocódigo descrito e representado no diagrama de estados. O fluxo de dados, adaptado da experiência anterior, é integrado com a unidade de controle para formar um sistema hierárquico. Um fluxograma do processo incluiria estados como "Espera", "Inicializa", "Compara", "Atualiza Contador" e "Finaliza", permitindo a visualização clara das transições e ações. Isso demonstra a organização incremental e modular do desenvolvimento, possibilitando simulação e síntese em FPGA para testes e validação prática.

3 Detalhamento do Projeto Lógico

3.1 Projeto do Fluxo de Dados





O projeto do fluxo de dados apresentado é estruturado para processar um sistema digital sequencial, integrando os principais componentes: contador, memória sincronizada, registrador e comparador. Cada um desempenha um papel fundamental no processamento de dados e na sincronização com os sinais de controle. A memória armazenada contém 16 dados de 4 bits, que são endereçados pelo contador. O comparador verifica se o valor armazenado no registrador, derivado das chaves de entrada, é equivalente ao valor atual da memória.

O fluxo de dados opera conforme descrito no diagrama. O contador, controlado pelos sinais **zeraC** e **contaC**, avança sequencialmente pelos endereços da memória. O registrador, habilitado por **zeraR** e **registraR**, armazena os dados das chaves para serem comparados. O comparador então avalia a igualdade entre as saídas da memória e do registrador, gerando o sinal **chavesIgualMemoria**, enquanto os valores de debug, como **db_contagem**, **db_memoria**, e **db_chaves**, são exibidos para monitoramento. Esse design modular facilita a integração com a unidade de controle, garantindo uma execução eficiente e precisa do sistema.

3.2 Projeto da Unidade de Controle

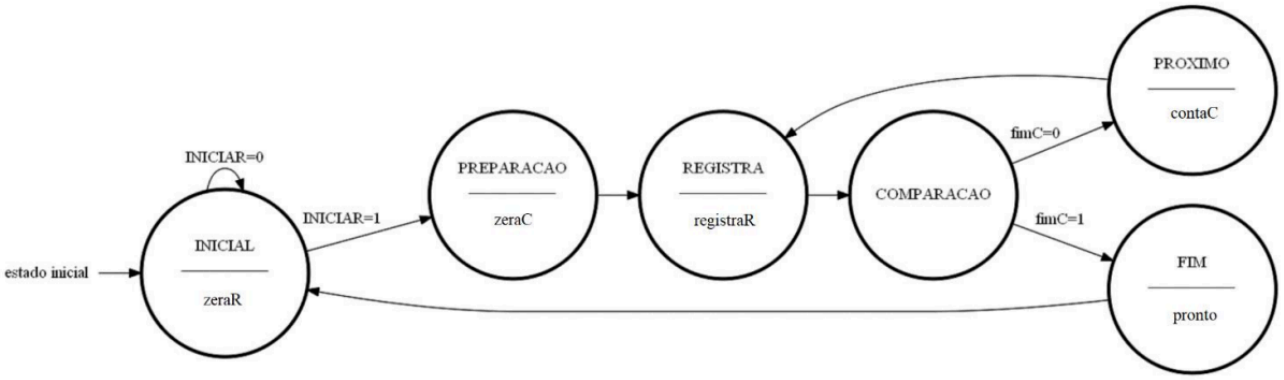


Tabela 1 – Descrição da Unidade de Controle do Sistema

Nome do Estado	Descrição do Estado	Próximo Estado	Condições e Justificativas para a Transição entre Estados
Inicial	Estado inicial, responsável por zerar o registrador	Preparação	Quando o iniciar é ativado ele muda de estado na subida de clock. Até lá continua no estado inicial.
Preparação	Zera o Contador	Registra	Na próxima subida de clock
Registra	Registrar o valor das chaves	Comparação	Na próxima subida de clock
Comparação	Verifica se a contagem terminou e compara o valor do contador e as chaves.	1: Fim 2:Próximo	1: se fimC = 1 vai para o estado Fim na próxima subida de clock 2: se fimC = 0 vai para o estado Próximo na próxima subida de clock
Próximo	acrescenta um na contagem do contador	Registra	Na Próxima subida de clock
Fim	Indica o fim	-	Só muda de estado com reset
			Obs: Todos os estados podem voltar pro inicial com o aperto do reset

3.3 Projeto do Sistema Digital

Integração:

- **Sinais de Controle:** A unidade de controle emite sinais como **zeraC**, **contaC**, **zeraR** e **registraR**, que controlam o comportamento do contador e do registrador no fluxo de dados. Por exemplo, no estado "Inicial", o sinal **zeraR** é ativado para limpar o registrador, e no estado "Próximo", o sinal **contaC** incrementa o contador para acessar o próximo endereço de memória.

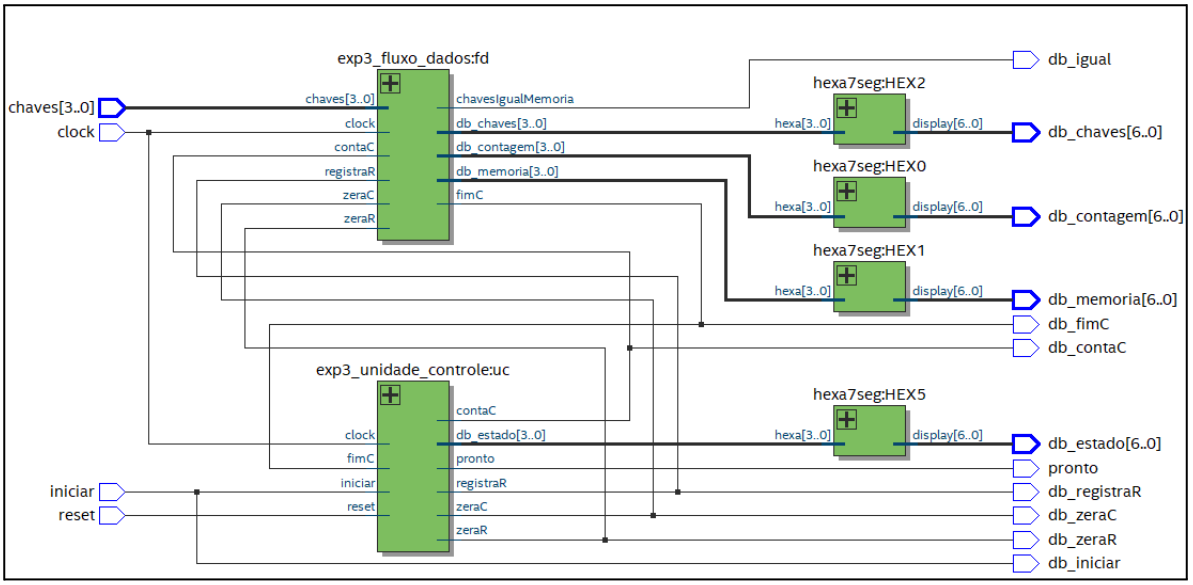
- **Sinais de Condição:** O fluxo de dados gera sinais de condição como **fimC** (indicando que a contagem alcançou o final da memória) e **chavesIgualMemoria** (verificando a igualdade entre chaves de entrada e dados da memória), que guiam as transições de estado na unidade de controle.

