Projeto de uma Unidade de Controle

Versão 2023

OBJETIVOS

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

- *Projeto da unidade de controle de um circuito digital;*
- *Projeto hierárquico de circuitos digitais com fluxo de dados e unidade de controle;*
- *Codificação de uma máquina de estados em VHDL;*
- *Síntese em uma placa FPGA, usando Quartus Prime.*

RESUMO

Esta experiência tem como objetivo o estudo do desenvolvimento do projeto da unidade de controle de um circuito digital simples usando VHDL. A estrutura da unidade de controle é desenvolvida a partir de um pseudocódigo. Essa unidade de controle deve ser usada em conjunto com um fluxo de dados baseado no circuito da Experiência 2 para compor um sistema digital completo.

1. PARTE EXPERIMENTAL

A parte experimental desta experiência visa desenvolver a unidade de controle de um circuito digital. O projeto deve ser sintetizado para a placa DE0-CV. Posteriormente, deve-se executar um estudo detalhado de seu funcionamento, através da execução de um plano de testes minucioso. Esta experiência será dividida em algumas atividades, onde o projeto será desenvolvido em algumas etapas, de forma incremental.

1.1. Atividade 1 – Projeto Lógico de um Sistema Digital

Nesta primeira atividade, será desenvolvido o projeto lógico da unidade de controle de um sistema digital simples, tendo como base os resultados dos estudos do fluxo de dados da Experiência 2.

1.1.1. Especificação do Sistema Digital

Inicialmente apresentamos a especificação do sistema digital da experiência.

a) Considere um sistema digital simples com a interface externa de sinais conforme diagrama de blocos da Figura 1. As saídas db_chaves, db_contagem, db_memoria e db_estado devem ser apresentadas em *displays* de 7 segmentos da placa FPGA. O **funcionamento deste sistema digital** deve seguir a descrição subsequente:

> *"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 que é realizado com o acionamento do sinal de entrada `iniciar`. Depois de iniciar seu funcionamento, o circuito deve armazenar o valor das chaves de entrada (sinal `chaves`) e depois comparar o conteúdo armazenado das chaves com o respectivo dado da memória e deve indicar o resultado da comparação na saída de depuração `db_igual`. O conteúdo armazenado das chaves é apresentado na saída de depuração `db_chaves`. Em seguida, o contador interno deve ser incrementado para posicionar o endereçamento da memória para permitir o acesso ao próximo dado da memória. 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[1]. O ciclo de comparação e reposicionamento da memória deve prosseguir até que todos os 16 dados sejam verificados. Ao final da operação, o sinal de saída `pronto` deve ser acionado por um período de clock. Depois disso, o circuito deve voltar para o estado inicial para aguardar a próxima ativação de `iniciar`".*

[1] Para mais detalhes sobre a codificação dos estados da Unidade de Controle, consulte o código fonte em VHDL.

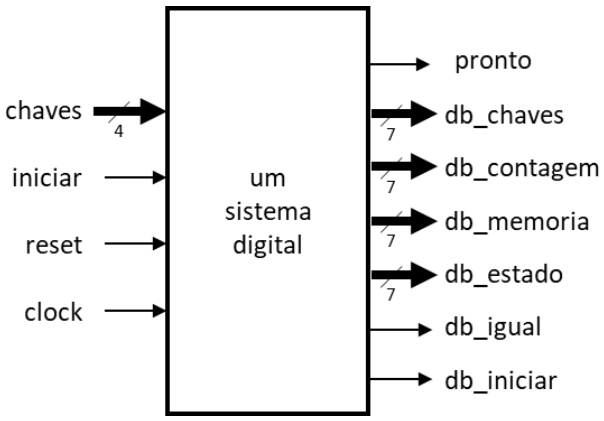

Figura 1: Diagrama de Blocos da Interface Externa de Sinais do Circuito para a Atividade 1

