Lab # 6

Design of 1 Hz Clock and Counters

Objectives

- To learn how to divide a given clock to generate a desired frequency clock. To generate a 1 Hz clock from available 25.175 MHz on-board clock.
- To learn how to use a counter with the following capabilities
    - Parallel Load
    - Up/Down Count
- To use a modulo-16 counter available in library.
- To covert a modulo-16 counter into modulo-10 counter and modulo-6 counters.
- To design a modulo-60 counter.

Overview

The Digilab FPGA board has a built-in clock of frequency 25.175 MHz (40ns Time Period). This clock is too fast for visual observations. Thus we need to reduce its frequency to some value that our eye can monitor.

In this lab, you will be reducing the clock to 1 Hz by designing a clock division circuitry. For using this clock circuit in future, you will be making a macro of the circuit.

The lab also requires you to design modulo (mod in short) 16, mod-10 and mod-60 counters. An up/down counter can count up in increasing count value (… 0, 1, 2 …_) or down to reducing count values (… 11, 10, 9, ….). A counter with preload capability can load any user given count value at the start of its operation before continuing its either count-up or count-down function. A mod-16 up/down counter can count up from 0 to 15 and back to 0 or down from 15 to 0 and back to 15. It is so called mod-16 counter as it counts 16 states (0 -15). Similar to mod-16 counter, mod-10 and mod-60 counters imply 10 states (0 – 9) and 60 states (0 -59) counters respectively. A mod-10 counter is also called BCD counter.

This lab also requires you to design and implement mod-16, mod-10 and mod-60 counters.

Design Specifications

1 Hz Clock

Consider a mod-4 counter. The output of mod-4 counter consists of four values (0, 1, 2, 3). These four values can be represented using 2-bits. Thus another name of mod-4 counter is 2-bit counter. Generally, an n-bit counter (that can count 2^n count values) can be named as mod-2^n counter.

A 2-bit (mod-4) counter's count sequence is shown in Figure-6.1. Note that its Q1 output, i.e. its 2nd bit which is also its MSB bit, toggles its value after every 2 clock-cycles. One clock-cycle for Q1 output requires 4 (2^2) clock pulses. Thus we can effectively use this Q1 bit to generate a clock that is divided by 4. Generally speaking, we can divide a clock by (2^n) if we take clock output from its n^{th}-bit.

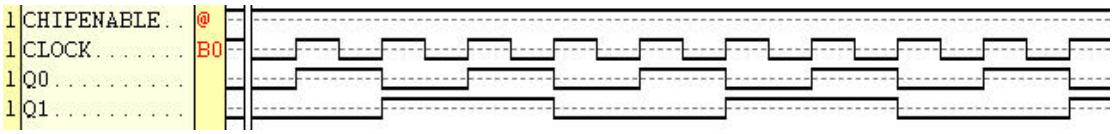

Figure 6.1: Mod-4 Counter Simulation

Thus we see that a counter implements a simple clock division circuitry. In this lab, you are required to divide the provided 25 MHz clock (approximately) to 1 Hz. A simple equation for finding the value for n to generate 1Hz clock can be formulated as

$$\frac{25MHz}{2^{n}{}_{n}} \equiv 1Hz$$

What remains to be found is value for n.

The counters available in Spartan library are maximum 16-bit counters (e.g. CC16CE). If the value of n is coming out to be more than 16, two or more counters have to be cascaded. Cascading is done by using CEO (Chip Enable Output) signal. The CEO signal is at logic high when the counter's value is at its maximum. An example circuit for n = 25 is shown in Figure 6.2

N.B. Cascading by providing derived clocks to successive counters in chain has to be avoided. The derived clock will be dependant on the FPGA routing of the clock signal which in turn depends on the design implementation.