Sinais de Depuração:

Sinais de depuração são fundamentais para monitorar o comportamento do sistema e diagnosticar falhas:

- **db_estado:** Indica o estado atual da unidade de controle.
- **db_contagem:** Apresenta o endereço atual do contador, útil para rastrear a progressão na memória.
- **db_memoria:** Mostra o dado atualmente acessado na memória.
- **db_chaves:** Exibe o valor armazenado no registrador das chaves.
- **db_igual:** Indica se houve correspondência entre o dado da memória e o valor das chaves.
- **pronto:** Sinal final que confirma a conclusão do ciclo de operações.

Esses sinais, exibidos em displays de 7 segmentos e LEDs, são úteis para depurar e validar o comportamento do sistema durante a simulação e testes na FPGA, assegurando que cada transição de estado e operação do fluxo de dados ocorre como planejado.

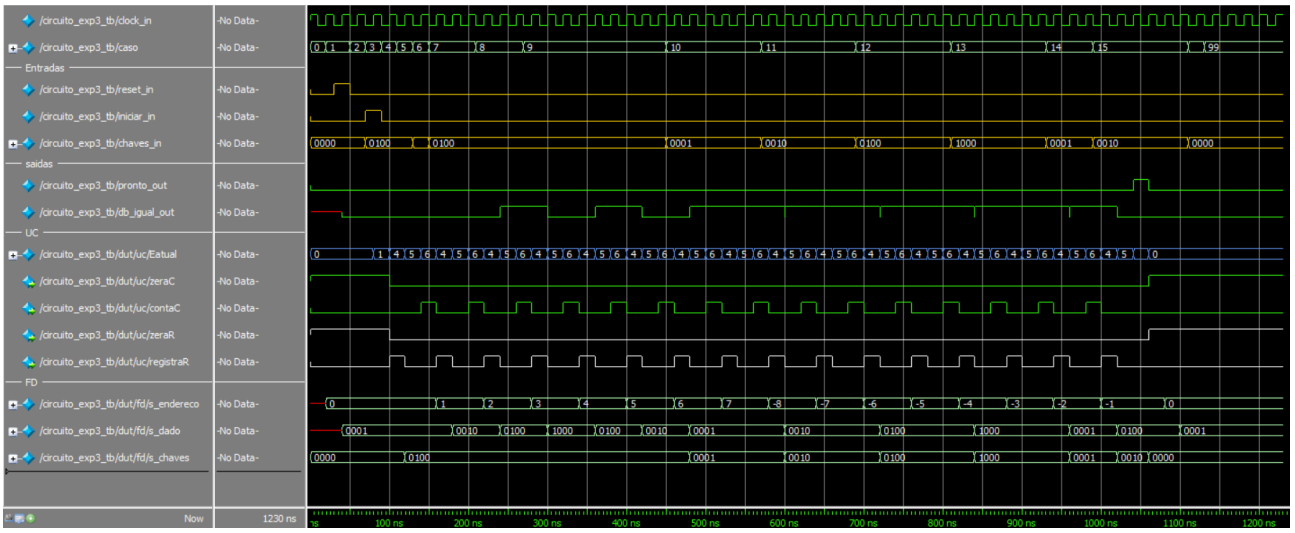


4 Plano de Testes do Sistema e Simulações

4.1 Cenário de Teste 1 – Teste do Fluxo de Dados

O plano de testes do fluxo de dados está descrito no enunciado da experiência. Os resultados da simulação com o Model Sim estáo na figura abaixo da tabela. Nota-se que os resultados obtidos correspondem aos esperados.

#	Operação	Sinais de controle	Resultado esperado
c.i.	Condições iniciais (todas as entradas desativadas)	clock=0, zeraC=0, contaC=0, zeraR=0, registraR=0, chaves=0000	chavesIgualMemoria=0, fimC=0, db_contagem=0000, db_memoria=0001, db_chaves=0000
1	Zerar contador e registrador (manter outras entradas desativadas)	zeraC=1, zeraR=1, clock ↑	chavesIgualMemoria=0, fimC=0, db_contagem=0000, db_memoria=0001, db_chaves=0000
2	Verificar saídas com chaves=0001 (manter outras entradas desativadas)	zeraC=0, contaC=0, zeraR=0, registraR=0, chaves=0001, clock ↑	chavesIgualMemoria=0, fimC=0, db_contagem=0000, db_memoria=0001, db_chaves=0000
3	Registrar chaves com chaves=0001 (manter outras entradas desativadas)	chaves=0001, registrarR=1, clock ↑	chavesIgualMemoria=1, fimC=0, db_contagem=0000, db_memoria=0001, db_chaves=0001
4	Verificar saídas com chaves=0001 (manter outras entradas desativadas)	chaves=0001, clock ↑	chavesIgualMemoria=1, fimC=0, db_contagem=0000, db_memoria=0001, db_chaves=0001
5	Incrementar contador (manter outras entradas desativadas)	contaC=1, clock ↑	chavesIgualMemoria=0, fimC=0, db_contagem=0001, db_memoria=0010, db_chaves=0001
6	Registrar chaves com chaves=0010 (manter outras entradas desativadas)	chaves=0010 clock ↑	chavesIgualMemoria=1, fimC=0, db_contagem=0001, db_memoria=0010, db_chaves=0010
7	Verificar saídas com chaves=0010 (manter outras entradas desativadas)	chaves=0010 clock ↑	chavesIgualMemoria=1, fimC=0, db_contagem=0001, db_memoria=0010, db_chaves=0010
8	Incrementar contador (manter outras entradas desativadas)	contaC=1, clock ↑	chavesIgualMemoria=0, fimC=0, db_contagem=0010, db_memoria=0100, db_chaves=0010
9	Registrar chaves com chaves=1000 (manter outras entradas desativadas)	chaves=1000 clock ↑	chavesIgualMemoria=0, fimC=0, db_contagem=0010, db_memoria=0100, db_chaves=1000
10	Verificar saídas com chaves=1000 (manter outras entradas desativadas)	chaves=1000 clock ↑	chavesIgualMemoria=0, fimC=0, db_contagem=0010, db_memoria=0100, db_chaves=1000
11	Incrementar contador até final da contagem	contaC=1 clock ↑ (13x)	chavesIgualMemoria=0, fimC=1, db_contagem=1111, db_memoria=0100, db_chaves=1000



4.2 Cenário de Teste 2 – Teste do Circuito

Cenário presente no guia da experiência para teste do circuito.

