Desenvolvimento de Projeto de Circuitos Digitais em FPGA

Versão 2023

OBJETIVOS

Após a conclusão desta experiência, os seguintes tópicos devem ser conhecidos pelos alunos:

- *Interface de circuitos digitais com elementos externos de entrada de dados;*
- *Projeto de circuitos digitais com sinal periódico como clock;*
- *Síntese em uma placa FPGA, usando Quartus Prime.*

RESUMO

Esta experiência tem como objetivo o estudo de aspectos de projeto de circuitos digitais usando um sinal periódico como entrada de clock e a interface de elementos externos como entrada de dados. O circuito da experiência é baseado no projeto desenvolvido na experiência anterior.

1. PARTE EXPERIMENTAL

A parte experimental desta experiência visa desenvolver um circuito digital que opera com um sinal periódico como *clock* com frequência fixa (p.ex. 1 KHz ou 50 MHz) e processa a entrada de dados de jogadas acionadas por chaves.

1.1. Atividade 1 – Adaptações do Projeto

Nesta primeira atividade, será desenvolvido o projeto lógico da adaptação do circuito da experiência anterior para uso de um sinal periódico como clock e a interface externa de dados de entrada com uso de chaves.

a) Considere o sistema digital com a interface externa de sinais da Figura 1 com funcionamento de acordo com a seguinte descrição. As alterações introduzidas estão destacadas em azul.

> "O circuito do sistema digital sequencial inclui um conjunto de 16 dados de 4 bits que é armazenado em uma memória interna, cujos endereços são percorridos por meio de um contador interno. Depois do acionamento do sinal `reset`, o circuito deve aguardar o início de sua operação até o acionamento do sinal de entrada `iniciar`.
>
> Depois de iniciar seu funcionamento, o circuito deve aguardar o acionamento de uma das chaves de entrada. A ocorrência desse acionamento deve ser indicada pela saída de depuração `db_temjogada`. O dado das chaves (jogada realizada) deve ser armazenado pelo circuito e apresentado nos `leds` de saída e também na saída de depuração `db_jogadafeita`.
>
> Em seguida, deve-se comparar o conteúdo armazenado da entrada de jogada com o respectivo dado da memória e deve-se indicar, na saída de depuração `db_igual`, o resultado da comparação. Em seguida, o contador interno deve ser incrementado para posicionar o acesso à memória interna para o próximo dado. As saídas de depuração `db_contagem` e `db_memoria` indicam, respectivamente, o endereço e o dado armazenado pela memória, ao passo que a saída de depuração `db_estado`, por sua vez, deve indicar o código do estado vigente da Unidade de Controle em determinado instante do funcionamento do sistema digital. Essas quatro saídas (`db_jogadafeita`, `db_contagem`, `db_memoria` e `db_estado`) devem ser projetadas para serem exibidas em *displays* de sete segmentos.
>
> O ciclo de espera pela jogada, armazenamento das chaves, comparação e reposicionamento da memória deve prosseguir enquanto o jogador acertar o dado armazenado na memória **e** até que todos os 16 dados da memória sejam verificados. Se o jogador acertar todos os dados, o sinal de saída `acertou` deve ser ativado. Se o jogador errar um dado, o ciclo deve ser imediatamente interrompido, e o sinal de saída `errou` deve ser ativado. Ao final da operação, o sinal de saída `pronto` também deve ser ativado. Depois disso, o circuito deve aguardar o próximo acionamento do sinal `iniciar`. Essas três saídas (`acertou`, `errou` e `pronto`), quando ativadas, devem permanecer ativadas até o reinício da operação do circuito."

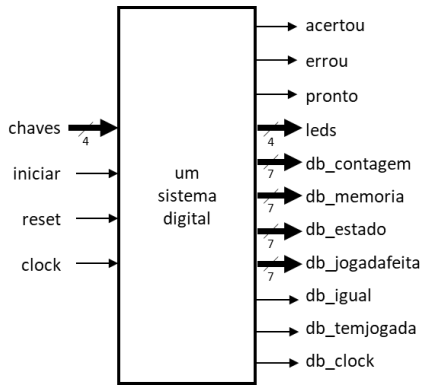

Figura 1: Diagrama de Blocos da Interface Externa de Sinais do Circuito para a Atividade 1

