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 Nº7
Dispositivos de Lógica Programável tipo FPGA
Circuitos Sequenciais

SÃO CARLOS

2023

1. Introdução

O objetivo deste projeto é realizar a implementação das entidades dos Flip-Flops JK-MS, Tipo-D e Tipo-T, utilizando a linguagem de descrição de hardware VHDL com o uso de estruturas lógicas sequenciais. Os arquivos jk_ff.vhd, d_ff.vhd e t_ff.vhd foram criados para representar esses flip-flops, com entrada de Clock clk de borda positiva, entrada de Clear clr ativa em '1', uma saída q e as entradas j, k, d e t, respectivamente.

2. Equipamentos Necessários para Prática:

- Kit Mercurio® IV
- Software Quartus II Web Edition

3. Implementação e Resultados

Os diferentes flip-flops foram unidos em uma entidade superior denominada “MercurioIV_FF”, na qual são declarados e associados os elementos de hardware da placa física, às entradas e saídas dos componentes da lógica em código (no caso, os flip-flops).

Cada implementação teve então seu funcionamento definido com sua respectiva equação característica (ou lógica simples como o flip-flop tipo D).

- **Implementação da entidade *top level* “MercurioIV_FF”:**

```
--Projeto Flip-Flops com estruturas sequenciais
--SEL0384 - Atividade 7
--Autores:
--Johnny Caselato Guimaraes - N: 11915481
--Barbara Fernandes Madera - N: 11915032
--Prof.: Maxmillian Lupe
--Entidade e arquitetura para a operação de 3 flip-flops

entity MercurioIV_FF is
      port(
            SW    : in bit_vector(3 downto 0);  -- Entradas SW (chaves) de
4 bits
            KEY   : in bit_vector(11 downto 0);  -- Entradas KEY (botões)
```

```
de 12 bits
            LEDM_C: out bit_vector(4 downto 0);  -- Saída LEDM_C de 5 bits
            LEDM_R: out bit_vector(7 downto 0)  -- Saída LEDM_R de 8 bits
      );
end MercurioIV_FF;

architecture top of MercurioIV_FF is
      signal LEDs :     bit_vector(3 downto 1);  -- Sinal interno para LEDS
begin
      jkff_0:     work.jk_ff port map(clk => KEY(0), clr => KEY(2), j =>
SW(3), k => SW(2), q => LEDs(1));  -- Mapeamento do flip-flop JK
      dff_0       :     work.d_ff port map(clk => KEY(0), clr => KEY(2), d
=> SW(1), q => LEDs(2));  -- Mapeamento do flip-flop D
      tff_0       :     work.t_ff port map(clk => KEY(0), clr => KEY(2), t
=> SW(0), q => LEDs(3));  -- Mapeamento do flip-flop T

      LEDM_R <= "11111110";  -- Configuração da saída LEDM_R
      LEDM_C(0) <= '1';  -- Configuração do primeiro bit de LEDM_C
      LEDM_C(1) <= not LEDs(1);  -- Configuração do segundo bit de LEDM_C
      LEDM_C(2) <= not LEDs(2);  -- Configuração do terceiro bit de LEDM_C
      LEDM_C(3) <= not LEDs(3);  -- Configuração do quarto bit de LEDM_C
      LEDM_C(4) <= '1';  -- Configuração do quinto bit de LEDM_C
end top;
```

- **Implementação da entidade “d_ff”:**

```
--Projeto Flip-Flops com estruturas sequenciais

--SEL0384 - Atividade 7

--Autores:

--Johnny Caselato Guimaraes - N: 11915481

--Barbara Fernandes Madera - N: 11915032

--Prof.: Maxmillian Lupe

--Entidade e arquitetura de um flip-flop tipo D


entity d_ff is
      port(
       clk, clr, d: in bit;  -- Entradas do flip-flop: Clock, Clear e Dado
       q : buffer bit  -- Saída do flip-flop: Q
      );
end d_ff;

architecture rtl of d_ff is
begin
      check_clock : process(clk, clr)
      begin
            if(clr = '1') then
                  q <= '0';   -- Se Clear estiver ativo, a saída Q é
definida como 0
            elsif (clk'event and clk = '1') then
                  q <= d;  -- Na borda de subida do Clock, a saída Q recebe
o valor de D
            end if;
      end process;
end rtl;
```

