UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS

SEL0384 – Laboratório de Sistemas Digitais I

Prof. Dr. Maximiliam Luppe

Bárbara Fernandes Madera - nº: 11915032

Johnny Caselato Guimarães - nº: 11915481

PRÁTICA Nº8
Dispositivos de Lógica Programável tipo FPGA
Registradores e Parametrização

SÃO CARLOS

2023

1. Objetivo

O propósito deste relatório é apresentar o processo de implementação de um registrador de N bits utilizando a linguagem de descrição de hardware VHDL no kit Mercurio® IV (Cyclone® IV EP4CE30F23). O registrador será projetado de forma parametrizável através de uma estrutura de arquitetura genérica, podendo ser sobrescrita para outras aplicações.

2. Introdução

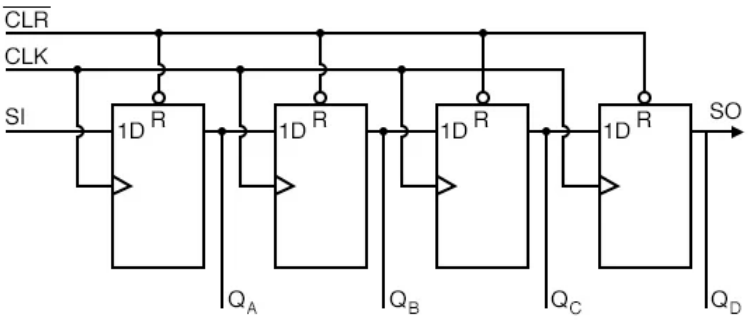
2.1 Introdução Teórica

Conceitos de Registradores:

Registradores são dispositivos de armazenamento de dados em circuitos digitais. Eles podem ser classificados como paralelos ou seriais, dependendo da forma como armazenam e recuperam dados, ou seja, da forma como as entradas e as saídas lidam com os dados. Enquanto os registradores paralelos armazenam todos os bits de uma palavra de dados simultaneamente, sendo úteis para operações em largura de dados maiores e acesso rápido; os registradores seriais, por sua vez, armazenam e recuperam dados bit a bit, serialmente, de forma sequencial. Sendo assim, mais simples e eficientes para transmissão serial de dados e economizam espaço.

É interessante notar também que é possível utilizar arquiteturas mistas, onde lógicas de dados em série e em paralelo, nas entradas ou saídas de bancos de registradores, são aplicadas ao mesmo tempo como na figura 1. Isso pode ser muito útil para otimizar determinados processos em um sistema maior, a fim de obter a máxima eficiência.

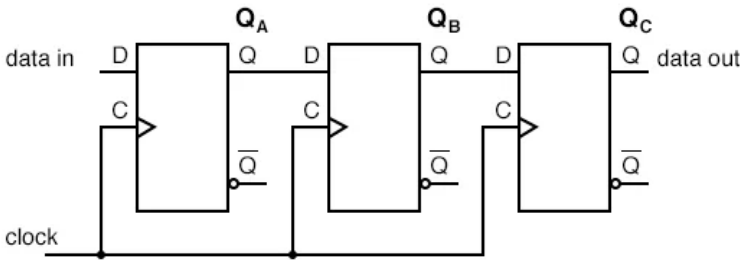
Figura 1 - Registrador de entrada serial e saída paralela.



Registradores Paralelos: Possuem entradas e saídas de dados múltiplos, mostrando as linhas de dados e os sinais de controle. Cada bit é armazenado e

recuperado simultaneamente, permitindo a operação paralela. No caso apresentado acima, apenas a saída dos dados é paralela, enquanto a entrada ainda é sequencial, ou seja, depende de apenas um sinal a ser deslocado para cada bit.

Figura 2 - Registrador de entrada serial e saída serial.



Registradores Seriais: Apresentam uma entrada e saída de dados sequenciais. Mostram a entrada serial de bits, bem como a operação sequencial de armazenamento e recuperação dos dados, como é visto na figura 2, onde vemos apenas uma via de entrada e uma via de saída.

Circuitos Registradores Paralelos com Clock Enable:

Os circuitos de registradores paralelos com Clock Enable possuem a capacidade de carregar ou transferir dados de forma paralela, condicionados à ativação de um sinal de clock, garantindo a sincronização e a estabilidade das operações. Dessa forma, compreende-se como contador como sendo circuitos sequenciais que geram uma sequência predefinida de valores, tais que podem ser síncronos ou assíncronos. Contadores Síncronos apresentam contagem sincronizada com um sinal de clock tal que se atualiza de maneira simultânea ao pulso do clock. Assim, são utilizados em aplicações onde a precisão temporal é crucial. Os Contadores Assíncronos, por outro lado, não dependem necessariamente de um sinal de clock para atualizar a contagem, operando de acordo com regras lógicas ou eventos específicos. Podem, por conseguinte, ser menos precisos e mais complexos.