O funcionamento do sistema digital da Figura 1 pode ser descrito pelo **pseudocódigo** da Figura 2.

```
Algoritmo: um sistema digital simples
    entradas: iniciar, chaves
    saídas: pronto
    depuração: chaves, contagem, memória, estado, igual, iniciar
1.  {
2.     while (verdadeiro) {
3.        espera acionamento do sinal iniciar
4.        inicia circuito para condições iniciais
5.        while (não atingiu o último dado) {
6.            compara chaves de entrada com dados armazenados e atualiza saídas
7.            incrementa contador interno
8.        }
9.        ativa saída pronto
10.    }
11. }
```

Figura 2: Pseudocódigo do Sistema Digital da Experiência

Adote a definição para a entidade principal do circuito conforme apresentada na Figura 3. O sinal de depuração `db_iniciar` é usado para verificar o acionamento da entrada `iniciar` do circuito sintetizado para a placa de desenvolvimento com FPGA DE0-CV.

```
entity circuito_exp3 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;
        db_igual    : out std_logic;
        db_iniciar  : out std_logic;
        db_contagem : out std_logic_vector (6 downto 0);
        db_memoria  : out std_logic_vector (6 downto 0);
        db_chaves   : out std_logic_vector (6 downto 0);
        db_estado   : out std_logic_vector (6 downto 0)
    );
end entity;
```

Figura 3: Entidade principal do Sistema Digital da Atividade 1

Internamente, o circuito deve ser composto por dois componentes internos, um fluxo de dados e uma unidade de controle, conforme diagrama (gerado pela ferramenta RTL Viewer do Intel Quartus Prime) apresentado na Figura 4. Os codificadores para *displays* de 7 segmentos servem para interfacear os sinais de saída do circuito para os elementos da placa FPGA.

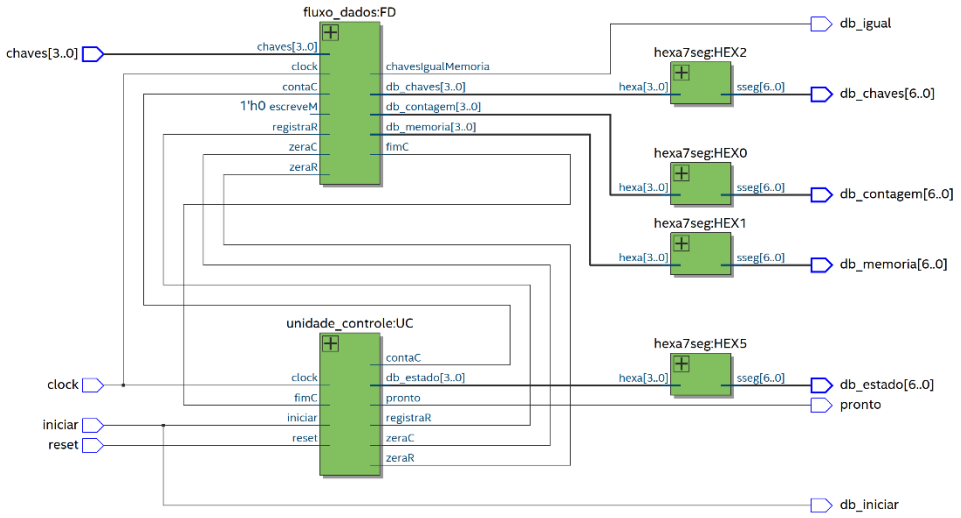

Figura 4: Diagrama de Blocos da Estrutura Interna do Circuito da Experiência

Nos itens seguintes, vamos apresentar tarefas específicas para cada um dos componentes do circuito.

1.1.2. Adaptação do Projeto Lógico do Fluxo de Dados

Em primeiro lugar, iremos adaptar o fluxo de dados estudado na Experiência 2 para incluir um registrador. Depois de um breve estudo do componente VHDL correspondente, o componente da experiência anterior deve ser adaptado.

Projeto Lógico do Fluxo de Dados

b) A Figura 5 apresenta um diagrama de blocos do fluxo de dados da experiência. Compare-o com o fluxo de dados da experiência anterior e analise as diferenças.