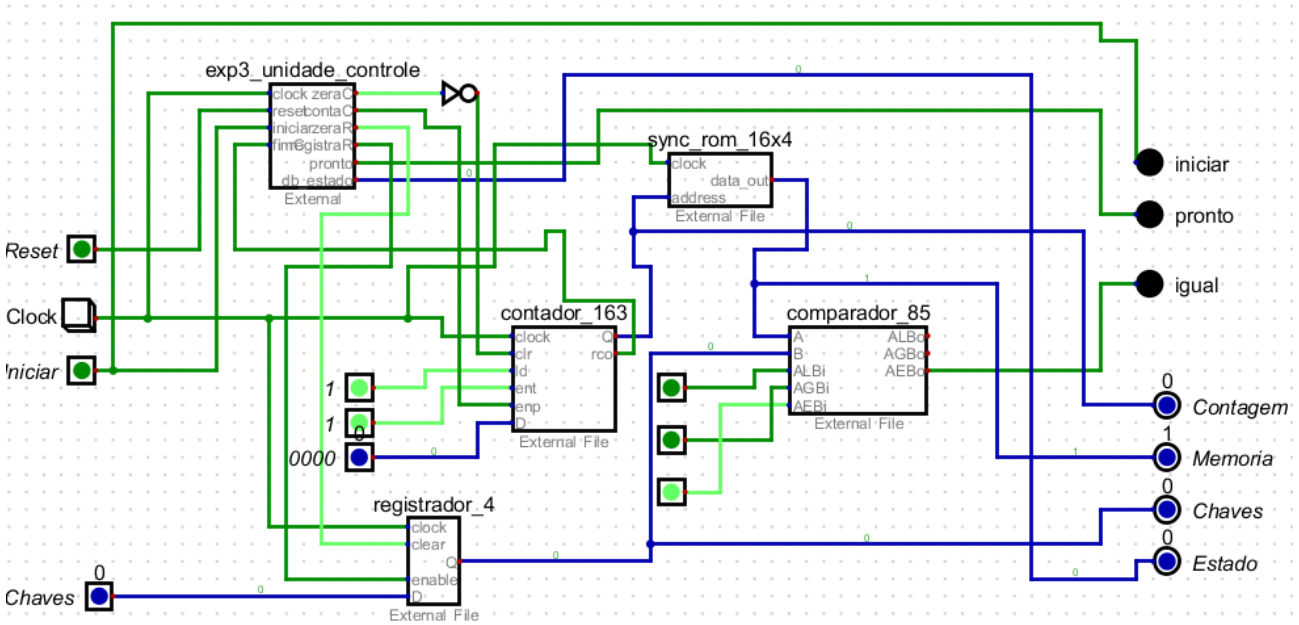
#	Operação	Sinais de Controle	Resultado esperado
c.i	Condições iniciais	clock=0 reset=0 iniciar=0 chaves=0000	pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0000
1	“Resetar” circuito e observar a saída da memória	reset=1 clock	pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0000
2	Acionar sinal de clock 5 vezes com iniciar=0	reset=0 iniciar=0 clock (5x)	(permanece no estado inicial) pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0000
3	Ajustar chaves para 0100, ativar iniciar=1 e acionar clock 1x	chaves=0100 iniciar=1 clock	(muda para estado preparação) pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001

			db_chaves=0000 db_estado=0001
4	Mantém chaves em 0100 e acionar clock 1x	chaves=0100 iniciar=0 clock	(muda para estado registra) pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0100
5	Mantém chaves em 0100 e acionar clock 1x	chaves=0100 iniciar=0 clock	(muda para estado comparação) pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0100 db_estado=0101
6	Mantém chaves em 0100 e acionar clock 1x	chaves=0100 clock	(muda para estado próximo) pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0100 db_estado=0110
7	Mantém chaves em 0100 e acionar clock 3x	chaves=0100 clock (3x)	(passa pelos estados registra, comparação e próximo) pronto=0 db_igual=0 db_contagem=0001 db_memoria=0010 db_chaves=0100 db_estado=0110
8	Mantém chaves em 0100 e acionar clock 3x	chaves=0100 clock (3x)	(passa pelos estados registra, comparação e próximo) pronto=0 db_igual=1 db_contagem=0010 db_memoria=0100 db_chaves=0100 db_estado=0110

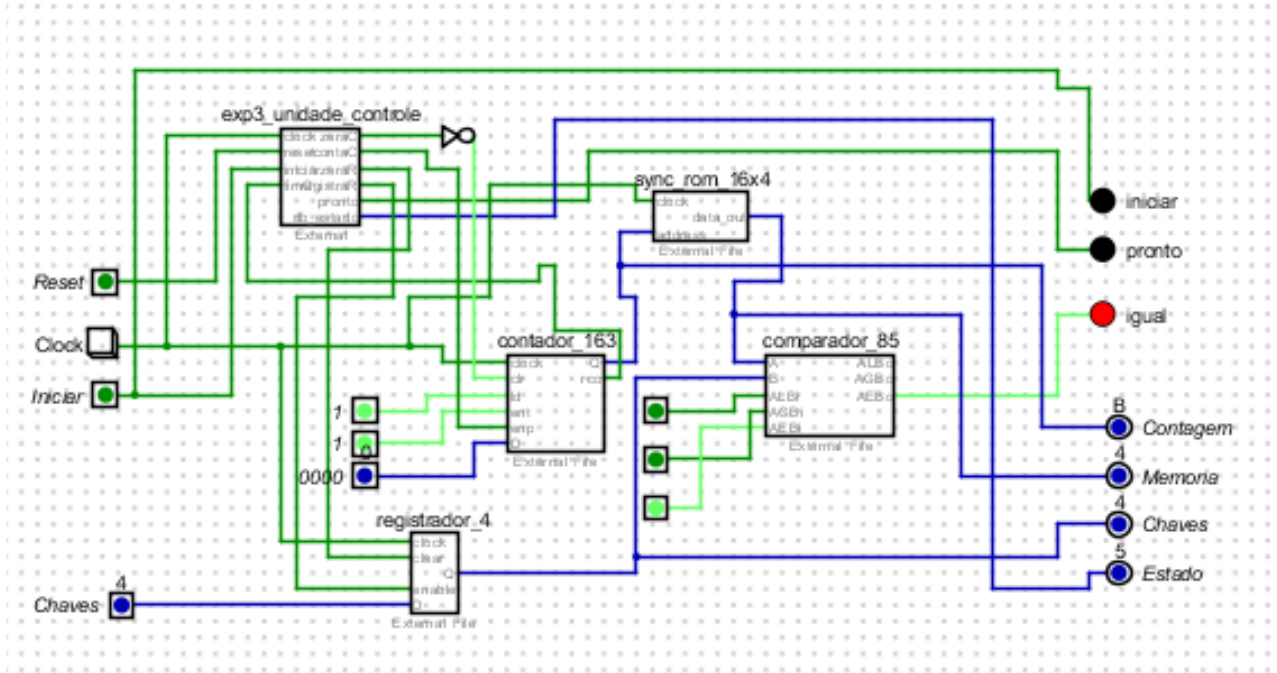
9	Mantém chaves em 0100 e acionar clock 9x	chaves=0100 clock (9x)	(passa 3x pelos estados registra, comparação e próximo) pronto=0 db_igual=0 db_contagem=0101 db_memoria=0010 db_chaves=0100 db_estado=0110
10	Ajustar chaves para 0001 e acionar clock 6x	chaves=0001 clock (6x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 db_igual=1 db_contagem=0111 db_memoria=0001 db_chaves=0001 db_estado=0110
11	Ajustar chaves para 0010 e acionar clock 6x	chaves=0010 clock (6x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 db_igual=1 db_contagem=1001 db_memoria=0010 db_chaves=0010 db_estado=0110
12	Ajustar chaves para 0100 e acionar clock 6x	chaves=0100 clock (6x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 db_igual=1 db_contagem=1011 db_memoria=0100 db_chaves=0100 db_estado=0110
13	Ajustar chaves para 1000 e acionar clock 6x	chaves=1000 clock (6x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 db_igual=1 db_contagem=1101 db_memoria=1000

