Pulse-Width Modulation

Version 2: Corrected schematic for decimal counter.

Introduction

The average voltage of a pulse waveform is proportional to its duty cycle. A pulse-width modulator is a circuit that generates a pulse signal whose duty cycle can be varied. Pulse-width modulation (PWM) has many applications, including varying the output voltage of DC power supplies.

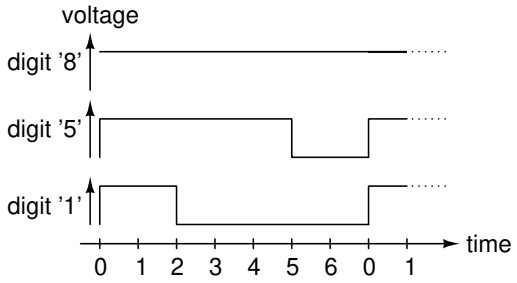
In this lab you will build a pulse-width modulator that varies the duty cycle of the common-anode signal driving a 7-segment LED display. This makes the brightness appear independent of the number of segments that are turned on without having to use one current-limiting resistor per cathode.

The following table shows the number of segments that are lit for each possible decimal digit:

digit	0	1	2	3	4	5	6	7	8	9
segments on	6	2	5	5	4	5	6	3	7	6

For example when displaying the digit '1' only two segments are turned on, while for the digit '8' all seven segments are on.

The diagram below shows the PWM waveforms that would appear on the common anode output for three different digits:



The PWM duty cycle must be set to values between $2/7 = 29\%$ and $7/7 = 100\%$ depending on the digit being displayed.

The frequency of the PWM signal should be high enough that you cannot see the LED turn on and off[1]. However, the period should be long enough that the duty cycle is not affected by the rise and fall times of the circuit. A frequency above 100 Hz or more will be higher than most people's flicker fusion threshold.

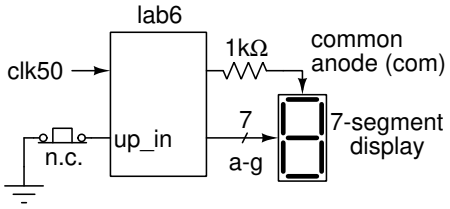
To demonstrate the operation of the PWM function you will also implement a counter that counts from 0 to 9 and drives a 7-segment LED display.

[1] This is called "flicker."

Requirements

Inputs and Outputs

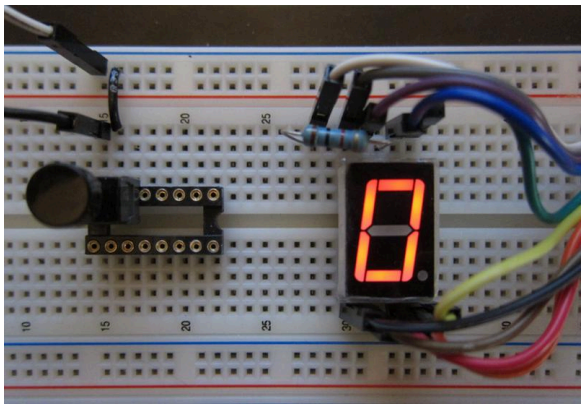
Your circuit will need a 50 MHz clock input, an active-high pushbutton input (**`up_in`**) and a common-anode 7-segment display output:

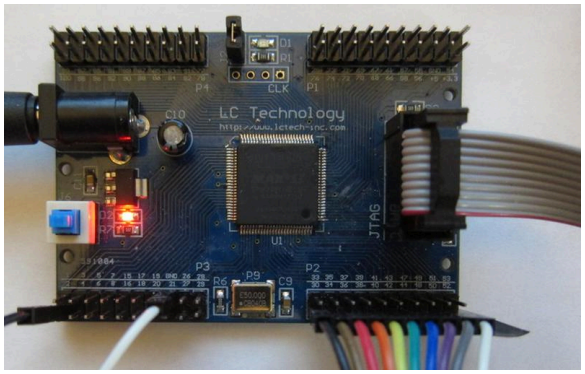


To make use of the supplied **`.qsf`** file the inputs and outputs should be connected as in the previous lab:

To	Assignment Name	Value	Er
up_in	Weak Pull-Up Resistor	On	Yes
up_in	Location	PIN_2	Yes
clk50	Location	PIN_12	Yes
e	Location	PIN_30	Yes
d	Location	PIN_34	Yes
c	Location	PIN_36	Yes
dp	Location	PIN_38	Yes
b	Location	PIN_42	Yes
a	Location	PIN_44	Yes
f	Location	PIN_48	Yes
g	Location	PIN_50	Yes
com	Location	PIN_52	Yes
test	Location	PIN_77	Yes

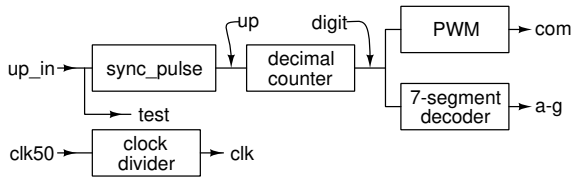
The suggested component layout and wire colours are the same as in the previous lab and are shown below:





Design

A block diagram of the complete circuit is:



Each component is described below.