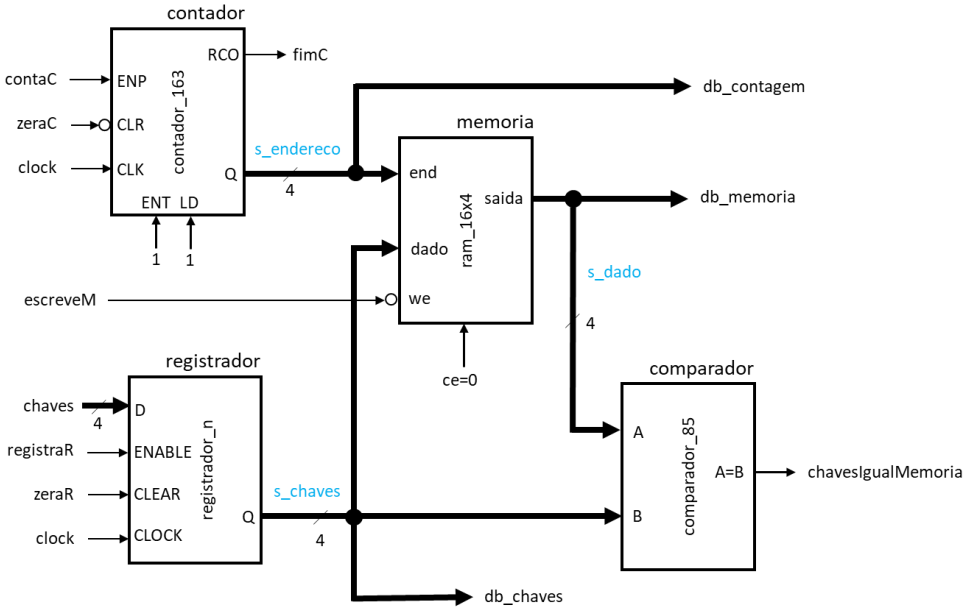

Figura 5: Diagrama de Blocos da Estrutura Interna do Fluxo de Dados.

Estude o funcionamento do componente `registrador_n` (fornecido no e-Disciplinas) e seu uso em um projeto.

O código do fluxo de dados deve ser baseado no circuito projetado na experiência anterior[2]. Revise o código, modifique as identificações da entidade e dos sinais, atualize-o para acrescentar o registrador e armazene-o no arquivo `fluxo_dados.vhd` com uma adaptação da entidade VHDL conforme definição subsequente (Figura 6).

```
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);
        chavesIgualMemoria  : out std_logic;
        fimC                : out std_logic;
        db_contagem         : out std_logic_vector (3 downto 0);
        db_memoria          : out std_logic_vector (3 downto 0);
        db_chaves           : out std_logic_vector (3 downto 0)
    );
end entity fluxo_dados;
```

Figura 6: Entidade VHDL do componente fluxo_dados

Verificação do fluxo de dados e simulação com ModelSim

c) O circuito do fluxo de dados pode ser verificado com o Plano de Testes da Tabela 1. Estude os testes e descreva no Planejamento como este plano aciona todos os componentes e as ligações entre eles para verificar o correto funcionamento do circuito. Na tabela, alguns resultados relevantes estão marcados em azul.

Simule o circuito do fluxo de dados usando o arquivo de *testbench* fornecido com o ModelSim. Anexe as formas de onda obtidas e **anote a figura** com algumas análises dos resultados dos testes. Por exemplo, a Figura 7 apresenta um exemplo com 2 anotações na saída de formas de onda do ModelSim.