			db_chaves=1000 db_estado=0110
14	Ajustar chaves para 0000 e acionar clock 3x	chaves=0001 clock (3x)	(passa pelos estados registra, comparação e proximo) pronto=0 db_igual=1 db_contagem=1110 db_memoria=0001 db_chaves=0001 db_estado=0110
15	Mantém chaves em 0000 e acionar clock 3x	chaves=0010 clock (3x)	(passa pelo estado fim) pronto=1 db_igual=0 db_contagem=1111 db_memoria=0100 db_chaves=0010 db_estado=1111
16	Mantém chaves em 0000 e acionar clock	chaves=0000 clock	(termina no estado inicial) pronto=0 db_igual=0 db_contagem=1111 db_memoria=0100 db_chaves=0000 db_estado=0000

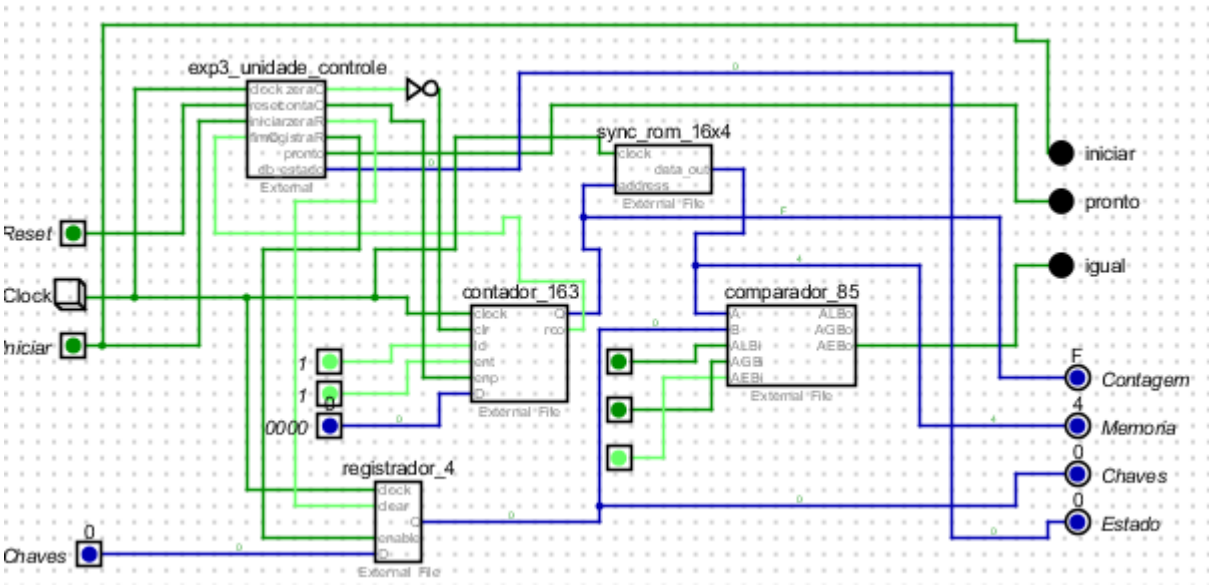
Simulação com Digital:



condições iniciais,



operação 12



operação 16

5 Implantação do Projeto

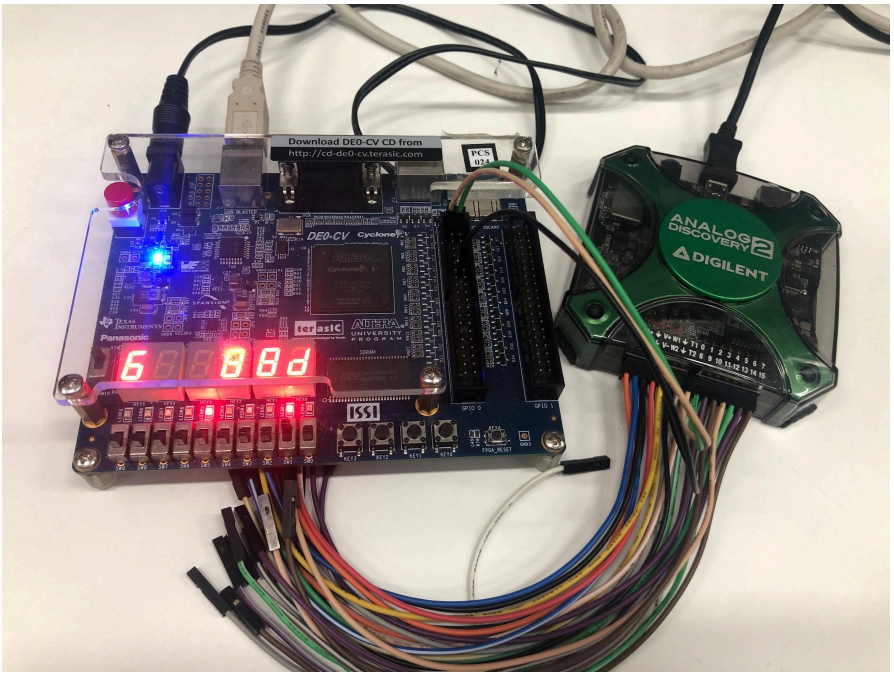
5.1 Pinagem da Placa FPGA

Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Slew Rate
chaves[3]	Input	PIN_AA15	4A	B4A_N0	PIN_AA15	2.5 V		12mA (default)	
chaves[2]	Input	PIN_T12	4A	B4A_N0	PIN_T12	2.5 V		12mA (default)	
chaves[1]	Input	PIN_T13	4A	B4A_N0	PIN_T13	2.5 V		12mA (default)	
chaves[0]	Input	PIN_V13	4A	B4A_N0	PIN_V13	2.5 V		12mA (default)	
clock	Input	PIN_N16	5B	B5B_N0	PIN_N16	2.5 V		12mA (default)	
db_chaves[6]	Output	PIN_AB21	4A	B4A_N0	PIN_AB21	2.5 V		12mA (default)	1 (default)
db_chaves[5]	Output	PIN_AB22	4A	B4A_N0	PIN_AB22	2.5 V		12mA (default)	1 (default)
db_chaves[4]	Output	PIN_V14	4A	B4A_N0	PIN_V14	2.5 V		12mA (default)	1 (default)
db_chaves[3]	Output	PIN_Y14	4A	B4A_N0	PIN_Y14	2.5 V		12mA (default)	1 (default)
db_chaves[2]	Output	PIN_AA10	3B	B3B_N0	PIN_AA10	2.5 V		12mA (default)	1 (default)
db_chaves[1]	Output	PIN_AB17	4A	B4A_N0	PIN_AB17	2.5 V		12mA (default)	1 (default)
db_chaves[0]	Output	PIN_Y19	4A	B4A_N0	PIN_Y19	2.5 V		12mA (default)	1 (default)
db_contaC	Output	PIN_N1	2A	B2A_N0	PIN_N1	2.5 V		12mA (default)	1 (default)
db_contagem[6]	Output	PIN_AA22	4A	B4A_N0	PIN_AA22	2.5 V		12mA (default)	1 (default)
db_contagem[5]	Output	PIN_Y21	4A	B4A_N0	PIN_Y21	2.5 V		12mA (default)	1 (default)
db_contagem[4]	Output	PIN_Y22	4A	B4A_N0	PIN_Y22	2.5 V		12mA (default)	1 (default)
db_contagem[3]	Output	PIN_W21	4A	B4A_N0	PIN_W21	2.5 V		12mA (default)	1 (default)
db_contagem[2]	Output	PIN_W22	4A	B4A_N0	PIN_W22	2.5 V		12mA (default)	1 (default)
db_contagem[1]	Output	PIN_V21	4A	B4A_N0	PIN_V21	2.5 V		12mA (default)	1 (default)
db_contagem[0]	Output	PIN_U21	4A	B4A_N0	PIN_U21	2.5 V		12mA (default)	1 (default)
db_estado[6]	Output	PIN_W19	4A	B4A_N0	PIN_W19	2.5 V		12mA (default)	1 (default)
db_estado[5]	Output	PIN_C2	2A	B2A_N0	PIN_C2	2.5 V		12mA (default)	1 (default)
db_estado[4]	Output	PIN_C1	2A	B2A_N0	PIN_C1	2.5 V		12mA (default)	1 (default)
db_estado[3]	Output	PIN_P14	4A	B4A_N0	PIN_P14	2.5 V		12mA (default)	1 (default)
db_estado[2]	Output	PIN_T14	4A	B4A_N0	PIN_T14	2.5 V		12mA (default)	1 (default)
db_estado[1]	Output	PIN_M8	3B	B3B_N0	PIN_M8	2.5 V		12mA (default)	1 (default)
db_estado[0]	Output	PIN_N9	3B	B3B_N0	PIN_N9	2.5 V		12mA (default)	1 (default)
db_fimC	Output	PIN_U2	2A	B2A_N0	PIN_U2	2.5 V		12mA (default)	1 (default)
db_igual	Output	PIN_AA1	2A	B2A_N0	PIN_AA1	2.5 V		12mA (default)	1 (default)
db_iniciar	Output	PIN_W2	2A	B2A_N0	PIN_W2	2.5 V		12mA (default)	1 (default)
db_memoria[6]	Output	PIN_U22	4A	B4A_N0	PIN_U22	2.5 V		12mA (default)	1 (default)
db_memoria[5]	Output	PIN_AA17	4A	B4A_N0	PIN_AA17	2.5 V		12mA (default)	1 (default)
db_memoria[4]	Output	PIN_AB18	4A	B4A_N0	PIN_AB18	2.5 V		12mA (default)	1 (default)
db_memoria[3]	Output	PIN_AA18	4A	B4A_N0	PIN_AA18	2.5 V		12mA (default)	1 (default)

db_memoria[4]	Output	PIN_AB18	4A	B4A_N0	PIN_AB18	2.5 V		12mA (default)	1 (default)
db_memoria[3]	Output	PIN_AA18	4A	B4A_N0	PIN_AA18	2.5 V		12mA (default)	1 (default)
db_memoria[2]	Output	PIN_AA19	4A	B4A_N0	PIN_AA19	2.5 V		12mA (default)	1 (default)
db_memoria[1]	Output	PIN_AB20	4A	B4A_N0	PIN_AB20	2.5 V		12mA (default)	1 (default)
db_memoria[0]	Output	PIN_AA20	4A	B4A_N0	PIN_AA20	2.5 V		12mA (default)	1 (default)
db_registraR	Output	PIN_L1	2A	B2A_N0	PIN_L1	2.5 V		12mA (default)	1 (default)
db_zeraC	Output	PIN_N2	2A	B2A_N0	PIN_N2	2.5 V		12mA (default)	1 (default)
db_zeraR	Output	PIN_L2	2A	B2A_N0	PIN_L2	2.5 V		12mA (default)	1 (default)
iniciar	Input	PIN_U13	4A	B4A_N0	PIN_U13	2.5 V		12mA (default)	
pronto	Output	PIN_AA2	2A	B2A_N0	PIN_AA2	2.5 V		12mA (default)	1 (default)
reset	Input	PIN_B16	7A	B7A_N0	PIN_B16	2.5 V		12mA (default)	

5.2 Estratégia de Montagem

Primeiramente devemos aterrar a analog discovery interligando os canais GND da uc com os pinos GPIO_0 da DE0-CV. Temos que então programar o programa sintetizado para a placa DE0_CV. Fazemos isso usando o Quartus. A ligação da placa e feita da seguinte forma:



Os principais sinais que podemos observar são:db_contagem, db_igual, db_contagem, db_memoria e db_estado.

Observamos eles nos seguintes lugares:
hex_0: db_contagem;
hex_1: db_memoria;
hex_2: db_chaves;
hex_5: db_estado
led_1: db_igual

5.3 Estratégia de Depuração

A depuração do circuito é feito em um primeiro momento através dos softwares modelsim e digital, através deles garantimos que qualquer erro não é de programação. O próximo passo é então sintetizar o código na FPGA e durante os testes monitorar os sinais de depuração, que estão pintados em diferentes componentes da placa, dependendo principalmente da quantidade de bits necessários para a representação de um dado sinal. caso ser identificado algum erro, temos que verificar a montagem.