O funcionamento do sistema digital da Figura 1 é descrito pelo **pseudocódigo** da Figura 2. As modificações introduzidas sobre o circuito da Atividade 3 (Desafio) da Experiência 3 estão sinalizadas em azul.

```
Algoritmo: sistema digital simples modificado
    entradas: iniciar, chaves
    saídas: acertou, errou, pronto, leds
    depuração: contagem, memória, estado, jogadafeita, igual, temjogada
1.  {
2.      while (verdadeiro) {
3.          espera acionamento do sinal iniciar
4.          inicia circuito com condições iniciais
5.          while (acertou dado E não atingiu o último dado) {

6.              espera acionamento das chaves (jogada)

7.              registra entrada da jogada
8.              compara jogada realizada com dados armazenados
9.              incrementa contador interno
10.         }
11.         ativa acertou se acertou todos os dados da memória
12.         ativa errou se errou um dado
13.         ativa saída pronto (até reiniciar)
14.     }
15. }
```

Figura 2: Pseudocódigo do Sistema Digital da Atividade 1

Adote a seguinte definição para a entidade principal do circuito (Figura 3). Se necessário, o grupo poderá adicionar sinais de depuração. Documente as **decisões de projeto do grupo** no Planejamento.

```
entity circuito_exp4 is
    port (
        clock          : in  std_logic;
        reset          : in  std_logic;
        iniciar        : in  std_logic;
        chaves         : in  std_logic_vector (3 downto 0);
        pronto         : out std_logic;
        acertou        : out std_logic;
        errou          : out std_logic;
        leds           : out std_logic_vector (3 downto 0);
        db_igual       : out std_logic;
        db_contagem    : out std_logic_vector (6 downto 0);
        db_memoria     : out std_logic_vector (6 downto 0);
        db_estado      : out std_logic_vector (6 downto 0);
        db_jogadafeita : out std_logic_vector (6 downto 0);
        db_clock       : out std_logic;
        db_tem_jogada  : out std_logic
    );
end entity;
```

Figura 3: Entidade VHDL do Circuito da Atividade 1

1.1.1. Dicas de Projeto para o Fluxo de Dados

Repare que a definição da entidade `circuito_exp4` introduz dois sinais adicionais de saída não presentes na especificação funcional do circuito: `db_clock` e `db_jogadafeita`. Estes sinais foram incluídos para serem usados durante a depuração do circuito na placa FPGA para verificar o funcionamento dos sinais de entrada `clock` e das chaves e, assim, auxiliar a realização dos testes durante a verificação de funcionamento do circuito na bancada. O sinal `db_clock` deve ser uma cópia da entrada `clock`, e a saída `db_jogadafeita` deve indicar, em um *display* de sete segmentos, o valor armazenado nas chaves de entrada do circuito em dado instante de tempo.

Na Figura 4, é apresentado um diagrama de blocos do **circuito para detecção de jogadas**. Note a utilização de um componente identificado como `edge_detector`. Esse componente é um circuito que faz parte do **fluxo de dados** e realiza a detecção da ativação do sinal de entrada (que pode ter vários períodos de *clock* de duração – ver próximo parágrafo para mais detalhes sobre isso) e gera na saída um pulso com um período de *clock*. Essa saída deve ser usada pela unidade de controle para verificar a ocorrência de uma jogada.

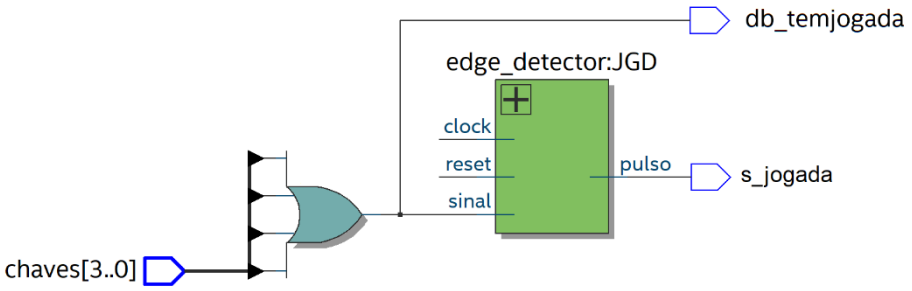

Figura 4: Diagrama de Blocos do Circuito para Detecção de Acionamento das Chaves

Isso é necessário para que o circuito digital possa **gerenciar sinais externos** (normalmente acionados por um ser humano). A explicação para isso é a de que o acionamento de um botão eletromecânico tem duração típica de centenas de milissegundos, e esse período corresponde a vários períodos de sinais de *clock* tipicamente utilizados nos sistemas digitais (por exemplo, centenas de períodos de *clock* para um circuito operando a 1 KHz ou centenas de milhares de períodos para um circuito operando a 1 MHz). Assim, quando um circuito opera com um sinal de *clock* periódico, este deve estar preparado para detectar e tratar alterações de sinais de entrada com "longa duração".