Clock Divider

Your PWM signal should have a frequency equal to $100 + n$ Hz where n is the last two digits of your BCIT ID number. For example, if you BCIT ID number is A01234567, the frequency of your PWM signal should be 167 Hz. To produce the PWM signal your circuit must generate a clock (named **`clk`** above) with a frequency of seven times the PWM frequency: $7 \times (100 + n)$.

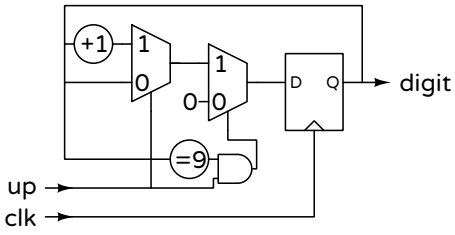
You can use the same clock divider code as in the previous lab and substitute the appropriate divider value.

Pushbutton Debouncer, Synchronizer and Pulse Generator

You can use the **`sync_pulse`** entity provided to debounce, synchronize and generate a one-cycle pulse each time the pushbutton input is pressed. This is the same file provided for the lab exam. The **`clk`** signal should be used as the clock signal.

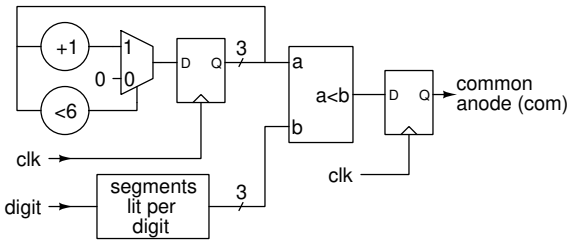
Decimal Counter

To demonstrate that the PWM circuit works you will design a counter counts from 0 to 9 and "wraps around" back to 0. The circuit is as follows:



Pulse-Width Modulator

The following circuit shows a simple PWM generator:



The counter counts from 0 to 6. The PWM output is turned on when the count value is less than the number of segments that are turned on. For example, for the digit '1' (2 segments on) the PWM output is only on for count values 0 and 1 while for the digit '8' (7 segments on) the PWM output is on for count values 0 to 6 (i.e. always on).

A lookup table, labelled "segments lit per digit" can be implemented as an array or a selected assignment and is used to determine the duration of the PWM signal.

Since the PWM signal is going off-chip we pass it through a flip-flop to avoid glitches on the output.

Seven-Segment Decoder

You can use the same array-lookup 7-segment LED decoder as in an earlier lab.

Procedure

Because of the limited time available to prepare for this lab you will be supplied with Quartus archive (`.qar`) file as in the lab exam. It will be missing the clock divider, decimal counter and PWM parts which you will need to design by converting the block diagrams above into VHDL and adding your code to the `lab6.vhd` file.

Download the `lab6.qar` file and extract the archive. Ellipsis ("...") indicates missing code that you need to supply.

Online Lab Help and Demonstration

Since you will not be able to attend the lab in person, you will need to complete the lab at home.

The most effective way to get help will likely be the course discussion forum. This will help avoid repeatedly answering the same question.

However, instructions and a schedule will be made available for sessions during which you can speak with an instructor using the course web site's "Virtual Classroom" (web conferencing) feature to get help on completing your lab.

Once your circuit is working, submit the following to the Activities > Assignments > Lab 6 folder to demonstrate successful completion of the lab:

- a video with a duration of less than one minute in a file format that the browser can display (e.g. `.mp4`) showing the display cycling through the digits 0 to 9 and back to 0. The segment brightness should be constant. An example is available on the course web site[2].

  You may be able to capture the video from your browser using the "Record Video" button in the Assignments folder submission web page.

- the completed VHDL code (as a `.vhd` file)

- a screen capture of your compilation report in a format that the browser can display (e.g. `.png`) similar to this:

Flow Status	Successful - Sat Mar 14 20:30:31 2020
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Web Edition
Revision Name	lab6
Top-level Entity Name	lab6
Family	MAX II
Device	EPM240T100C5
Timing Models	Final
Total logic elements	70 / 240 (29 %)
Total pins	12 / 80 (15 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)

[2] This was made with a hand-held phone, feel free to produce something better; keep the file size reasonable (a few MB).