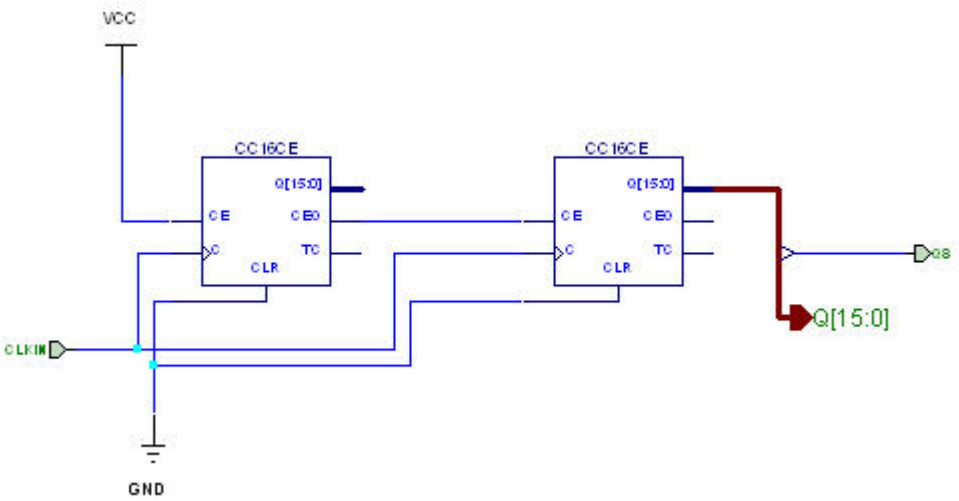

Figure 6.2: An Example Clock Division Circuit (n = 25)

A snapshot for simulation for the above circuit is shown in Figure 6.3. Note how CEO signal enables the cascaded counter. The CEO signal is produced when the first counter has reached its maximum count (FFFF). This CEO signal enables the cascaded counter which increments its count value by one at the occurrence of next active clock edge.

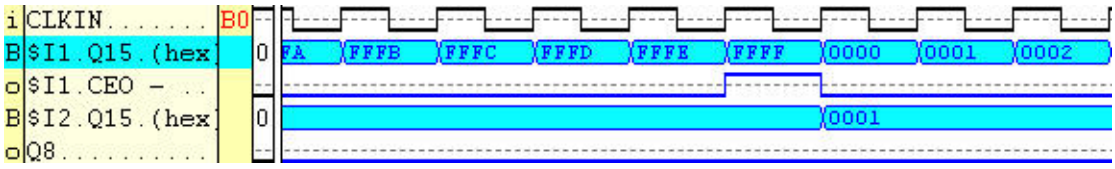

Figure 6.3: Simulation for circuit given in Figure 6.2

Mod-16 Counter

The mod-16 (4-bit) counter with up/down and preload capabilities is already available in the Spartan library (CB4CLED). Apart from counter outputs (Q) and clock input (C), the library component CB4CLED, also has data input lines (denoted by D) for parallel load. The parallel load loads binary pattern available at the D-inputs as its initial value. It is initiated by providing a pulse at the pin 'L' after providing the load value at the D-lines. The counter also features Up/Down counting. This is controlled by using the control signal 'UP'. A high on this signal makes the counter count upwards whereas a low makes it count downwards.

Mod-10 Counter

The modulo-10 counter is a counter that has states from 0 to 9. Thus it is also called a BCD counter.

In this lab, you are required to use your existing mod-16 counter to design a mod-10 counter. Note the counter sequence required for modulo-10 counting

0000 – (0)
0001 – (1)
0010 – (2)
.
.
.
1001 – (9)
0000 – (0)
0001 – (1)
.
.
.

A modulo-10 counter can be made from modulo-16 counter. We can use any 16 bit counters available in library like CB16CLED or CD4CE for this purpose. The only thing required to convert modulo-16 counter into modulo-10 it to reset the counter to

initial value 0000 whenever it counts to value 1010 (Decimal-10). The asynchronous clear (CLR) signal available in the counter clears the counter value to all zeros. We can use this signal in modulo-16 counter to asynchronously clear its contents to 0000 whenever value 1010 is reached, providing desired functionality for modulo-10 counting.

A modulo-10 counter made from modulo-16 counter is shown in Figure 6-4

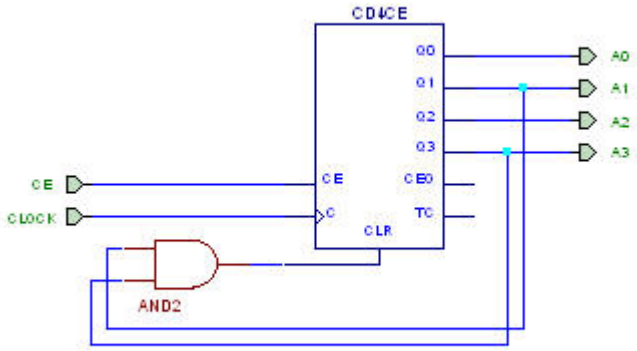

Figure 6.4: Modulo-10 counter

Mod-6 Counter

A mod-6 counter counts from decimal 0 to 5 and then rolls back to 0. A modulo-6 counter can also be constructed using 16-bit counter like we developed modulo-10 counter. This time, we would be asynchronously clearing the counter value to all zeros (0000) on value 0110 (Decimal-6). A mod-6 counter is shown in Figure 6.5.