Na Figura 5, é apresentado um trecho da simulação do circuito para detecção do acionamento das entradas usando o software de simulação **ModelSim** que acompanha o Intel Quartus Prime. Nela podemos observar que uma das chaves permanece ativada por 15 períodos de *clock*, mas o sinal `s_jogada` é gerado na próxima borda de subida do *clock* e tem apenas 1 período de duração.

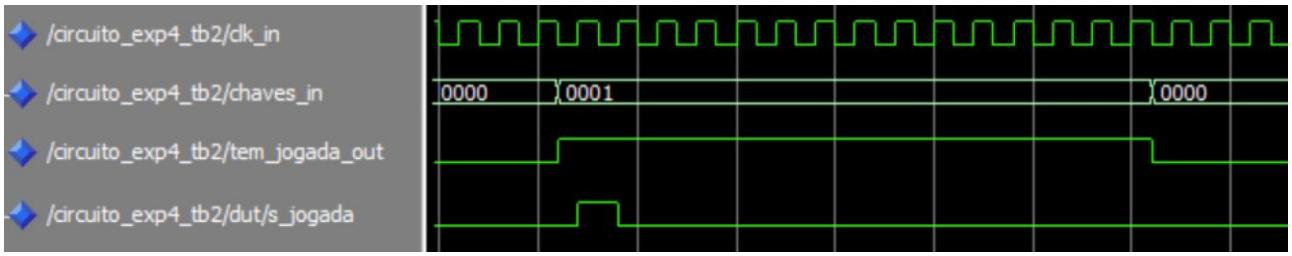

Figura 5: Trecho de Simulação do Circuito para Detecção de Acionamento das Chaves com Software ModelSim

1.1.2. Dicas de Projeto para a Unidade de Controle

Da mesma forma, a **unidade de controle** também precisa ser ajustada para estas condições (*clock* periódico e entradas externas). Isso é feito pelo acréscimo de estados de espera pelo acionamento das entradas, aguardando até que uma das chaves seja acionada. O diagrama de estados de alto nível que trata essas condições é apresentado na Figura 6.

Convém ressaltar três estados em particular, marcados na Figura 6 em amarelo, azul e vermelho:

- Estado "`espera jogada`": Serve para aguardar o acionamento de uma das chaves de entrada. Enquanto o circuito não detectar o acionamento de uma das entradas, a unidade de controle permanece neste estado e não realiza a comparação das chaves de entrada com o dado da memória.
- Estados de finalização do jogo ("`final com acertos`" e "`final com erro`"): Correspondem aos estados de acionamento das saídas `acertou`, `errou` e `pronto`. Para um sinal de *clock* de 1 KHz, caso estas saídas fossem ativadas por apenas 1 período de *clock*, elas ficariam ativadas por somente 1 ms, imperceptível ao olho humano. Desta forma, optou-se por fazer com que estas saídas permaneçam ativas até o reinício de operação.