5.4 Execução Prática do Cenário de Teste 1 – Teste do Circuito

Tabela 4 – Descrição e Resultados Práticos do Cenário de Teste 1

#	Operação	Sinais de Controle	Resultado esperado
c.i	Condições iniciais	clock=0 reset=0 iniciar=0 chaves=0000	pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0000
1	“Resetar” circuito e observar a saída da memória	reset=1 clock	pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0000
2	Acionar sinal de clock 5 vezes com iniciar=0	reset=0 iniciar=0 clock (5x)	(permanece no estado inicial) pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0000
3	Ajustar chaves para 0100, ativar iniciar=1 e acionar clock 1x	chaves=0100 iniciar=1 clock	(muda para estado preparação) pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0001
4	Mantém chaves em 0100 e acionar clock 1x	chaves=0100 iniciar=0 clock	(muda para estado registra) pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0100
5	Mantém chaves em 0100 e acionar clock 1x	chaves=0100 iniciar=0 clock	(muda para estado comparação) pronto=0 db_igual=0 db_contagem=0000

			db_memoria=0001 db_chaves=0100 db_estado=0101
6	Mantém chaves em 0100 e acionar clock 1x	chaves=0100 clock	(muda para estado próximo) pronto=0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0100 db_estado=0110
7	Mantém chaves em 0100 e acionar clock 3x	chaves=0100 clock (3x)	(passa pelos estados registra, comparação e próximo) pronto=0 db_igual=0 db_contagem=0001 db_memoria=0010 db_chaves=0100 db_estado=0110
8	Mantém chaves em 0100 e acionar clock 3x	chaves=0100 clock (3x)	(passa pelos estados registra, comparação e próximo) pronto=0 db_igual=1 db_contagem=0010 db_memoria=0100 db_chaves=0100 db_estado=0110
9	Mantém chaves em 0100 e acionar clock 9x	chaves=0100 clock (9x)	(passa 3x pelos estados registra, comparação e próximo) pronto=0 db_igual=0 db_contagem=0101 db_memoria=0010 db_chaves=0100 db_estado=0110
10	Ajustar chaves para 0001 e acionar clock 6x	chaves=0001 clock (6x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 db_igual=1

			db_contagem=0111 db_memoria=0001 db_chaves=0001 db_estado=0110
11	Ajustar chaves para 0010 e acionar clock 6x	chaves=0010 clock (6x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 db_igual=1 db_contagem=1001 db_memoria=0010 db_chaves=0010 db_estado=0110
12	Ajustar chaves para 0100 e acionar clock 6x	chaves=0100 clock (6x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 db_igual=1 db_contagem=1011 db_memoria=0100 db_chaves=0100 db_estado=0110
13	Ajustar chaves para 1000 e acionar clock 6x	chaves=1000 clock (6x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 db_igual=1 db_contagem=1101 db_memoria=1000 db_chaves=1000 db_estado=0110
14	Ajustar chaves para 0000 e acionar clock 3x	chaves=0001 clock (3x)	(passa pelos estados registra, comparação e proximo) pronto=0 db_igual=1 db_contagem=1110 db_memoria=0001 db_chaves=0001 db_estado=0110
15	Mantém chaves em 0000 e acionar clock 3x	chaves=0010 clock (3x)	(passa pelo estado fim) pronto=1

			db_igual=0 db_contagem=1111 db_memoria=0100 db_chaves=0010 db_estado=1111
16	Mantém chaves em 0000 e acionar clock	chaves=0000 clock	(termina no estado inicial) pronto=0 db_igual=0 db_contagem=1111 db_memoria=0100 db_chaves=0000 db_estado=0000

A nossa simulação bateu com o resultado esperado em todos os itens, por isso estão todos verdes.

6 Projeto do Desafio da Experiência

6.1 Descrição do Desafio

O desafio propõe implementar sinais de acertou e errou para os palpites do jogador, bem como uma nova condição dentro da lógica do sistema. As alterações estão propostas em azul:

```
Algoritmo: sistema digital simples modificado
    entradas: iniciar, chaves
    saídas: pronto, acertou, errou
    depuração: contagem, memória, estado, igual
1.  {
2.      while (verdadeiro) {
3.          espera acionamento do sinal iniciar
4.          inicia circuito para condições iniciais
5.          while (não atingiu o último dado e acertou todos os dados) {
                // continua enquanto chaves=dado e não chegou no
                // final da memória
6.              compara chaves de entrada com dados armazenados
7.              incrementa contador interno
8.          }
9.          ativa acertou se acertou todos os dados da memória
10.         ativa errou se errou um dado
11.         ativa saída pronto
12.         }
13.  }
```

6.2 Descrição do Projeto Lógico

O projeto lógico foi a implementação de um novo estado e de novas condições para cada transição de estados. As alterações estão descritas a seguir.

6.2.1 Alterações do Fluxo de Dados

A única alteração sofrida pelo fluxo de dados foi o redirecionamento do sinal chavesIgualMemoria como um sinal de condição para a unidade de controle.

6.2.2 Alterações da Unidade de Controle

Adicionamos um novo estado: “erra (2)” e alteramos a lógica de transição de estados para adicionar a nova condicional proposta.

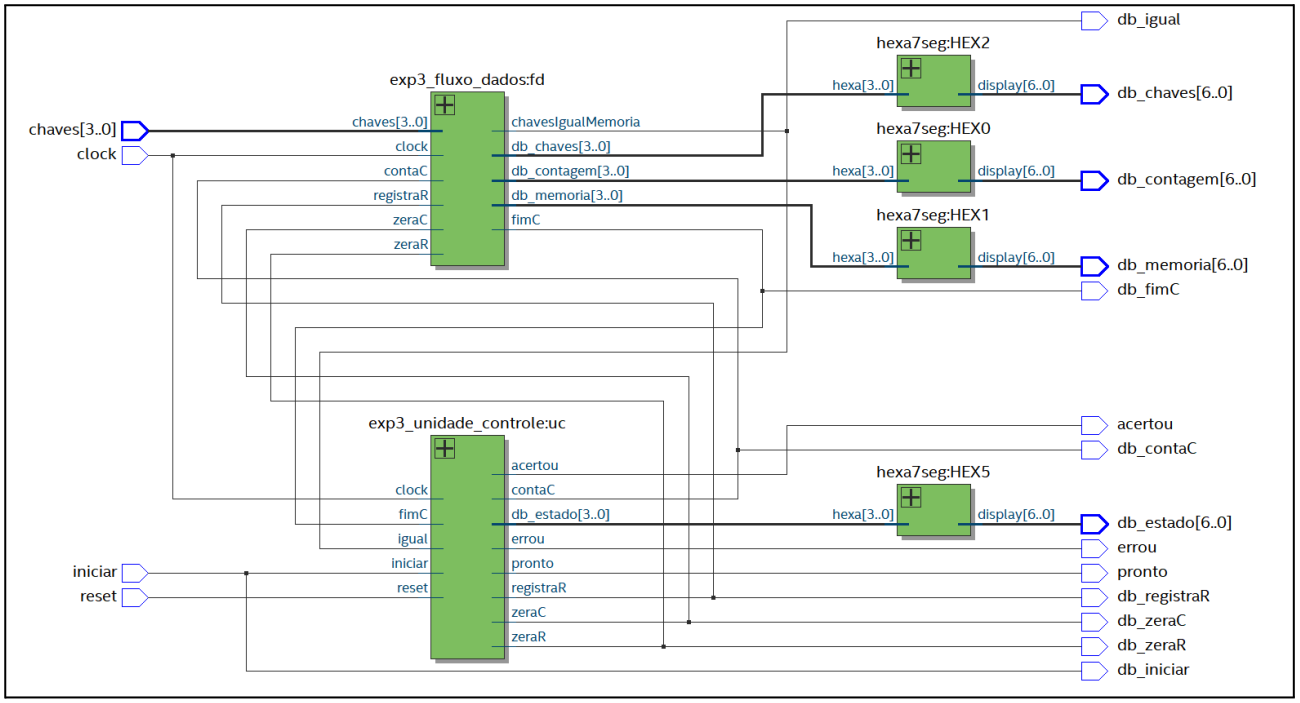
```
comparacao:  Eprox = ~fimC && igual ? proximo : ~igual ? erra : fim;
erra: Eprox = inicial;
```

Já na lógica de saída, adicionamos os sinais para as saídas de acertou e errou.

```
pronto    = (Eatual == fim || Eatual == erra) ? 1'b1 : 1'b0;
acertou = (Eatual == erra) ? 1'b0 : 1'b1;
errou = (Eatual == erra) ? 1'b1 : 1'b0;
```

6.2.3 Alterações do Sistema Digital

As alterações no sistema completo podem ser vistas a seguir.



