Laboratory Exercise 6 - Week 2
Registers and Applications

The purpose of this exercise is to investigate latches, flip-flops, and registers. This second week focuses on advanced storage elements and their applications.

Part IV

Figure 7 shows a circuit with three different storage elements: a gated D latch, a positive-edge triggered D flip-flop, and a negative-edge triggered D flip-flop.

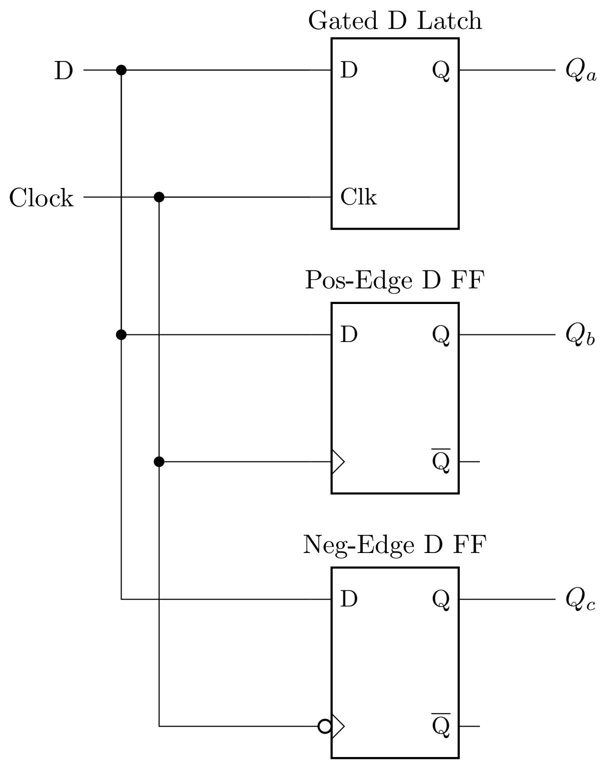

Figure 7: Circuit for Part IV.

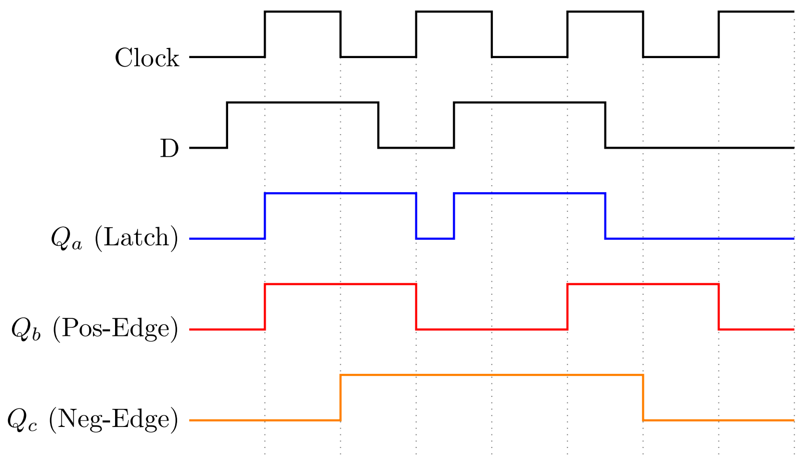

Figure 8: Waveforms for Part IV.

Implement and simulate this circuit using the Quartus software as follows:

1. Create a new Quartus project.
2. **First, implement the circuit using Schematic Design.** Draw the circuit using the Block Editor.
3. Compile the schematic design and perform a functional simulation to verify its correctness.
4. **Next, implement the circuit using Structural VHDL Design.**
5. Write a VHDL file that instantiates the three storage elements. For this part you should no longer use the KEEP directive.
6. Compile your code and use the Technology Map Viewer to examine the implemented circuit. Verify that the latch uses one lookup table and that the flip-flops are implemented using the flip-flops provided in the target FPGA.
7. Create a Vector Waveform File (.vwf) that specifies the inputs and outputs of the circuit. Draw the inputs D and Clock as indicated in Figure 7. Use functional simulation to obtain the three output signals. Observe the different behavior of the three storage elements.

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

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

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

LIBRARY ieee;
USE ieee.std_logic_1164.all;

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

ARCHITECTURE Behavior OF pos_edge_d_ff IS
BEGIN
    PROCESS ( Clk )
    BEGIN
        IF RISING_EDGE(Clk) THEN
            Q <= D;
        END IF;
    END PROCESS;
END Behavior;

LIBRARY ieee;
USE ieee.std_logic_1164.all;