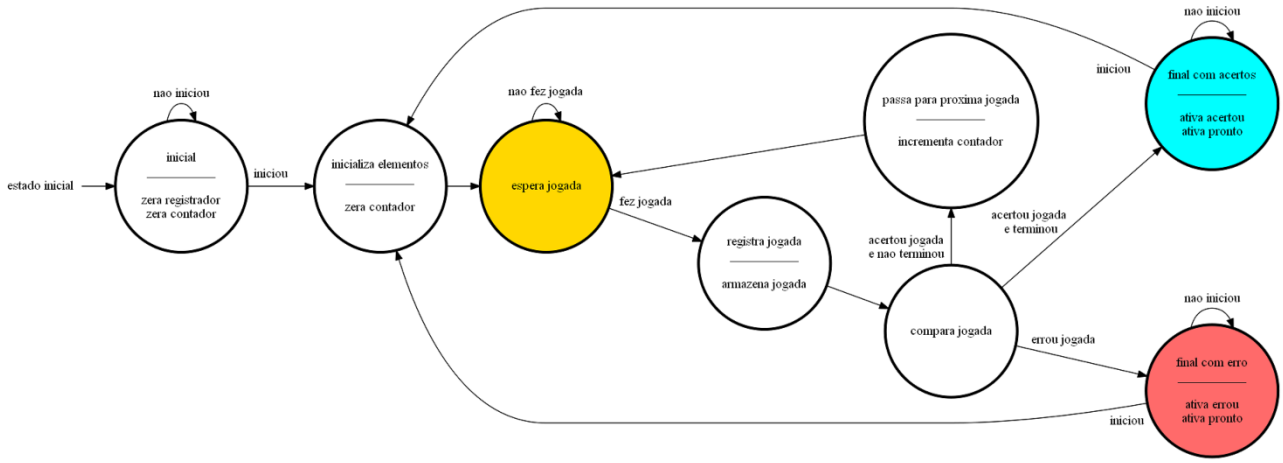

Figura 6: Diagrama de Estados de Alto Nível Adaptado para a Experiência 4

Adote a seguinte definição para a entidade `unidade_controle` (Figura 7).

```
entity unidade_controle is
    port (
        clock     : in  std_logic;
        reset     : in  std_logic;
        iniciar   : in  std_logic;
        fim       : in  std_logic;
        jogada    : in  std_logic;
        igual     : in  std_logic;
        zeraC     : out std_logic;
        contaC    : out std_logic;
        zeraR     : out std_logic;
        registraR : out std_logic;
        acertou   : out std_logic;
        errou     : out std_logic;
        pronto    : out std_logic;
        db_estado : out std_logic_vector(3 downto 0)
    );
end entity;
```

Figura 7: Entidade VHDL da Unidade de Controle

1.1.3. Dicas de Projeto para o Sistema Digital

Internamente, o circuito deve ser composto por dois componentes: um fluxo de dados e uma unidade de controle, conforme diagrama apresentado na Figura 8. Os codificadores para *displays* de 7 segmentos servem para efetuar a interface dos sinais de saída do circuito para os elementos da placa FPGA.

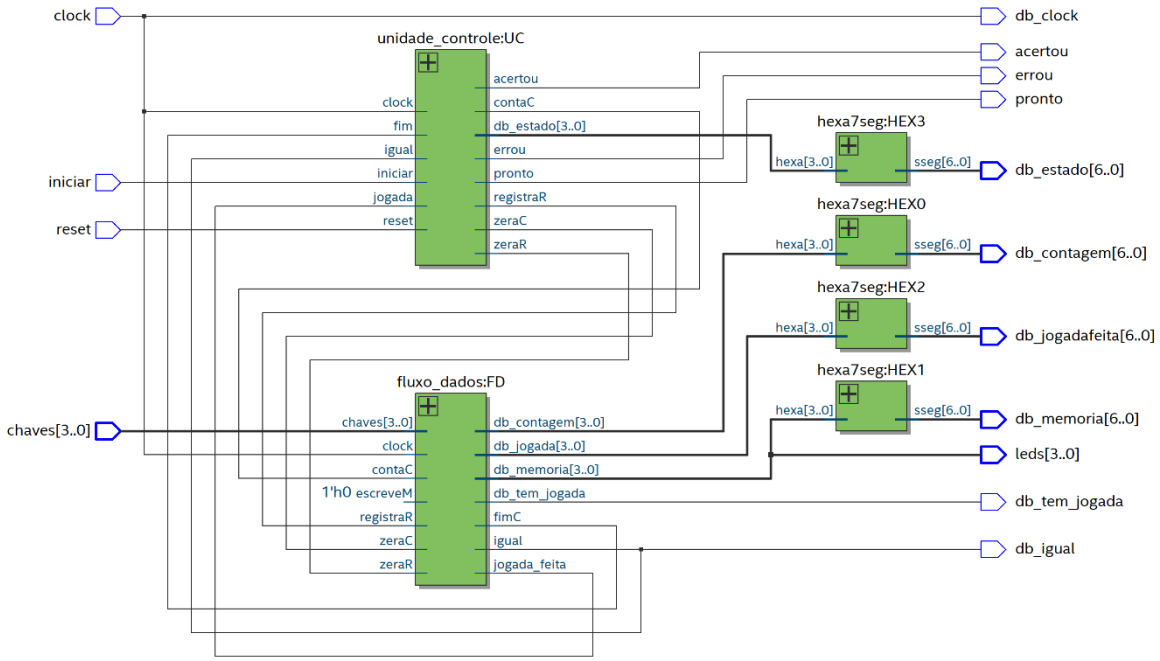

Figura 8: Diagrama de Blocos da Estrutura Interna do Circuito para a Atividade 1

O Fluxo de Dados do circuito deve ser elaborado a partir do projeto da Experiência 3, adicionando-se o circuito para detecção de jogadas. Na Figura 9, é apresentado um diagrama de blocos com a estrutura interna do componente do Fluxo de Dados.

