ECE 2534 – Microcontroller Programming and Interfacing
Homework 4: Using Timer Modules on the PIC32 (Patterson section)
Due at Scholar before 11:00 pm on Oct. 2

$x = \frac{10}{8}$

Honor Code: This is an *individual* assignment. You may discuss general concepts relating to timer operation with other students, but the work you do and the code you submit must be your own.

Problem 1. Consider the timer modules of the PIC32. Assume "16-bit Synchronous Clock Counter Mode," and assume a peripheral-bus clock frequency of f_{PBCLK} = 10 MHz. Let D represent the user-selected prescale value, and let P represent the value in the period register (PRx). Let $f_{RollOver}$ represent how frequently the timer register (TMRx) "rolls over" from P to 0.

Clock freq = 10MHz
Period register = 9999
Prescale = 8

a) If D = 8 and P = 9999, then $f_{RollOver}$ = 125 Hz.

b) If D = 4, then what value of P will cause the same (or nearly the same) value for $f_{RollOver}$ as you obtained in part (a)?
19999

c) Explain why D = 1 may be not be a good choice for generating the value of $f_{RollOver}$ that you obtained in part (a).
It would cause P to be too large a number for a 16-bit clock to contai

Problem 2. Again consider the timer modules of the PIC32, with 16-bit Synchronous Clock Counter Mode and a peripheral-bus clock frequency of f_{PBCLK} = 10 MHz. If possible, find good choices for D and P in order to generate the roll-over frequencies listed below. For each case, report an error value as the actual frequency of roll-over minus the desired $f_{RollOver}$. Naturally, your goal is to select values for D and P that minimize the error magnitude in each case.

a) Desired frequency: $f_{RollOver}$ = 1 Hz. D = 256 P = 39061 Error = .00127994%

b) Desired frequency: $f_{RollOver}$ = 50 Hz. D = 8 P = 24999 Error = 0

$50 = \frac{10}{N}$

c) Desired frequency: $f_{RollOver}$ = 1000 Hz. D = 1 P = 9999 Error = 0

d) Desired frequency: $f_{RollOver}$ = 20000 Hz. D = 1 P = 499 Error = 0

Problem 3. Again consider the timer modules of the PIC32.

a) What is the maximum roll-over <u>frequency</u> when using 16-bit Synchronous Clock Counter Mode?
It can be 5MHz given D=1 and P+1=2

b) What is the maximum roll-over <u>frequency</u> when using 32-bit Synchronous Clock Counter Mode?
It can be 5MHz given D=1 and P+1=2

c) What is the maximum roll-over <u>period</u> when using 16-bit Synchronous Clock Counter Mode?
Given D=256 and P=65536-1 we get .596, but 1/.596 = 1.677 which is next roll period.

d) What is the maximum roll-over <u>period</u> when using 32-bit Synchronous Clock Counter Mode?
Given D=256 and $P = 2^{32}$, we get .00009 or 1/.00009 = 10995 when using?

Problem 4. Write a program for your Digilent board that implements a <u>stopwatch</u>. Show elapsed time on your OLED display in the format MM:SS, representing minutes and seconds, respectively. Each of these is a 2-digit value that increments modulo 60. For example, the displayed value 00:00 should advance to 00:01, and the value 00:59 should increment to 01:00, etc. Your design does not need to keep track of hours, so after the displayed value eventually reaches 59:59, it should simply advance to 00:00 and continue from there.

Your program should initialize the displayed time to 00:00, and this value should not change until the user presses PB1. When the user presses PB1 for the first time, your program starts measuring elapsed time and updates the OLED display once per second. When the user presses PB1 the second time, your program stops updating the display, allowing the user to read it at his or her leisure. When the user presses PB1 the third time, the displayed time is reset to 00:00 and the value stays unchanged until the user initiates the cycle again by pressing PB1 again.

$$f_{preat} = \frac{f_{PBClk}}{Prescaler}$$

$$f_{rollover} = \frac{f_{preat}}{PR_x + 1} = \frac{(f_{PBClk} / PRESCaler)}{PR_x + 1}$$

#2)

a) $\frac{\frac{10,000,000}{8}}{9999 + 1}$

$1Hz = \frac{\frac{10,000,000}{256}}{x} = 39062.2 - 1$

b) $50Hz = \frac{\frac{10,000,000}{8}}{x} = 25000 - 1$

b) $125 = \frac{\frac{10,000,000}{4}}{x}$

$125 = \frac{2500000}{x}$

$x = \frac{2500000}{125}$

$x = 20000 - 1 =$

$x = 19999$

c) $1000 = \frac{\frac{10000000}{1}}{x} = 10000$

d) $20,000 = \frac{\frac{10000000}{1}}{x} = -1$

$10mHz \quad D=256 \quad \frac{\frac{10,000,000}{1}}{1}$

$60000000 \quad P=USE$

$\frac{\frac{10,000,000}{256}}{65536} = \frac{1}{.596} = 1.6775$ sec

$5mHz$

Your program should debounce the PB1 switch, and should take actions based on (debounced) 0-to-1 transitions. Use a timer module to measure the passage of time. Select a timer module that is different from those that were used in Lab 1. (I.e., you should pick either Timer3 or Timer4 or Timer5. It is acceptable to use Lab 1's code as your starting point.) Do not use interrupts to implement your digital clock, and do not use the "sec1000" global variable that has appeared in previous assignments. Instead, initialize your timer so that it rolls over once per second. In your main loop, read the TMRx register to determine when roll-over has occurred. (An alternative approach would be to poll the TxIF Event Flag. But this flag is not available to us in the Timer module, so you are not required to use that flag.)

Use a state-machine approach to develop your code. Update the OLED only once every second – do not update the OLED on every pass through your main "while" loop.

Coding style: Please refer to programming style guidelines posted on Scholar. Provide header comments at the top of your main.c file. Include your name and the assignment name, and state whether or not your code meets the assignment's requirements. Use the header block to describe anything in particular that was difficult about writing this program.

What to submit to Scholar: For problems 1 to 3, place your solutions in a single document. The preferred format is PDF. Show your work to a reasonable extent, and clearly indicate your answers. In order to give you faster feedback, only a subset of these problems will be graded in detail.

For problem 4, submit all of your source files (all *.c and *.h files).

Place all of these files into a single ZIP archive. Use the following file name, and upload it to Scholar before the announced submission deadline:

<Last name>_<First name>_HW4.zip

Take care to upload the correct files. After submitting to Scholar, you can download what you submitted and verify that they are correct. When it is time to grade your work, the grader will download your ZIP file from Scholar, and will compile and run your program on a Digilent board.