Rotating LED Circuit

Signature and Grading Sheet

Group #:_____ Name(s):_____ .

Signature

Section 4.4(g): _____.

Grading

- Section 4.1(a): VHDL code (10 points):_____.
  Attach code printout
- Section 4.2(a): VHDL code (10 points):_____.
  Attach code printout
- Section 4.2(b): Simulation (8 points):_____.
  Attach simulation timing diagram printout.
- Section 4.2 (c) diagram (7 points):_____.
  Attach one-page block diagram
- Section 4.3(a): VHDL code (10 points):_____.
  Attach code printout
- Section 4.3(b): Simulation (8 points):_____.
  Attach simulation timing diagram printout.
- Section 4.3 (c) diagram (7 points):_____.
  Attach one-page block diagram
- Section 4.4(b): VHDL code (10 points):_____.
  Attach code printout
- Section 4.4 (e) (5 points):_____.
  Attach one-page report printout with # of LEs circled .
- Section 4.4 (f) : demo signature (15 points):_____.

Total points: _____.

Experiment
Rotating LED Circuit

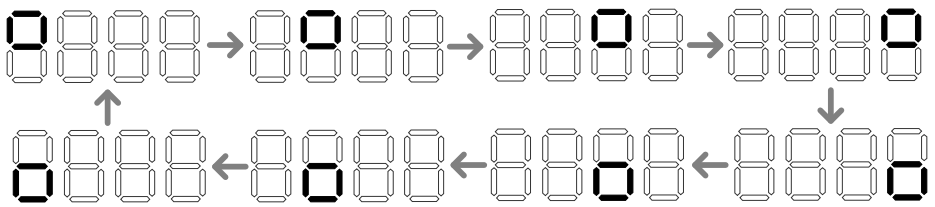
1 Purpose

To design a intermediate-sized sequential circuit

2 Reading

- Chapter 5 of *Embedded SoPC Design with Nios II Processor and VHDL Examples*

3 Project specification



In a seven-segment LED display, a square pattern can be created by enabling the a, b, f, and g segments or the c, d, e, and g segments. We want to design a circuit that circulates the square patterns in the four-digit seven-segment LED display. The clockwise circulating pattern is shown above. The control signals of the circuit can specify the rotation speed, the direction of rotation (i.e., clockwise or counterclockwise), and pause the operation.

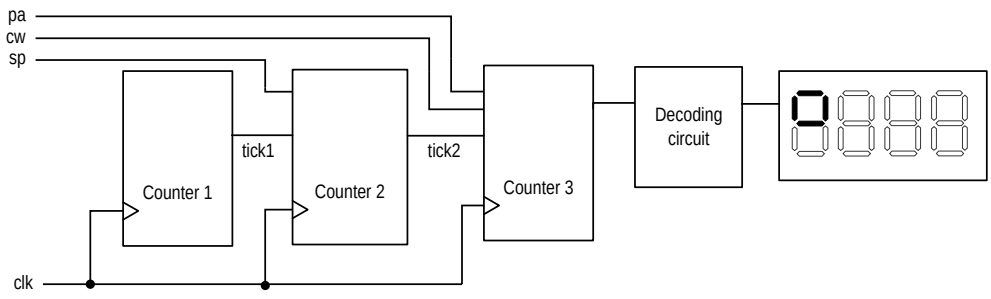
The input and output are

- input:
  - `clk`: 50 MHz clock signal from the DE1 board.
  - `pa`: 1-bit enable signal. The circulation pauses when it is 1.
  - `cw`: 1-bit direction signal. The pattern circulates clockwise when it is 1 and counter clockwise when it is 0.
  - `sp`: 2-bit speed control:
    - 00: each pattern stays 20 ms
    - 01: each pattern stays 40 ms
    - 10: each pattern stays 80 ms
    - 11: each pattern stays 160 ms
- output
  - Four seven-segment LED displays.

The design must synchronous or 50% will be deducted. Note that no reset signal is used in this circuit.

4 Design Procedures

We can divide this circuit into four segments:



- Counter 1: generate a one-clock pulse (`tick1`) every 10 ms.
- Counter 2: utilizes `tick1` pulse and generates a one-clock pulse (`tick2`) every 20 ms, 40 ms, 80 ms or 160 ms based on the `sp` input signal.
- Counter 3: mod-8 counter that utilizes `tick2` pulse and can pause, count up and count down.
- A decoding circuit that generates the desired LED patterns.

4.1 Counter 1

(a) Design the counter 1 as an independent VHDL module and derive VHDL code.

4.2 Counter 2

(a) Design the counter 2 as an independent VHDL module. The entity declaration of this design is

```
entity counter2 is
    port(
         clk: in std_logic;
         tick1: in std_logic;
         sp: in std_logic_vector(1 downto 0);
         tick2: out std_logic
    );
END counter2;
```

Derive the architecture body.

(b) Compile the design and perform simulation to verify its operation. In the timing diagram, the different input combinations should be properly tested.

(c) Draw the top-level diagram of this circuit.

4.3 Counter 3

(a) Design the counter 3 as an independent VHDL module. The entity declaration of this design is

```
entity counter3 is
    port(
         clk: std_logic;
         tick2, cw, pa: std_logic;
         q: out std_logic_vector(2 downto 0);
    );
END counter2;
```

Derive the architecture body.

(b) Compile the design and perform simulation to verify its operation. In the timing diagram, the different input combinations should be properly tested.

(c) Draw the top-level diagram of this circuit.

4.4 Implementation and testing

(a) Derive the decoding circuit VHDL segment.

(b) Derive the VHDL code for the combined circuit. You can either cut-and-paste the previous codes or create a top-level VHDL file with component instantiations.

(c) Use 4 switches for the control signals and 50MHz oscillator for clock.

(d) Perform pin assignment and synthesize the code.

(e) Look at the compiling report and determine the number of LEs used in circuit.

(f) Download the file to the FPGA device and verify the operation of physical circuit. Demonstrate the circuit to instructor and get signature.