NOTA: repare que figura não mostra a origem do sinal de entrada `reset` do componente `edge_detector`. O grupo deve escolher a melhor solução tendo como base o tratamento dos sinais das `chaves` conforme especificação do circuito da experiência.

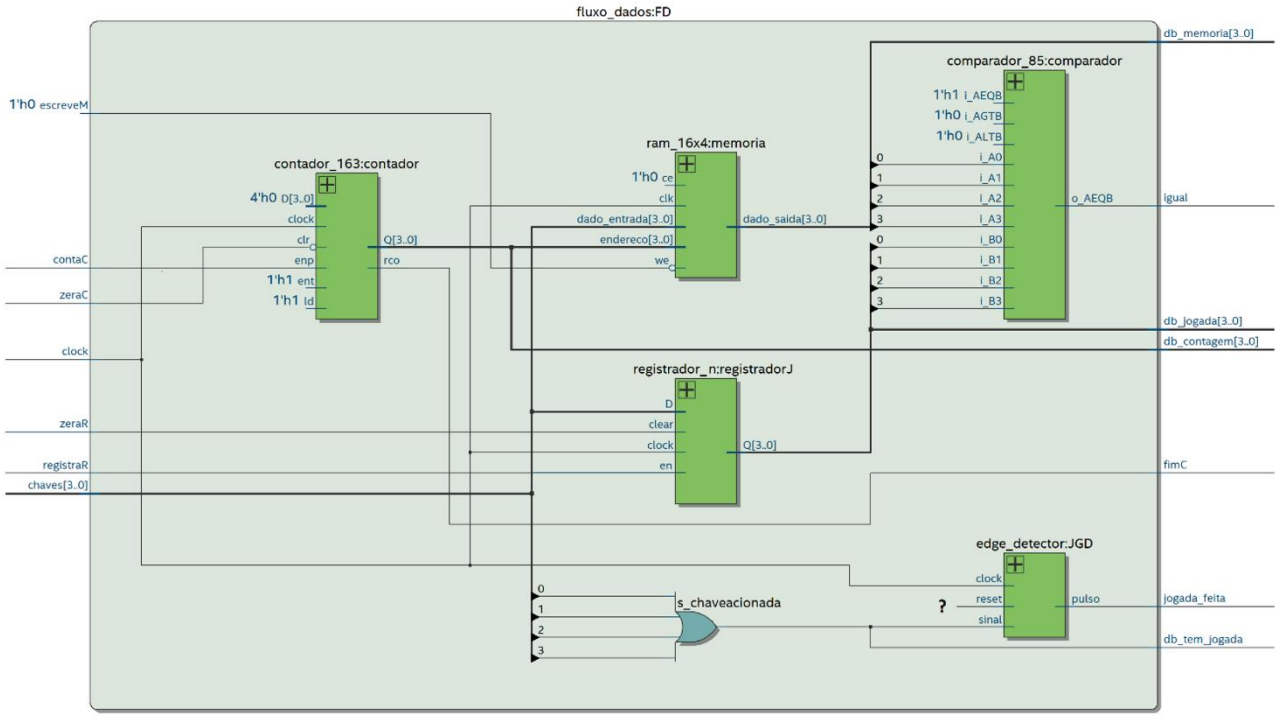

Figura 9: Diagrama de Blocos da Estrutura Interna do Fluxo de Dados

Adote a seguinte definição para a entidade `fluxo_dados` (Figura 10).

```
entity fluxo_dados is
    port (
        clock         : in  std_logic;
        zeraC         : in  std_logic;
        contaC        : in  std_logic;
        escreveM      : in  std_logic;
        zeraR         : in  std_logic;
        registraR     : in  std_logic;
        chaves        : in  std_logic_vector (3 downto 0);
        igual         : out std_logic;
        fimC          : out std_logic;
        jogada_feita  : out std_logic;
        db_tem_jogada : out std_logic;
        db_contagem   : out std_logic_vector (3 downto 0);
        db_memoria    : out std_logic_vector (3 downto 0);
        db_jogada     : out std_logic_vector (3 downto 0)
    );
end entity;
```

Figura 10: Entidade VHDL do Fluxo de Dados

A saída `jogada_feita` é um sinal de condição gerado pelo fluxo de dados para sinalizar à unidade de controle a ocorrência de uma jogada através do acionamento de uma das chaves de entrada.

