**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**

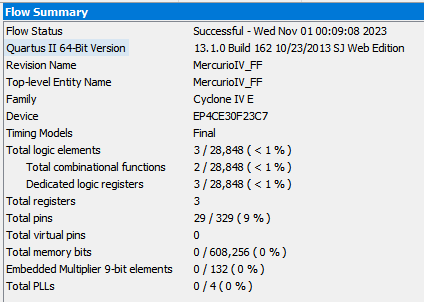
**end if;**

**end process;**

**end rtl;**

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

Figura 1 - Resultados da compilação



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.

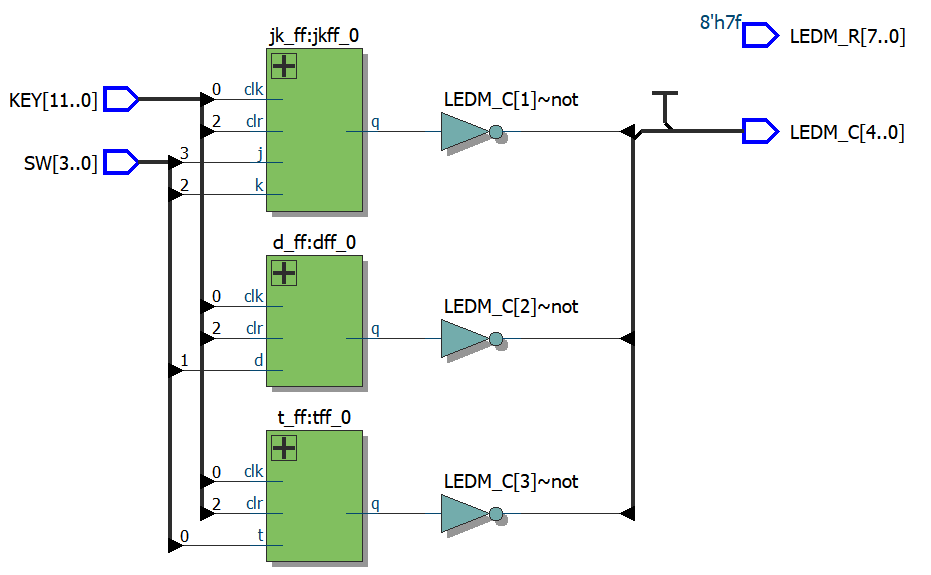
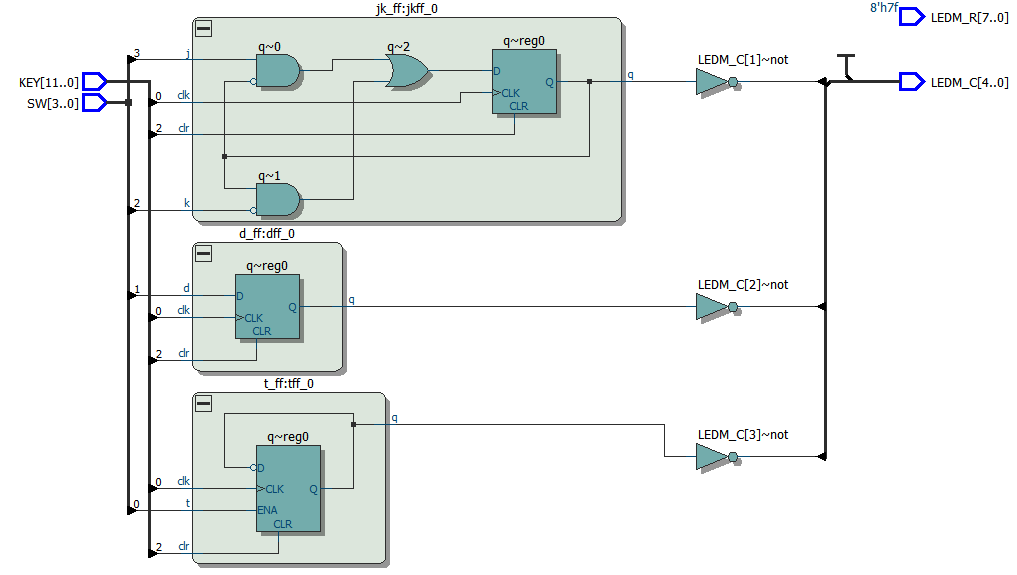


Figura 3 - Visualização RTL expandida.



**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.