Note que possuimos novos sinais de saída: acertou e errou, que correspondem aos sinais pedidos no enunciado. Além disso, a saída chavesIgualMemoria agora é também uma input da unidade de controle, isto é, um sinal de condição.

6.3 Verificação e Validação do Desafio

Os testes feitos seguem o padrão sugerido pelos professores. O primeiro caso no qual o jogador acerta todos os palpites, e o segundo teste no qual o jogador erra na quarta rodada.

Os testes foram realizados no Model Sim e o cenário 1 foi realizado na placa FPGA do Laboratório Digital.

6.3.1 Cenário de Teste 1 – Acertar todos os sinais

Descrever cenário em até dois parágrafos e detalhar as características técnicas em uma tabela com a seguinte estrutura.

Tabela 6 – Descrição e Resultados do Cenário de Teste 1 para o Desafio

#	Operação	Sinais de Controle	Resultado esperado
c.i	Condições iniciais	clock=0 reset=0 iniciar=0	pronto=0 acertou = 1 errou = 0

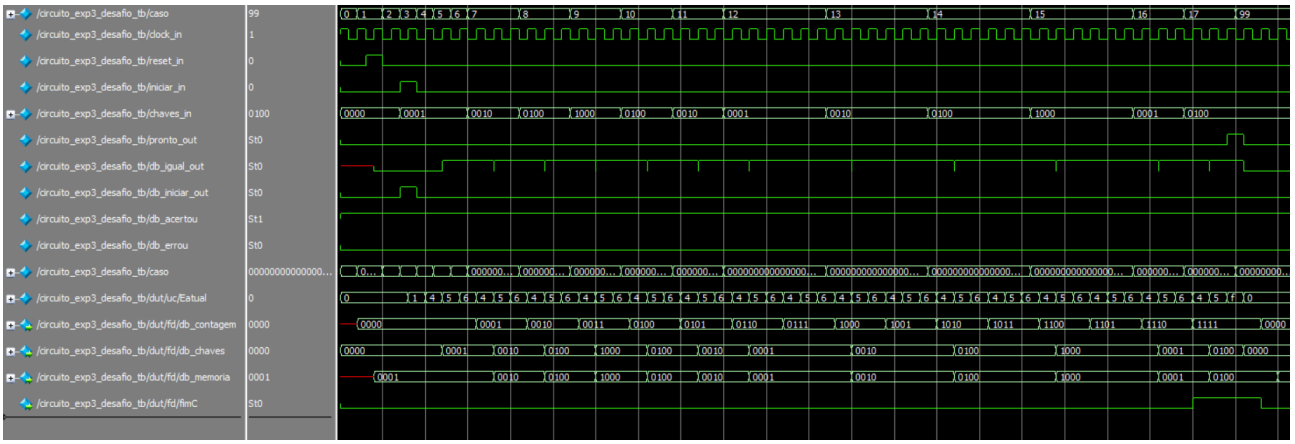
		chaves=0000	db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0000
1	“Resetar” circuito e observar a saída da memória	reset=1 iniciar=0 clock	pronto=0 acertou = 1 errou = 0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0000
2	Acionar sinal de clock 1 vezes com iniciar=0	reset=0 iniciar=0 clock (1x)	(permanece no estado inicial) pronto=0 acertou = 1 errou = 0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0000
3	Ajustar chaves para 0001, ativar iniciar=1 e acionar clock 1x	chaves=0001 iniciar=1 clock	(muda para estado preparação) pronto=0 acertou = 1 errou = 0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0001 db_estado=0001
4	Mantém chaves em 0001 e acionar clock 1x	chaves=0001 iniciar=0 clock	(muda para estado registra) pronto=0 acertou = 1 errou = 0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0100
5	Mantém chaves em 0001 e acionar clock 1x	chaves=0001 iniciar=0 clock	(muda para estado comparação) pronto=0 acertou = 1

			errou = 0 db_igual= 1 db_contagem=0000 db_memoria=0001 db_chaves=0001 db_estado=0101
6	Mantém chaves em 0001 e acionar clock 1x	chaves=0001 clock	(muda para estado próximo) pronto=0 acertou = 1 errou = 0 db_igual= 1 db_contagem=0001 db_memoria=0001 db_chaves=0001 db_estado=0110
7	Ajustar chaves para 0010 e acionar clock 3x	chaves=0010 clock (3x)	(passa pelos estados registra, comparação e próximo) pronto=0 acertou = 1 errou = 0 db_igual= 1 db_contagem=0001 db_memoria=0010 db_chaves=0010 db_estado=0110
8	Ajustar chaves para 0100 e acionar clock 3x	chaves=0100 clock (3x)	(passa pelos estados registra, comparação e próximo) pronto=0 acertou = 1 errou = 0 db_igual= 1 db_contagem=0010 db_memoria=0100 db_chaves=0100 db_estado=0110
9	Ajustar chaves para 1000 e acionar clock 3x	chaves=1000 clock (3x)	(passa pelos estados registra, comparação e próximo) pronto=0 acertou = 1 errou = 0 db_igual=1

			db_contagem=0011 db_memoria=1000 db_chaves=1000 db_estado=0110
10	Ajustar chaves para 0100 e acionar clock 3x	chaves=0100 clock (3x)	(passa 3x pelos estados registra, comparação e próximo) pronto=0 acertou = 1 errou = 0 db_igual = 1 db_contagem=0100 db_memoria=0100 db_chaves=0100 db_estado=0110
11	Ajustar chaves para 0010 e acionar clock 3x	chaves=0010 clock (3x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 acertou = 1 errou = 0 db_igual=1 db_contagem=0101 db_memoria=0010 db_chaves=0010 db_estado=0110
12	Ajustar chaves para 0001 e acionar clock 6x	chaves=0001 clock (6x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 acertou = 1 errou = 0 db_igual=1 db_contagem=0111 db_memoria=0001 db_chaves=0001 db_estado=0110
13	Ajustar chaves para 0010 e acionar clock 6x	chaves=0010 clock (6x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 acertou = 1 errou = 0 db_igual=1