Contadores com Clock Enable e Parallel Load

Os contadores com Clock Enable e Parallel Load oferecem a capacidade de controlar a ativação da contagem com o sinal de clock, enquanto a operação de carregamento paralelo permite o carregamento instantâneo de dados em paralelo.

2.2 Introdução da Prática

A seguinte prática em laboratório tem como foco a criação de um registrador baseado em Flip-Flop Tipo-D no kit Mercurio® IV. Neste sentido,os registradores são componentes-chave em sistemas digitais e possuem a função de armazenar dados por um determinado período de tempo. Dessa forma, são compostos por Flip-Flops (FF) e possuem terminais como entrada e saída de dados, clock, set/reset (síncronos ou assíncronos) e habilitação para o clock.

Na hierarquia de memória de um sistema digital, os registradores desempenham um papel fundamental. Eles podem ser classificados como auxiliares (não visíveis ao programador, usados para o funcionamento interno) e de dados (visíveis ao programador). Exemplos incluem registradores temporários da Unidade Lógica Aritmética (ULA), Banco de Registradores em arquiteturas como RISC-V, e registradores especiais como o Program Counter (PC).

3. Equipamentos Necessários para Prática:

- Kit Mercurio® IV
- Software Quartus II Web Edition

4. Implementação e Resultados

- Código VHDL:

Nesta atividade foram utilizados 3 arquivos VHDL: MercurioIV_Reg, reg e MercurioIV_decod. O código correspondente ao decodificador não será apresentado aqui pois já havia sido disponibilizado anteriormente, além de ter sido implementado em outras atividades como um complemento.

  - MercurioIV_Reg

No código de “MercurioIV_Reg” é feita a interface entre os elementos do hardware no kit com os parâmetros e componentes do software para a operação do registrador, através das chaves e botões, e do decodificador, através do display de 7 segmentos.

```
-- Projeto Registrador
-- Autores:
-- Bárbara Fernandes Madera   - n°: 11915032
-- Johnny Caselato Guimarães  - n°: 11915481
-- Professor: Maximiliam Luppe
-- Data: 01/11/2023

--Definição da entrada MercurioIV_Reg com as portas de entrada e saída
```

```
entity MercurioIV_Reg is
	port(
		SW: in bit_vector(3 downto 0);		--Entrada dos switches
		KEY : in bit_vector(11 downto 0);	--Entrada dos botões
		DISP0_D : out bit_vector(7 downto 0) --Saída para o display de
7 segmentos
	);
end MercurioIV_Reg;

--Definição da arquitetura top para a entidade MercurioIV_Reg
architecture top of MercurioIV_Reg is
	signal q_out : bit_vector(3 downto 0);  --Declaração do sinal q_out

begin
	--Instanciação de um componente registrador de 4 bits
	reg_0 : work.reg
		--Mapeamento dos elementos/parâmetros genéricos
		generic map (n => 4)  --Definição do n° de bits do registrador

		--Mapeamento das portas
		port map(clk => KEY(0), en => KEY(2),  d => SW, q => q_out);

	--Instanciação do decodificador MercurioIV_decod
	disp_0 : work.MercurioIV_decod
		--Mapeamento das portas
		port map(hexa => q_out, segments => DISP0_D(6 downto 0));
end top;
```

- reg

E no código de “reg” é implementada lógica de um registrador propriamente dito, inspirado na mesma lógica de funcionamento de um simples flip-flop tipo D, mas com a adição da condicional de *enable* para que o dado seja registrado. Nesta implementação não foi inserida a opção de *reset*.

```
-- Projeto Registrador
-- Autores:
-- Bárbara Fernandes Madera   - n°: 11915032
-- Johnny Caselato Guimarães  - n°: 11915481
-- Professor: Maximiliam Luppe
-- Data: 01/11/2023

--Definição da entidade reg
entity reg is
	generic(
		n : integer := 8  --Tamanho n predefinido para 8 bits
	);
	port(
		clk, en : in bit;		--Entradas do clock e do enable
		d : in bit_vector(n-1 downto 0);--Entrada de dados de tamanho n
		q : out bit_vector(n-1 downto 0)--Saída de dados de tamanho n
	);
end reg;

--Arquitetura RTL para a entidade reg
architecture rtl of reg is
begin
	--Processo para verificar a borda de subida do clock
```

```
        check_clock : process(clk)
        begin

        --Se houver uma borda de subida no clock e o enable estiver ativo
              if(clk'event and clk = '1' and en = '1') then
                    q <= d;            -- saída Q recebe o valor de D
              end if;
        end process;
end rtl;
```

- Circuito RTL:

Figura 3 - Visualização RTL geral.