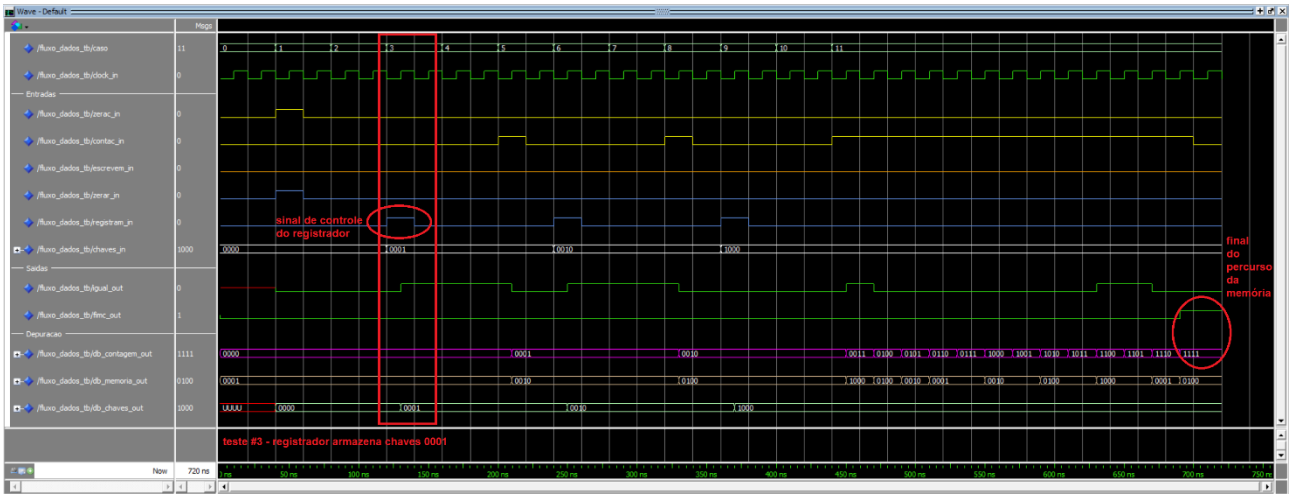

Figura 7: Exemplos de anotações na saída de formas de onda do ModelSim

[2] Um novo arquivo MIF (`ram_conteudo_jogadas.mif`) é disponibilizado para esta experiência no e-Disciplinas. Altere o código fonte do componente `ram_16x4` para usar este arquivo para síntese com o Intel Quartus Prime.

Tabela 1: Plano de Testes para o Circuito do Fluxo de Dados

#	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

1.1.3. Estudo do Projeto Lógico de uma Unidade de Controle

Agora vamos introduzir como uma unidade de controle pode ser projetada a partir do pseudocódigo do sistema digital.

d) O projeto da **unidade de controle** pode ser iniciado pelo pseudocódigo do circuito. Uma análise do algoritmo de funcionamento do circuito apresentado no item 1.1.a resulta na **máquina de estados de alto nível** da Figura 8. Observe que no diagrama temos apenas comandos e condições escritos em linguagem natural.

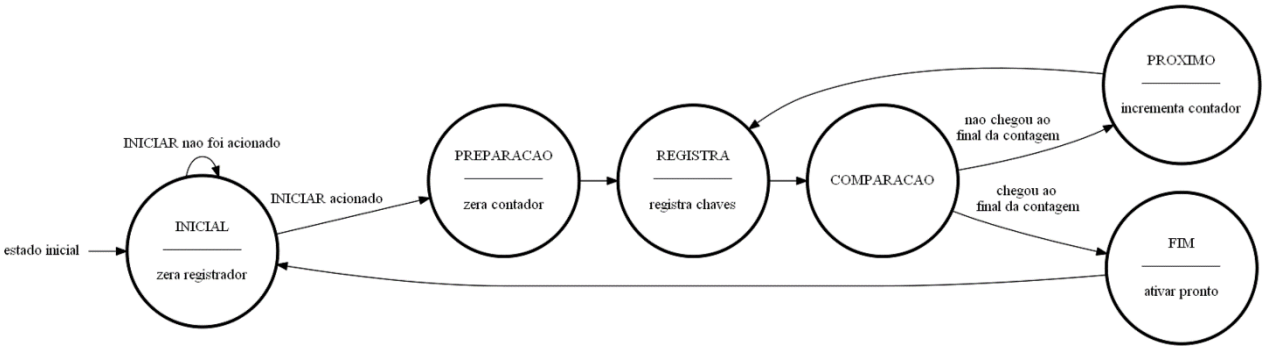

Figura 8: Diagrama de Transição de Alto Nível do Circuito para a Atividade 1

Repare que algumas operações internas do circuito (p.ex., comparação entre dados das chaves e da memória) não são apresentadas nesta máquina de estados por serem de responsabilidade do fluxo de dados. As operações apresentadas relacionam-se apenas aos **sinais de controle** do circuito, cuja geração automática fica sob responsabilidade da Unidade de Controle.

