Experiment 6: Playing with Time

30.11.2018
Res. Asst. Abdullah Cihan Ak
akab@itu.edu.tr

Most people think time is like a river that flows swift and sure in one direction, but I have seen the face of time and I can tell you they are wrong. Time is an ocean in a storm. You may wonder who I am and why I say this; sit down and I will tell you a tale like none that you have ever heard!

Prince of Persia The Sand of Time, 2003 Video Game

1 Introduction

In the following experiment, you are going to design a chronometer with centisecond precision.

2 Part 1

In this part, you should write an infinite loop as your main program code in order to lit different digits of 7-segment display panel simultaneously. You should be able to accomplish following output given in Figure 1 as the result of the first part.

Hint: If a light flashes in a high frequency, we can not see flashes but only constant light.

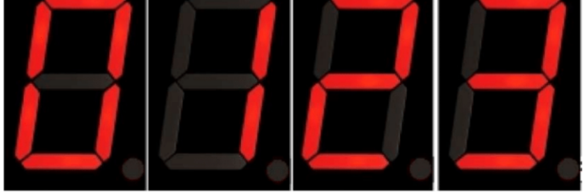

Fig. 1. Sample Output

3 Part 2

In this part, you are going to implement a chronometer. Your chronometer counts up continuously. When the counter overflows, it starts again from 0. Additionally, P2.5 should be used to reset the time back to 0. Reset should work as an interrupt subroutine. There are three additional code blocks that you should include your code in order to build a chronometer. The list of these code blocks are given below.

- Timer Interrupt Subroutine
- Interrupt Subroutine
- BCD Convertion Subroutine

At first define seconds and centiseconds variables in data section as following. You may change the initial value as needed.

```
                .data
seconds         .byte 00h
centiseconds    .byte 00h
```

Timer and Timer Interrupt Subroutine

MSP430 family micro-controllers contain two 16-bit timers which could be utilized independently. Basic building blocks of the time are represented in Figure 2. Basic steps of the configuration and operation of the timer will be explained briefly. For further information, you should read **"Timer-A"** chapter in **"MSP4330 User Guide"**.

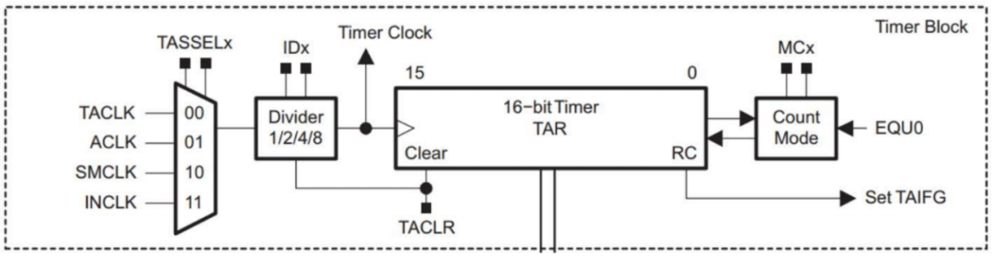

Fig. 2. Timer Circuit

Timer could use four different signals as counting input. In this experiment you should use SMCLK signal as counting input. SMCLK is a square wave with 1048576 Hz frequency. Timer contains two operation modes (capture and compare) four different timer (counting) modes (Stop, Up, Continues, UpAndDown). We are going to use compare mode with up counting mode in this experiments. The list of the registers you should set are given below. Please check **MSP430 User Guide** at pages 369-372.

Registers:

- TimerA Control (TA0CTL): Configuration of the timer
- TimerA Compare Capture (TA0CCR0): Holds the data to compare
- TimerA Comp. Cap. Control (TA0CCTL0): Configuration of Comp/Cap mechanism

Basically you should set the registers with the information given to you at the top. All the important information are given at the top. As you can remember from the previous lab session, microcomputer understands interrupts with interrupt flags and these flags should be cleared before returning from the interrupt. In timer this flag is located in one of the registers on top. Be sure to configure the timer correctly to generate interrupts with 10msec-period.
Hint: If you could not decide the value of a bit, it is probably 0
To declare the timer interrupt please add the following line in the interrupt vector section of your code. Timer interrupts are handled as an ordinary interrupt from this point on.

```
            .sect   ".int09"
            .short  TISR
```

Interrupt Subroutine

Remember your work in the previous experiment about the interrupts and implement an interrupt subroutine that resets the timer to 0 with the input from P2.5.

BCD Conversion Subroutine

You should divide *centiseconds, seconds* values into BCD-digits in order to print them through 7-segment displays. We are holding the time information in variables of centiseconds and seconds but these values can not be written to 7 segment display directly. Convert the time to BCD-digits and use the following array to print the time into the 7-segment display.

```
array   .byte 00111111b, 00000110b, 01011011b, 01001111b, 01100110b
    ,01101101b,01111101b,00000111b,01111111b,01101111b
lastElement
```