1.1.4. Verificação funcional do Projeto

b) Considere o Plano de Testes da Tabela 1 para o circuito da atividade 1, composto por dois cenários de teste. Complete a tabela com os testes faltantes, os sinais de entrada do circuito necessários para executar cada operação e os resultados esperados. Esses testes devem ser executados sequencialmente.

Tabela 1: Modelo de Plano de Testes para o Circuito da Atividade 1

Cenário #1 – Acerto das 16 jogadas				
#	**Operação**	**Sinais de entrada**	**Resultado esperado**	**Resultado observado**
c.i.	Condições Iniciais			
1	"Resetar" circuito	acionar reset		
2	Aguardar alguns segundos			
3	Acionar sinal iniciar	acionar iniciar		
4	Acionar primeira entrada (jogada 1)	acionar chave(0)		
5	Acionar segunda entrada (jogada 2)	acionar chave(1)		
...	...	...	...	...
19	Acionar última entrada (jogada 16)	acionar chave(2)	saídas pronto e acertou ativadas	
Cenário #2 – Acerto das 3 primeiras jogadas e erro na 4ª jogada				
#	**Operação**	**Sinais de entrada**	**Resultado esperado**	**Resultado observado**
c.i.	Condições Iniciais			
1	"Resetar" circuito	acionar reset		
2	Aguardar alguns segundos			
3	Acionar sinal iniciar	acionar iniciar		
4	Acionar primeira entrada (jogada 1)	acionar chave(0)		
5	Acionar segunda entrada (jogada 2)	acionar chave(1)		
6	Acionar terceira entrada (jogada 3)	acionar chave(2)		
7	Acionar quarta entrada errada (jogada 4)	acionar chave(0)	saídas pronto e errou ativadas	

Execute simulações do circuito com o ModelSim usando o Plano de Testes da Tabela 1 para verificar o funcionamento do seu circuito. A Figura 11 ilustra uma possível saída obtida como saída da ferramenta de simulação. Faça anotações nas formas de onda para ilustrar a execução dos testes.

DICA: um arquivo de *testbench* (bancada de testes) modelo é fornecido no e-Disciplinas como referência para o desenvolvimento desta tarefa.

Ao final da verificação, submeter os arquivos fonte VHDL usados com o ModelSim em um arquivo ZIP (`exp4_TxByy_modelsim.zip`). Estes arquivos devem ser usados como base para a simulação do projeto do Desafio (Atividade 3).