Os comandos de alto nível da máquina de estados devem ser traduzidos para sinais de controle[3] dos elementos do fluxo de dados do circuito digital. O diagrama de transição de estados da unidade de controle resultante é apresentado na Figura 9.

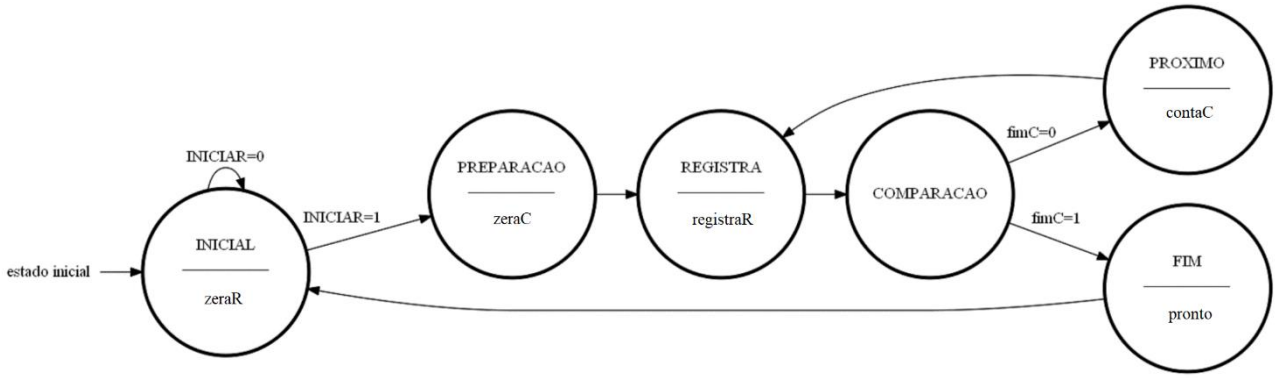

Figura 9: Diagrama de Transição de Estados da Unidade de Controle do Circuito para a Atividade 1

Repare também que o diagrama de transição de estados da Figura 9 apresenta os sinais de controle do fluxo de dados associados ao respectivo comando de alto nível do diagrama da Figura 8. Em seguida, note que, nas transições de estados, cada sinal avaliado corresponde ou a um sinal gerado pelo fluxo de dados[4] (cada sinal desses é, também, conhecido como **sinal de condição**), ou a um sinal de entrada de controle do circuito (p.ex., *iniciar*).

[3] Os sinais de controle dos elementos do fluxo de dados podem ser verificados na Figura 5.

[4] A Figura 5 mostra o diagrama de blocos do fluxo de dados em que o sinal de condição `fimC` é gerado pelo contador interno.

e) O componente VHDL correspondente à unidade de controle segue a definição subsequente (Figura 10). Seu código-fonte (arquivo `unidade_controle.vhd`) está disponível no material associado à experiência no e-Disciplinas.

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

Figura 10: Entidade VHDL do componente unidade_controle

Estude o código VHDL fornecido e analise a tradução dos elementos da máquina de estado da Unidade de Controle (memória de estado, lógica de próximo estado e lógica de saída) para comandos VHDL. Verifique se há alguma inconsistência nesta tradução. Anote suas conclusões no Planejamento.

1.1.4. Projeto do Sistema Digital

Em seguida, vamos concluir o projeto do circuito da experiência, elaborando o componente principal.

f) Elabore agora o projeto do componente principal do circuito da experiência, usando a definição da entidade apresentada na Figura 3.

g) Considere o **Plano de Testes** da Tabela 2 para o circuito da Experiência. Estude os testes propostos e complete a tabela com os sinais de controle necessários para executar cada operação e os resultados esperados. Considere que esses testes devem ser executados sequencialmente.

Execute uma simulação do circuito com o **ModelSim** usando o arquivo de *testbench* fornecido. Verifique os resultados previstos no Plano de Testes com as saídas das formas de onda do simulador. Elabore uma análise desta verificação no Planejamento. Faça anotações nas formas de onda. A Figura 11 ilustra uma parte de uma saída possível das formas de onda geradas pela simulação.