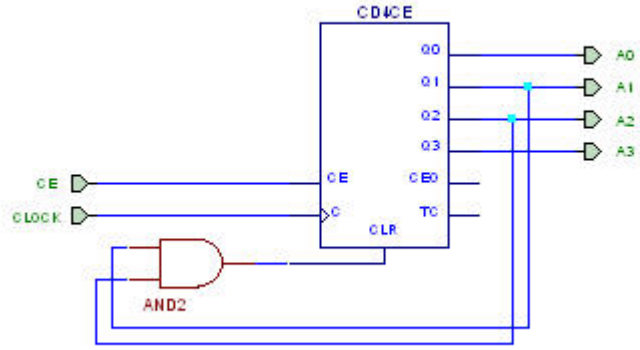

Figure 6.5: Modulo-6 counter

Mod-60 Counter

A modulo-60 counter can be constructed by using a modulo-10 counter for LSB digit and a modulo-6 counter for MSB digit. The MSB digit would only be enabled when

the external CE (Chip Enable) signal is high and LSB counter has reached a value of decimal-9 (Why not decimal-10?).

For our mod-60 counter macro, which is a part of our digital clock that we are developing, we need to produce a Chip Enable Output (CEO) signal to signal external circuitry that the mod-60 counter has reached its maximum value (Decimal-59).

Pre-Lab

- Design and verify the circuit for 1 Hz clock.
- Design and verify the schematic for modulo-10 counter.
- Design and verify the schematic for modulo-6 counter.
- Design, verify and explain the working for modulo-60 counter.

Figure 6.6: Interfacing 1 Hz Clock Macro for FPGA Download

In-Lab

1 Hz Clock

- Start a new project and name it as lab6a.
- Functionally simulate your design

To apply 25 MHz clock for simulation, you may need to change the default 100 MHz clock at B0 bit of the counter (available in Simulator Selection dialog box). This can be done by updating the clock value at Options\Preferences\Simulation to 25 MHz.

- Interface your 1 Hz clock generating macro as shown in Figure 6.6. Note the usage of Global-Buffer instead of Input Buffer. This is a special buffer for signals that are routed throughout the chip. The Clock is one of such signals.
- Download the design and verify that LED is blinking at 1Hz frequency.

Mod-16 Counter

- Start a new project and name it as lab6b.
- Start a new schematic, where you will design 4-bit counter with the above mentioned capabilities. Use one instance of CB4CLED (4-bit counter). It is an input loadable, up/down counter with asynchronous clear.
- Connect the asynchronous Clear (CLR) input to GND.
- Connect the C (clock) input of the counter to the output of your 1Hz clock macro.
- Make sure you provide system clock as input to your macro like you did previously.
- Verify the operation by doing functional simulation and making sure that
    - The counter counts from decimal 0 to 15 and back to 0

    - The counter counts upwards when signal-UP is high and vice versa.
    - The counter parallel loads 4-bit data that you provide on Data lines when signal-L is high.
- Constraint the four inputs (D0, D1, D2, D3) to four level switches.
- Connect chip enable (CE) to VCC to make it enable.
- Constraint the four outputs (Q0, Q1, Q2, Q3) to the four LEDs.
- Constraint the control input UP (for up/down counting) to a level switch.
- Constraint the control input L (Load) to a pulse switch.
- Perform an integrity test. Check to see if there is any error, remove it.
- Implement the design by downloading the design on Digilab board.
- Verify its functionality on the board.
- Demonstrate your work to the instructor.

Mod-10 Counter

- Start a new project and name it as lab6c.
- Import your design schematic for modulo-10 counter.
- Verify its operation by doing functional simulation and making sure that
    - The count sequence is from decimal 0 to 9 and back to 0
    - The CEO signal is high when counter counts to count 9

Mod-60 Counter

- Start a new project and name it as lab6d
- Import your designed mod-60 counter.
- Connect the clock input of the macro to the output of the 1Hz clock Macro
- Connect the input for 1Hz clock macro to internal clock available at P13.
- Verify the operation by doing functional simulation and making sure that
    - The count sequence is from decimal 0 to 59 and back to 0
    - The CEO signal is high when counter counts to count 59
- Constraint the eight outputs to eight LEDs. Use the input/output macros you previously developed.
- Implement and download the design. Verify the operation on the board.
- Demonstrate your work to the instructor

You can use blank boxes under each input/output unit in the diagram below for remembering its usage.