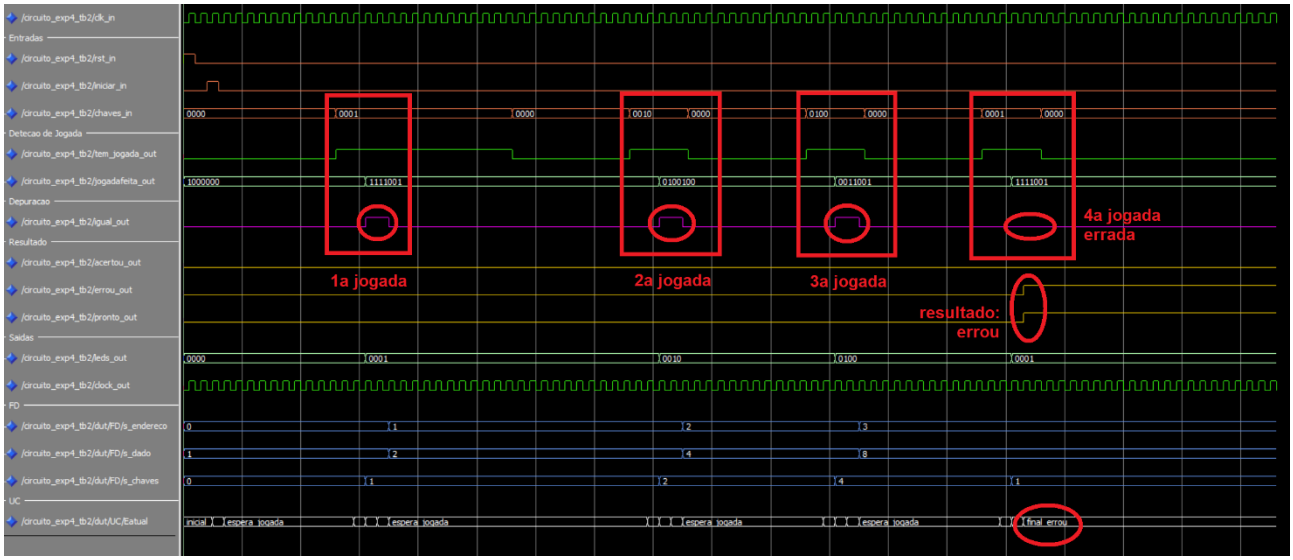

Figura 11: Exemplo de Saída (anotada) da Ferramenta de Simulação ModelSim

c) Crie um projeto no Intel Quartus Prime referente ao circuito da atividade 1.

d) Documente o projeto elaborado no Planejamento. Gere as saídas da ferramenta **RTL Viewer** para o sistema digital e o fluxo de dados e analise o projeto. Ao final, submeta o arquivo QAR (`exp4_TxByy_ativ1.qar`) junto com o Planejamento.

1.2. Atividade 2 – Síntese e Teste do Circuito na Placa FPGA

Esta atividade tem como objetivo a implementação e a síntese do circuito da atividade 1 na placa FPGA DE0-CV. Em seguida, este circuito deve ser estudado e seu comportamento deve ser validado conforme projetado.

e) No Laboratório Digital, sintetize o projeto do circuito para a FPGA Cyclone V 5CEBA4F23C7N da placa DE0-CV. Para isso, adote a designação de pinos da placa DE0-CV da Tabela 2 e complete-a com os dados faltantes no Planejamento.

Programe o projeto sintetizado na placa FPGA DE0-CV

Tabela 2: Designação de Pinos para a Atividade 2

Sinal	Pino na Placa DE0-CV	Pino na FPGA	Analog Discovery
CLOCK	GPIO_0_D0		StaticIO – LED – DIO0 ou Patterns – Clock – 1 kHz
RESET	GPIO_0_D1		StaticIO – Button 0/1 – DIO1
INICIAR	chave SW0		-
CHAVES(0)	GPIO_0_D11		StaticIO – Button 0/1 – DIO2
CHAVES(1)	GPIO_0_D13		StaticIO – Button 0/1 – DIO3
CHAVES(2)	GPIO_0_D15		StaticIO – Button 0/1 – DIO4
CHAVES(3)	GPIO_0_D17		StaticIO – Button 0/1 – DIO5
LEDS(0)	Led LEDR0		-
LEDS(1)	Led LEDR1		-
LEDS(2)	Led LEDR2		-
LEDS(3)	Led LEDR3		-
PRONTO	Led LEDR4		-
ACERTOU	Led LEDR5		-
ERROU	Led LEDR6		-
DB_IGUAL	Led LEDR7		-
DB_CLOCK	Led LEDR8		-
DB_TEM_JOGADA	Led LEDR9		-
DB_CONTAGEM	Display HEX0		-
DB_MEMORIA	Display HEX1		-
DB_JOGADAFEITA	Display HEX2		-
DB_ESTADO	Display HEX5		-

f) Execute o acionamento das sequências de sinais de entrada dos cenários do Plano de Testes do circuito. Anote os resultados experimentais obtidos para cada cenário.

Analise os resultados obtidos nos testes e elabore um resumo do funcionamento do circuito estudado.

Submeta o arquivo QAR final desta atividade (`exp4_TxByy_ativ2_final.qar`) junto com o Relatório.

DICA: No Planejamento, prepare a realização desta Atividade e elabore um roteiro de sua execução. Detalhe como os testes de funcionamento do circuito em bancada deverão ser executados no Laboratório Digital e acrescente uma análise com algumas possíveis causas de erros que podem ocorrer durante a execução dos testes.

1.3. Atividade 3 – Desafio: Modificação do Sistema Digital

Esta atividade tem como objetivo a familiarização com a composição de um fluxo de dados e uma unidade de controle para projetar um sistema digital. O sistema digital deve ser modificado a partir do projeto da atividade 1.

g) A especificação de uma modificação ao circuito da experiência será fornecida pelo professor. Projete a modificação do sistema digital e documente-a no relatório.

h) Elabore um Plano de Testes para verificar o funcionamento do circuito. Considere ao menos dois cenários de teste para o circuito modificado. Para cada cenário vislumbrado pelo grupo, gere uma tabela com o plano de testes seguindo a estrutura da Tabela 3 e complete-a com os sinais de controle necessários para executar cada operação e os resultados esperados.