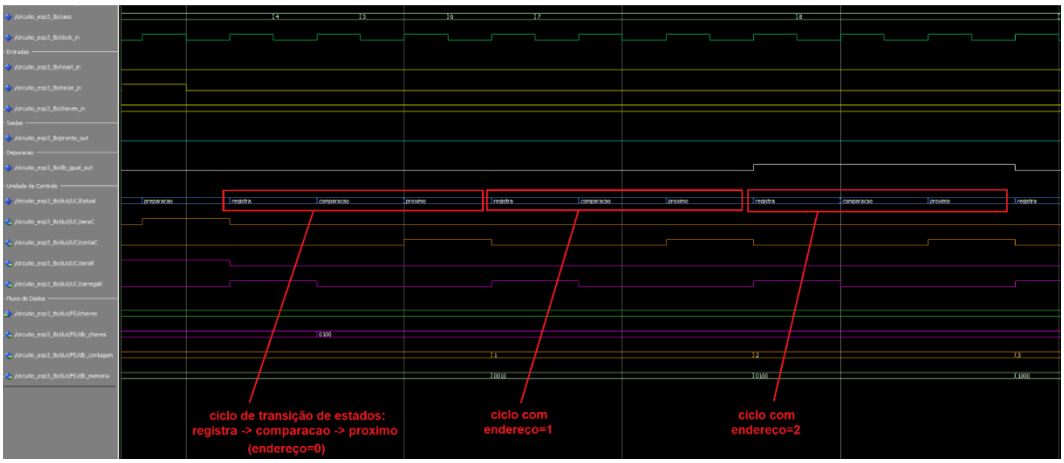

Figura 11: Trecho anotado das formas de onda da simulação do circuito da experiência

h) Crie um projeto no Intel Quartus Prime referente ao circuito da experiência. Documente o projeto elaborado no Planejamento e, ao final, submeta o arquivo QAR (`exp3_TxByy_ativ1.qar`).

Execute a ferramenta **RTL viewer** do Intel Quartus Prime e verifique se a saída produzida é similar ao da Figura 4. Documente no Planejamento.

Tabela 2: Plano de Teste para o Circuito da Experiência

#	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**) <completar todas as saídas>
4	Mantém chaves em 0100 e acionar clock 1x	chaves=0100 iniciar=0 clock ↑	(muda para estado **registra**) <completar todas as saídas>
5	Mantém chaves em 0100 e acionar clock 1x	chaves=0100 iniciar=0 clock ↑	(muda para estado **comparação**) db_igual=0 db_contagem=0000 db_memoria=0001 db_chaves=0100 <completar todas as saídas>
6	Mantém chaves em 0100 e acionar clock 1x	chaves=0100 clock ↑	(muda para estado **próximo**) <completar todas as saídas>
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
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
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 varia 0-1-0 db_contagem varia 0011-0100-0101 db_memoria varia 1000-0100-0010 db_chaves=0100
10	Ajustar chaves para 0001 e acionar clock 6x	chaves=0001 clock ↑ (6x)	<completar todas as saídas>
11	Ajustar chaves para 0010 e acionar clock 6x	chaves=0010 clock ↑ (6x)	<completar todas as saídas>
12	Ajustar chaves para 0100 e acionar clock 6x	chaves=0100 clock ↑ (6x)	<completar todas as saídas>
13	Ajustar chaves para 1000 e acionar clock 6x	chaves=1000 clock ↑ (6x)	<completar todas as saídas>
14	Ajustar chaves para 0000 e acionar clock 3x	chaves=0000 clock ↑ (3x)	<completar todas as saídas>
15	Mantém chaves em 0000 e acionar clock 3x	chaves=0000 clock ↑ (3x)	(passa pelo estado **fim**) pronto=1 <completar todas as saídas>
16	Mantém chaves em 0000 e acionar clock	chaves=0000 clock ↑	(termina no estado **inicial**) <completar todas as saídas>

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 experiência na placa FPGA DE0-CV. Em seguida, este circuito deve ser estudado e seu comportamento deve ser validado conforme projetado.

i) No Laboratório Digital, sintetize o projeto do circuito no FPGA Cyclone V 5CEBA4F23C7N da placa FPGA DE0-CV. Para isso, adote a designação de pinos da placa DE0-CV da Tabela 3 e complete-a com os dados faltantes no Planejamento.