- **Implementação da entidade “jk_ff”:**

```
--Projeto Flip-Flops com estruturas sequenciais
--SEL0384 - Atividade 7
--Autores:
--Johnny Caselato Guimaraes - N: 11915481
--Barbara Fernandes Madera - N: 11915032
--Prof.: Maxmillian Lupe
--Entidade e arquitetura de um flip-flop tipo JK

entity jk_ff is
            port(
             clk, clr, j, k   :  in  bit;   --  Entradas  do  flip-flop  JK:
Clock, Clear, J e K
             q : buffer bit  -- Saída do flip-flop JK: Q
            );
end jk_ff;

architecture rtl of jk_ff is
begin
      check_clock : process(clk, clr)
      begin
            if(clr = '1') then
                  q  <=  '0';   --  Se  Clear  estiver  ativo,  a  saída  Q  é
definida como 0
            elsif (clk'event and clk = '1') then
                  q <= (j and (not q)) or ((not k) and q);  -- Equação de
saída Q com base em J e K
            end if;
      end process;
end rtl;
```

- **Implementação da entidade “t_ff”:**

```
--Projeto Flip-Flops com estruturas sequenciais
--SEL0384 - Atividade 7
--Autores:
--Johnny Caselato Guimaraes - N: 11915481
--Barbara Fernandes Madera - N: 11915032
--Prof.: Maxmillian Lupe
--Entidade e arquitetura de um flip-flop tipo T

entity t_ff is
      port(
       clk, clr, t: in bit;  -- Entradas do flip-flop T: Clock, Clear e T
       q : buffer bit  -- Saída do flip-flop T: Q
      );
end t_ff;
```

```
architecture rtl of t_ff is
begin
	check_clock : process(clk, clr)
	begin
		if(clr = '1') then
			q <= '0';   -- Se Clear estiver ativo, a saída Q é
definida como 0
		elsif (clk'event and clk = '1') then
			q <= t xor q;  -- A saída Q é calculada com base em T e o
valor atual de Q
		end if;
	end process;
end rtl;
```

Após a compilação com sucesso do projeto foram utilizadas no total apenas 3 células lógicas.

Figura 1 - Resultados da compilação

Flow Summary	
Flow Status	Successful - Wed Nov 01 00:09:08 2023
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	MercurioIV_FF
Top-level Entity Name	MercurioIV_FF
Family	Cyclone IV E
Device	EP4CE30F23C7
Timing Models	Final
Total logic elements	3 / 28,848 (< 1 %)
Total combinational functions	2 / 28,848 (< 1 %)
Dedicated logic registers	3 / 28,848 (< 1 %)
Total registers	3
Total pins	29 / 329 (9 %)
Total virtual pins	0
Total memory bits	0 / 608,256 (0 %)
Embedded Multiplier 9-bit elements	0 / 132 (0 %)
Total PLLs	0 / 4 (0 %)

E então, a seguir, temos as conexões internas da lógica através da visualização em RTL do sistema na figura 2, comprovando a correta implementação dos flip-flops, observados com mais clareza na figura 3, com a visão interna dos componentes expandida, a qual explicita as funções lógicas aplicadas.

Figura 2 - Visualização RTL geral.

