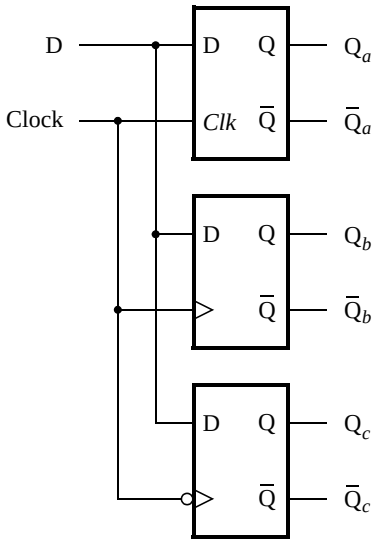
Laboratório 4

Armazenadores e Contadores

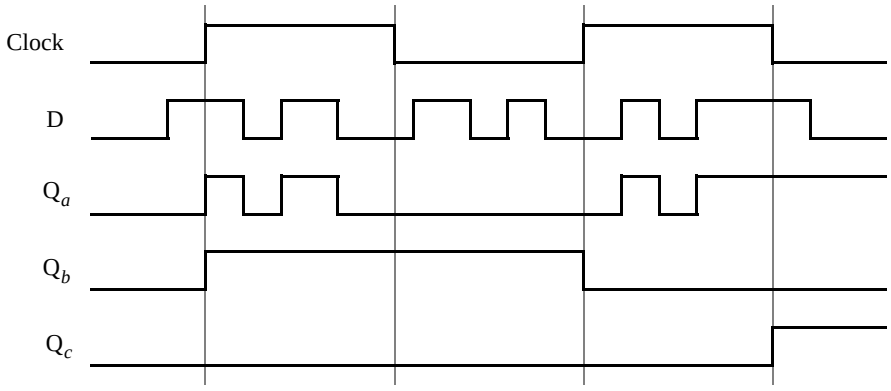
O objetivo deste laboratório é a construção e o uso de armazenadores e contadores.

Parte I

A Figura 1 mostra um circuito com três elementos distintos de armazenamento. um latch D, um flip-flop D sensível à borda de subida, e um flip-flop D sensível à borda de descida.



(a) Circuit



(b) Timing diagram

Figura 1. Circuitos e forma de onda para a Parte I.

Implemente e simule este circuito usando as portas de entrada e saída que achar adequado. Para esta parte, é permitido o uso de códigos em estilo comportamental, como o apresentado na Figura 2. Use as funções *rising_edge* e *falling_edge* para construir os flip-flops sensíveis às bordas de subida e descida, respectivamente.

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;

ENTITY latch IS
   PORT (  D, Clk    : IN      STD_LOGIC ;
           Q         : OUT     STD_LOGIC) ;
END latch ;

ARCHITECTURE Behavior OF latch IS
BEGIN
   PROCESS ( D, Clk )
   BEGIN
      IF Clk = '1' THEN
         Q <= D ;
      END IF ;
   END PROCESS ;
END Behavior ;
```

Figura 2. Código VHDL baseado em comportamento que define um latch tipo D.

Parte II

Considere o circuito da Figura 3. Este é um contador síncrono de 4 bits que utiliza quatro flip-flops do tipo T. O contador incrementa seu valor em cada subida do clock se o sinal *Enable* estiver ativo. O contador retorna para 0 quando o sinal *Clear* se tornar baixo. Você deve implementar um contador de 4 bits deste tipo.

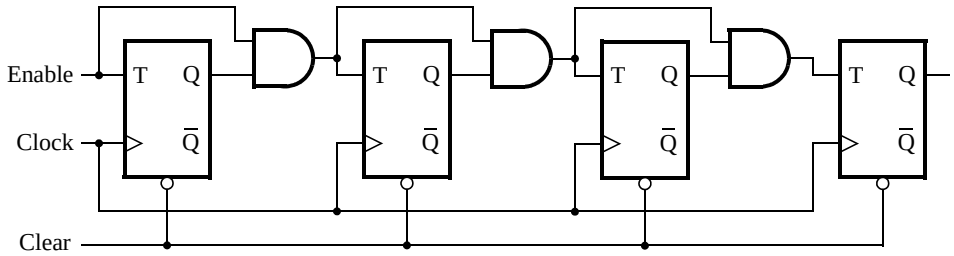


Figura 3. Um contador de 4 bits.

1. Escreva um arquivo VHDL que defina um contador de 4 bits conforme a estrutura da Figura 3. Você deve incluir o módulo que cria o flip-flop T quatro vezes.

2. Crie um novo projeto Quartus II que defina este contador e conecte o botão KEY_0 como *Clock*, as chaves SW_1 e SW_0 como as entradas *Enable* e *Clear*, e o display de 7 segmentos *HEX0* mostra a contagem hexadecimal do circuito

3. Compile e teste seu circuito.