Note que a Tabela 3 apresenta sugestões para designação de sinais de depuração adicionais. Sugere-se que os grupos adicionem tais sinais para a verificação de funcionamento do circuito da experiência.

DICA: Para armazenar os arquivos do projeto do Intel Quartus Prime do grupo na bancada do Laboratório Digital, abra o arquivo QAR do projeto da experiência em uma subpasta chamada `circuito_exp3` dentro da pasta da experiência (p.ex. `C:\Projetos\TxByy\Exp3`).

Tabela 3: Designação de Pinos para a Atividade 2

Sinal	Pino na Placa DE0-CV	Pino no FPGA	Analog Discovery
CLOCK	GPIO_0_D0		StaticIO – Button 0/1
RESET	GPIO_0_D1		StaticIO – Button 0/1
INICIAR	chave SW0		-
CHAVES(0)	chave SW1		-
CHAVES(1)	chave SW2		-
CHAVES(2)	chave SW3		-
CHAVES(3)	chave SW4		-
PRONTO	led LEDR0		-
DB_IGUAL	led LEDR1		-
DB_INICIAR	led LEDR2		-
DB_ZERAC	led LEDR4		-
DB_CONTAC	led LEDR5		-
DB_FIMC	led LEDR6		-
DB_ZERAR	led LEDR8		-
DB_REGISTRAR	led LEDR9		-
DB_CONTAGEM	display HEX0		-
DB_MEMORIA	display HEX1		-
DB_CHAVES	display HEX2		-
DB_ESTADO	display HEX5		-

j) Execute as atividades práticas da implementação e teste do circuito na bancada:

- Programe o projeto sintetizado na placa DE0-CV.
- Interligue os sinais dos canais digitais do Analog Discovery nos pinos da GPIO_0 da DE0-CV. DICA: ligue primeiro o fio GND do Analog Discovery no pino GND da GPIO_0.
- Execute o acionamento da sequência de sinais de controle do Plano de teste do circuito da experiência. Anote os resultados experimentais obtidos.
- Analise os resultados obtidos e elabore um resumo do funcionamento do circuito estudado.

k) Submeta o arquivo QAR final desta atividade (`exp3_TxByy_atividade2.qar`) junto com o Relatório.

1.3. Atividade 3 – Desafio: Modificação do Sistema Digital

Esta atividade adicional tem como objetivo a familiarização com o projeto de circuitos onde o sistema digital é a composição de um fluxo de dados e uma unidade de controle. O sistema digital da Atividade 1 deve ser modificado a partir da especificação apresentada.

I) O funcionamento de um sistema digital modificado é descrito pelo pseudocódigo da Figura 12. As modificações introduzidas no circuito da atividade 1 estão sinalizadas 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.  }
```

Figura 12: Pseudocódigo do Sistema Digital do Desafio

Os sinais de saída `acertou` e `errou` devem ser ativadas somente por um período de *clock*, junto com o sinal `pronto` para indicar o final da operação do circuito e o resultado final.

Adote a seguinte definição para a entidade principal do circuito modificado (Figura 13).