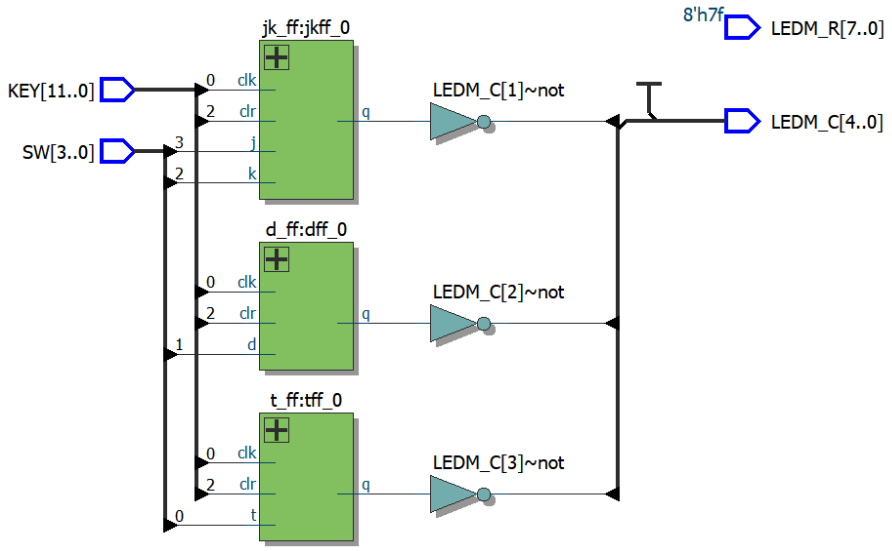
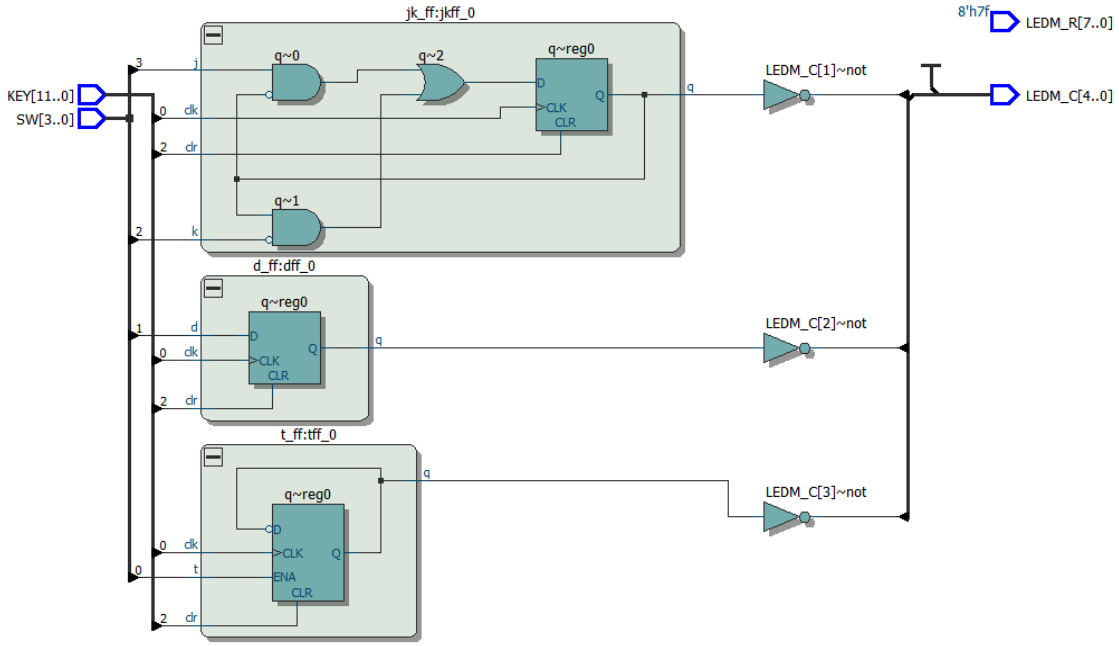


Figura 3 - Visualização RTL expandida.



4. Conclusão

Neste conjunto de implementações na atividade prática, destacamos o uso de estruturas lógicas sequenciais no VHDL. Os flip-flops D, T e JK são exemplos notáveis de elementos sequenciais, pois seu comportamento depende diretamente do tempo e da ordem das transições de sinal. O uso de processos sensíveis à borda

de clock e à ativação do sinal Clear (clr) em cada um desses flip-flops permite que eles capturem e armazenem informações de maneira sequencial.