			db_contagem=1001 db_memoria=0010 db_chaves=0010 db_estado=0110
14	Ajustar chaves para 0100 e acionar clock 6x	chaves=0100 clock (6x)	(passa pelos estados registra, comparação e proximo 2x) pronto=0 acertou = 1 errou = 0 db_igual=1 db_contagem=1011 db_memoria=0100 db_chaves=0100 db_estado=0110
15	Ajustar chaves para 1000 e acionar clock 6x	chaves=1000 clock (6x)	(passa pelos estados registra, comparação e proximo) pronto=0 acertou = 1 errou = 0 db_igual=1 db_contagem=1101 db_memoria=1000 db_chaves=1000 db_estado=0110
16	Ajustar as chaves em 0001 e acionar clock 3x	chaves = 0001 clock (3x)	(passa pelos estados registra, comparação e proximo) pronto=0 acertou = 1 errou = 0 db_igual=1 db_contagem=1110 db_memoria=0001 db_chaves=0001 db_estado=0110
17	Ajustar as chaves em 0100 e acionar clock 3x	chaves = 0100 clock (3x)	(passa pelos estados registra, comparação e proximo) pronto= 1 acertou = 1 errou = 0 db_igual=1

			db_contagem=1111 db_memoria=0100 db_chaves=0100 db_estado=1111



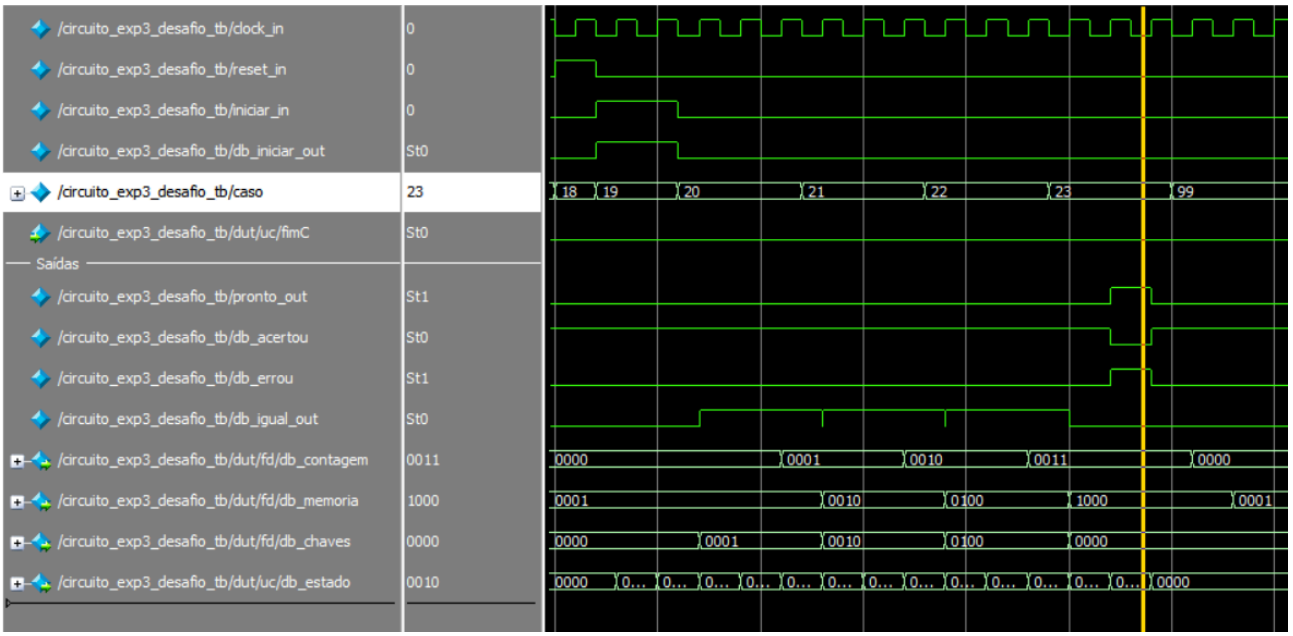
6.3.2 Cenário de Teste 2 – Teste do caso de erro

Nesse teste, na 4a rodada, a saída da chave não corresponde à saída da memória (errou = 1).

Tabela 7 – Descrição e Resultados do Cenário de Teste 2 para o Desafio

#	Operação	Sinais de Controle	Resultado esperado
c.i	Condições iniciais	clock=0 reset=0 iniciar=0 chaves=0000	pronto=0 acertou = 1 errou = 0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0000
1	“Resetar” circuito e observar a saída da memória	reset=1 clock	pronto=0 acertou = 1 errou = 0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0000
2	Acionar sinal de clock 2 vezes com iniciar=1	reset=0 iniciar=1 clock (2x)	(permanece no estado inicial) pronto=0 acertou = 1

			errou = 0 db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0000 db_estado=0001
3	Ajustar chaves para 0001, ativar iniciar=0 e acionar clock 3x	chaves=0001 iniciar=1 clock (3x)	(muda para estado preparação) pronto=0 acertou = 1 errou = 0 db_igual=1 db_contagem=0001 db_memoria=0001 db_chaves=0001 db_estado=0100
4	Mantém chaves em 0010 e acionar clock 3x	chaves=0010 iniciar=0 clock (3x)	(muda para estado registra) pronto=0 acertou = 1 errou = 0 db_igual=1 db_contagem=0010 db_memoria=0010 db_chaves=0010 db_estado=0100
5	Mantém chaves em 0100 e acionar clock 3x	chaves=0100 iniciar=0 clock (3x)	(muda para estado comparação) pronto=0 acertou = 1 errou = 0 db_igual= 1 db_contagem=0011 db_memoria=0100 db_chaves=0100 db_estado=0100
6	Mantém chaves em 0000 e acionar clock 3x	chaves=0000 clock (3x)	(erro) pronto=1 acertou = 0 errou = 1 db_igual= 0 db_contagem=0011 db_memoria=1000 db_chaves=0000 db_estado=0010



7 Conclusões

No planejamento, fizemos a simulação no Modelsim, e projetamos a tabela para teste durante o laboratório. Porém tivemos dificuldades em fazer a pinagem pois havia um bug onde alguns componentes estavam faltando dentro do Quartus. Também para melhor entendimento do circuito desenvolvemos ele no DIgital também para testes, contudo acabamos cometendo um erro nos sensores input impedindo o bom funcionamento do circuito do DIgital na fase do Planejamento.

Já para a experiência prática conseguimos vir com o Digital arrumado, e durante a experiência conseguimos com ajuda dos professores e monitores arrumar nossa pinagem. Depois disso conseguimos sintetizar o circuito na placa FPGA sem grandes problemas, os testes realizados deram de acordo com o previsto pela nossa tabela de testes.

É correto afirmar que nosso circuito funcionou como previsto dentro da experiência e compriu todos os objetivos pré estabelecidos. O desafio foi mais complicado e não conseguimos durante a aula terminar de fazer o Modelsim, porém com ajuda dos monitores nosso código foi verificado, e a sintese na FPGA ficou correta. Durante a tarde conseguimos terminar o Modelsim.