```
entity circuito_exp3_desafio 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;
        db_igual    : out std_logic;
        db_iniciar  : out std_logic;
        db_contagem : out std_logic_vector (6 downto 0);
        db_memoria  : out std_logic_vector (6 downto 0);
        db_chaves   : out std_logic_vector (6 downto 0);
        db_estado   : out std_logic_vector (6 downto 0)
    );
end entity;
```

Figura 13: Entidade VHDL para o Desafio

Projete a modificação do sistema digital. Documente as modificações necessárias tanto no Fluxo de Dados como na Unidade de Controle no Relatório.

DICA: não use o mesmo diretório ou pasta para armazenar os arquivos do novo projeto do Intel Quartus Prime na bancada do Laboratório Digital. Crie uma subpasta chamada `exp3_desafio` na pasta da experiência (p.ex. `C:\Projetos\TxByy\Exp3`).

m) Elabore um Plano de Testes para estudar o funcionamento do circuito. Considere ao menos dois **cenários de teste** para o circuito modificado: um com o **acerto dos 16 dados da memória** e outro com **erro na comparação do 4º dado da memória**. Para cada cenário, gere uma tabela com o Plano de testes seguindo a estrutura da Tabela 4 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 4: Modelo de Tabela para os Cenários do Plano de Testes do Desafio

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

n) Realize a simulação do projeto com o ModelSim executando os cenários de testes do item anterior. Inclua as formas de onda obtidas na documentação da experiência. Execute adaptações ao arquivo de *testbench* da experiência para criar os *tesbenches* de cada cenário.

o) Sintetize o projeto do circuito no FPGA Cyclone V 5CEBA4F23C7N. Para isto, adote a designação mínima de pinos da Tabela 5 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 no relatório).

Tabela 5: Designação Mínima de Pinos para o Desafio

Sinal	**Pino na Placa DE0-CV**	**Pino no FPGA**	**Analog Discovery**
CLOCK	GPIO_0_D0		StaticIO – Button 0/1
RESET	GPIO_0_D1		StaticIO – Button 0/1
INICIAR	chave SW0		-
CHAVES(0)	chave SW1		-
CHAVES(1)	chave SW2		-
CHAVES(2)	chave SW3		-
CHAVES(3)	chave SW4		-
PRONTO	Led LEDR0		-
DB_IGUAL	Led LEDR1		-
DB_INICIAR	Led LEDR2		-
ACERTOU	Led LEDR8		-
ERROU	Led LEDR9		-
DB_CONTAGEM	Display HEX0		-
DB_MEMORIA	Display HEX1		-
DB_CHAVES	Display HEX2		
DB_ESTADO	Display HEX5		-

p) Programe o projeto na placa DE0-CV, execute as ligações de sinais do Analog Discovery e execute o acionamento da sequência de sinais de controle do Plano de teste. Anote os resultados experimentais obtidos.

q) Submeta o arquivo QAR final desta atividade (`exp3_TxByy_desafio.qar`) e os arquivos de *testbench* de cada cenário de teste 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.

r) A experiência mostra como um sistema digital pode ser desenvolvido a partir de sua descrição textual. Com base no aprendizado da experiência, responda as seguintes questões no Relatório:

1. Mostre quais elementos do fluxo de dados da atividade 1 podem ser inferidos a partir do seguinte trecho da descrição: "*Depois de iniciar seu funcionamento, o circuito deve armazenar e depois comparar o conteúdo das chaves de entrada (sinal chaves) com o respectivo dado da memória e deve indicar o resultado da comparação na saída de depuração db_igual.*"
2. Mostre o trecho do diagrama de estados de alto nível do circuito da atividade 1 que pode ser associado com o seguinte trecho da descrição: "*O ciclo de comparação e reposicionamento da memória deve prosseguir até que todos os 16 dados sejam verificados. Ao final da operação, o sinal de saída pronto deve ser acionado por um período de clock. Depois disso, o circuito deve voltar para o estado inicial para aguardar a próxima ativação de iniciar*."
3. Que modificações seriam necessárias no projeto da atividade 1 caso o trecho final da descrição fosse mudado para "*Ao final da operação, o sinal de saída pronto deve permanecer ativado até que o circuito seja reiniciado com um novo acionamento da entrada iniciar*." Explique quais elementos do projeto deveriam ser alteradas e como.

Ilustre suas respostas com anotações nas figuras do circuito da experiência.

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, TEXAS INSTRUMENTS. **Digital Logic Pocket Data Book**. 2006.

[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 com FPGA DE0-CV com o dispositivo Cyclone V 5CEBA4F23C7N.
- 1 dispositivo Analog Discovery da Digilent ou equivalente.

Histórico de Revisões

E.T.M. / 2018 (versão inicial)
E.T.M. & A.V.S.N / 2021 (revisão e adaptação para ensino remoto)
E.T.M. / 2022 (revisão e adaptação)
E.T.M. / 2023 (revisão, reorganização e adaptação para ensino presencial)