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

ARCHITECTURE Behavior OF neg_edge_d_ff IS
BEGIN
    PROCESS ( Clk )
    BEGIN
        IF FALLING_EDGE(Clk) THEN
            Q <= D;
        END IF;
    END PROCESS;
END Behavior;
```

Figure 9: VHDL code for the D latch and D flip-flops.

Part V

We wish to display the hexadecimal value of an 8-bit number A on the two 7-segment displays HEX3-2. We also wish to display the hex value of an 8-bit number B on the two 7-segment displays HEX1-0. The values of A and B are inputs to the circuit which are provided by means of switches SW7-0. To input the values of A and B, first set the switches to the desired value of A, store these switch values in a register, and then change the switches to the desired value of B. Finally, use an adder to generate the arithmetic sum S = A + B, and display this sum on the 7-segment displays HEX5-4. Show the carry-out produced by the adder on LEDR(0).

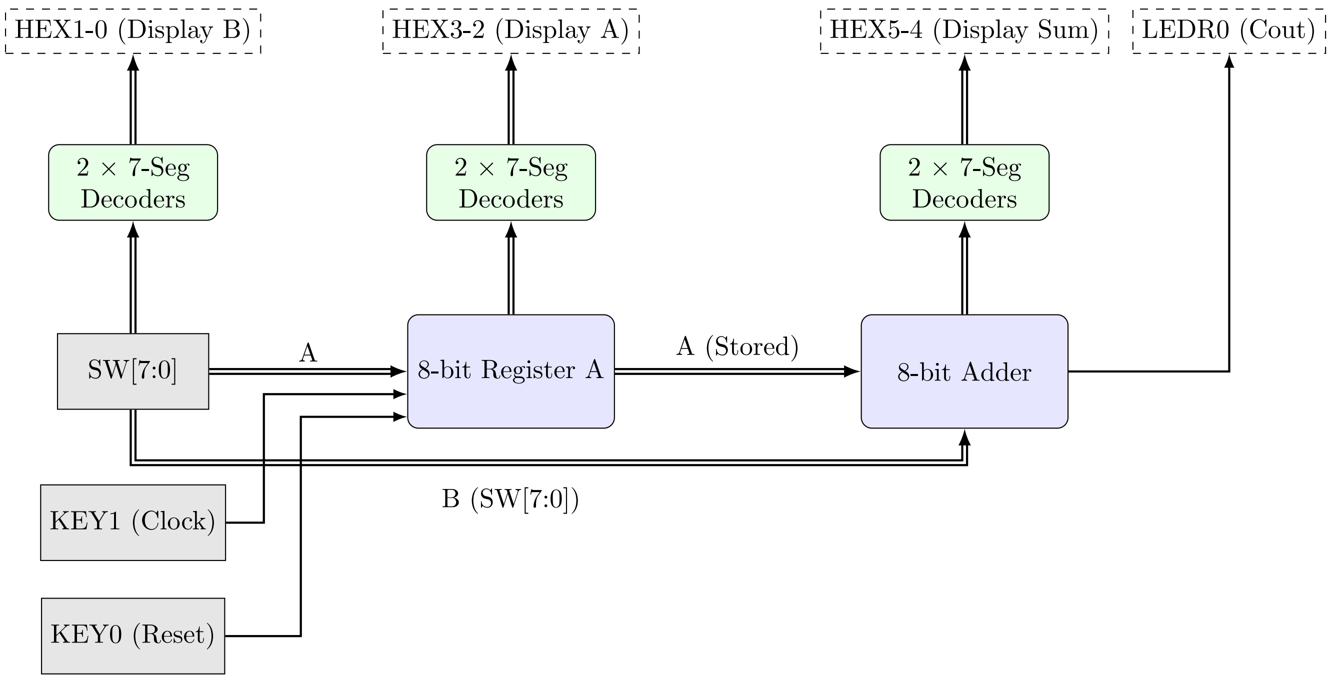

Figure 10: Block diagram for Part V.

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

ENTITY reg8 IS
    PORT ( D : IN STD_LOGIC_VECTOR(7 DOWNTO 0);
           Resetn, Clock : IN STD_LOGIC;
           Q : OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END reg8;

ARCHITECTURE Behavior OF reg8 IS
BEGIN
    PROCESS ( Resetn, Clock )
    BEGIN
        IF Resetn = '0' THEN
            Q <= (OTHERS => '0');
        ELSIF RISING_EDGE(Clock) THEN
            Q <= D;
        END IF;
    END PROCESS;
END Behavior;
```

Figure 11: VHDL code for an 8-bit register with asynchronous reset.

1. Create a new Quartus project which will be used to implement the desired circuit on your DE0-CV board.
2. **Implement the circuit using Structural VHDL Design.** Write a VHDL file that provides the necessary functionality. Use KEY0 as an active-low asynchronous reset, and use KEY1 as a clock input.

3. Include the necessary pin assignments for the pushbutton switches and 7-segment displays, and then compile the circuit.
4. Download the circuit onto your DE0-CV board and test its functionality by toggling the switches and observing the output displays.

Updated By: R. Sutthaweekul
Release Date: 2026-01-01