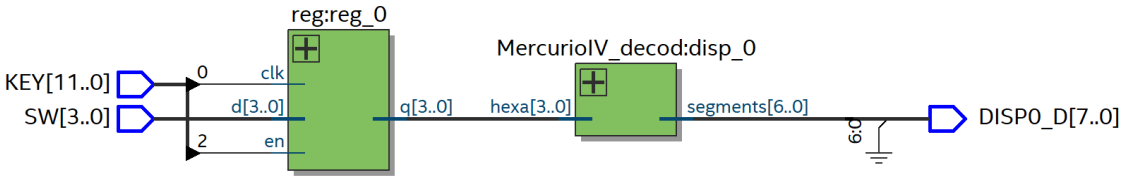
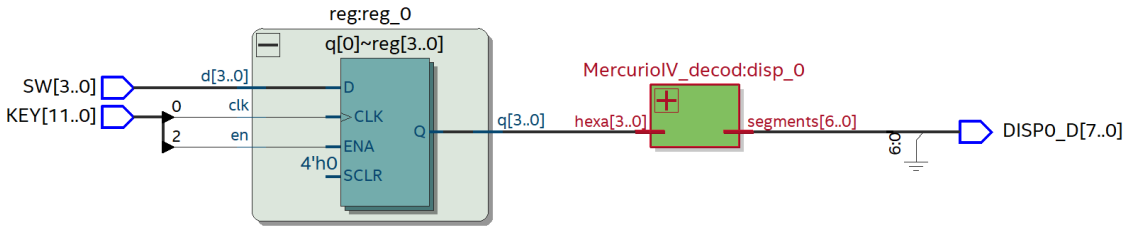


Figura 4 - Visualização RTL expandida.



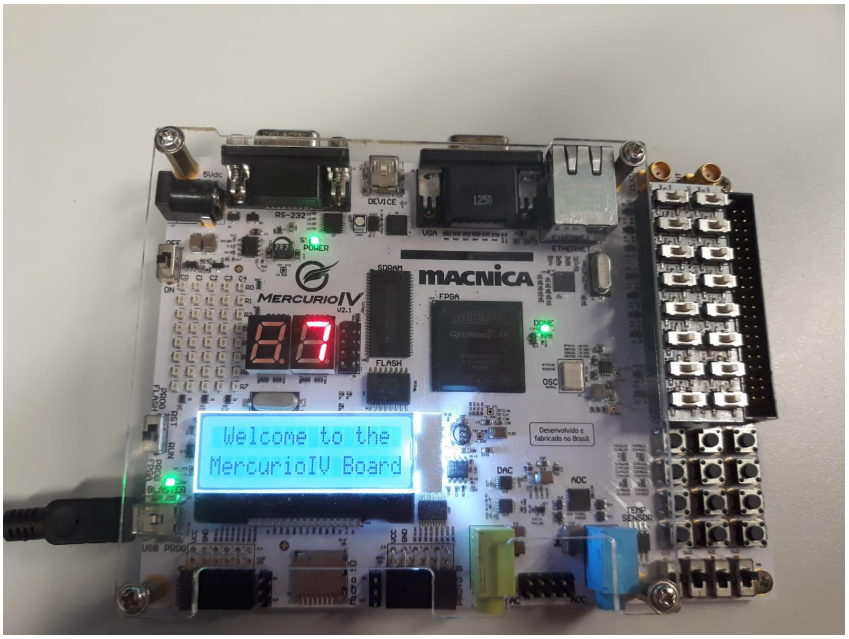
- Número de células lógicas utilizadas:

Figura 5 - Resumo dos resultados da compilação.

Flow Summary	
Flow Status	Successful - Tue Nov 07 22:18:23 2023
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	MercurioIV_Reg
Top-level Entity Name	MercurioIV_Reg
Family	Cyclone IV E
Device	EP4CE30F23C7
Timing Models	Final
Total logic elements	7 / 28,848 (< 1 %)
Total combinational functions	7 / 28,848 (< 1 %)
Dedicated logic registers	4 / 28,848 (< 1 %)
Total registers	4
Total pins	24 / 329 (7 %)
Total virtual pins	0
Total memory bits	0 / 608,256 (0 %)
Embedded Multiplier 9-bit elements	0 / 132 (0 %)
Total PLLs	0 / 4 (0 %)

- Foto do kit em funcionamento:

Figura 6 - Kit Mercurio® IV com implementação bem sucedida.



5. Conclusão

Por conseguinte, este relatório abordou os fundamentos dos registradores, diferenciando entre os tipos paralelos e seriais, além de destacar a importância dos registradores com Clock Enable. Também explorou os conceitos de contadores síncronos e assíncronos, destacando a função do Clock Enable e do Parallel Load, ao modo de ressaltar a representação esquemática dos registradores, ilustrando suas operações e funcionalidades de armazenamento e recuperação de dados.

Em sua parte prática, detalho-se o processo de implementação do registrador, enfatizando os conceitos fundamentais, a descrição do processo em VHDL e a validação prática no kit Mercurio® IV, oferecendo uma compreensão clara da implementação de circuitos sequenciais utilizando dispositivos de lógica programável do tipo FPGA.

Neste sentido, este documento oferece uma abordagem experimental e os resultados esperados para a implementação bem-sucedida do registrador no kit Mercurio® IV, mais especificamente sobre a implementação de um registrador de N bits utilizando a linguagem de descrição de hardware VHDL no kit Mercurio® IV (Cyclone® IV EP4CE30F23).