Nota Importante: Cada cenário de teste deve ser executado separadamente. Deve haver uma tabela com o plano de testes específico para cada cenário exercitado.

Tabela 3: Modelo de Tabela para um Cenário do Plano de Testes da Atividade 3

Cenário #n – Descrição do Cenário				
#	**Operação**	**Sinais de entrada**	**Resultado esperado**	**Resultado observado**
c.i.	Condições Iniciais			
1				
2				
3				
...				
n				

i) Realize algumas **simulações do projeto** com o ModelSim executando os cenários do Plano de Testes do item anterior. Inclua as formas de onda obtidas na documentação da experiência.
DICA: altere os arquivos de *testbench* conforme os cenários elaborados para esta atividade.

j) Sintetize o projeto do circuito no FPGA Cyclone V 5CEBA4F23C7N. Para isto, adote a mesma designação de pinos da Atividade 2 e complete-a com os dados faltantes. **Sinais adicionais de depuração** podem ser incluídos conforme a necessidade (consulte a designação de pinos disponível no e-Disciplinas e documente-a no relatório).

k) Programe o projeto na placa DE0-CV e execute o acionamento da sequência de sinais de entrada conforme Plano de Testes elaborado pelo grupo. Anote os resultados experimentais obtidos.

l) Submeta o arquivo QAR final desta atividade (`exp4_TxByy_desafio.qar`) junto com o Relatório.

1.4. Atividade 4 – Perguntas para Aprofundamento (opcional)

Esta atividade adicional tem como objetivo aprofundar o aprendizado da experiência. Sua realização é opcional, mas recomendada.

m) Após a conclusão das atividades no Laboratório Digital, responda as questões abaixo no Relatório:

1. O que acontece no circuito da experiência caso fosse possível (fisicamente) acionar duas chaves de entrada simultaneamente? Explique como o grupo elaborou a resposta desta questão. Sua resposta pode ser verificada de que forma?
2. Deseja-se acrescentar a função de seleção de nível de dificuldade para o circuito da experiência. Uma nova entrada nivel seleciona o número de jogadas certas que o jogador deve acertar. Se nivel=0, o jogador deve acertar apenas 8 jogadas para ganhar, e se nivel=1, deve acertar as 16 jogadas armazenadas na memória. Explique as alterações necessárias no fluxo de dados e na unidade de controle.

2. BIBLIOGRAFIA

[1] ALMEIDA, F.V. de; SATO, L.M.; MIDORIKAWA, E.T. **Tutorial para criação de circuitos digitais em VHDL no Quartus Prime 16.1**. Apostila de Laboratório Digital. Departamento de Engenharia de Computação e Sistemas Digitais, Escola Politécnica da USP. Edição de 2017.

[2] ALMEIDA, F.V. de; SATO, L.M.; MIDORIKAWA, E.T. **Tutorial para criação de circuitos digitais hierárquicos em VHDL no Quartus Prime 16.1**. Apostila de Laboratório Digital. Departamento de Engenharia de Computação e Sistemas Digitais, Escola Politécnica da USP. Edição de 2017.

[3] ALTERA / Intel. **DE0-CV User Manual**. 2015.

[4] ALTERA / Intel. **Quartus Prime Introduction Using VHDL Designs.** 2016.

[5] ALTERA / Intel. **Quartus Prime Introduction to Simulation of VHDL Designs.** 2016.

[6] D'AMORE, R. **VHDL - descrição e síntese de circuitos digitais**. 2ª edição, LTC, 2012.

[7] WAKERLY, John F. **Digital Design Principles & Practices**. 4th edition, Prentice Hall, 2006.

3. EQUIPAMENTOS NECESSÁRIOS

- 1 computador pessoal com os softwares Intel Quartus Prime e ModelSim.
- 1 placa de desenvolvimento FPGA DE0-CV da Altera com o dispositivo Cyclone V 5CEBA4F23C7N.
- 1 dispositivo Analog Discovery da Digilent ou equivalente.

Histórico de Revisões

E.T.M. & A.V.S.N / 2021 (versão inicial)
E.T.M. / 2022 (revisão e adaptação)
E.T.M. / 2023 (revisão, reorganização e adaptação para ensino presencial)