University of British Columbia
Electrical and Computer Engineering
ELEC 291

Module 4 –555 Timer/Capacitance Meter

Introduction

From clock sources to test signals, timers are often needed when designing and testing electronic circuits. One of such circuits is the iconic '555 timer' introduced by Signetics in 1971 and designed by Hans R. Camenzind. In this laboratory module, you will be designing, building, and testing a 555 timer equivalent circuit from its basic building blocks. Then you'll use the 555 timer to build a capacitance meter.

In this module you will be programming the C8051F38C/EFM8UB microcontroller using the C programming language. C is the de facto standard for the programming of embedded systems. The C8051F38C/EFM8UB is the microcontroller used in the F38x board you assembled for project 1.

Since the amount of work required to complete this module is more significant than in previous modules, you can work with a partner and submit just one circuit and program for both of you.

References

C51 user manual included with the latest version of CrossIDE.

Any book you have about programming in C. Also, information about the standard C runtime library functions could be handy for this module.
LM555 timer datasheet available at http://www.ti.com/lit/ds/symlink/lm555.pdf

Pre-laboratory

1) The oscillator frequency of a typical astable 555 circuit is given by:

$$f=\frac{1}{T}=\frac{1.44}{(R_A+2R_B)C}$$

Using the block diagram of a 555 timer presented in point 1 of the lab procedure below, derive the above formula.

Laboratory

1) The figure below shows the block diagram of the 555 timer circuit[1]. Using only the basic building blocks (not the 555 chip itself!) design and build an astable oscillator working at 2.88 kHz. Make $R_A=R_B$ and $C=0.1\ \mu F$. Use 5V to power all the ICs. Here are some tips that you may find useful:

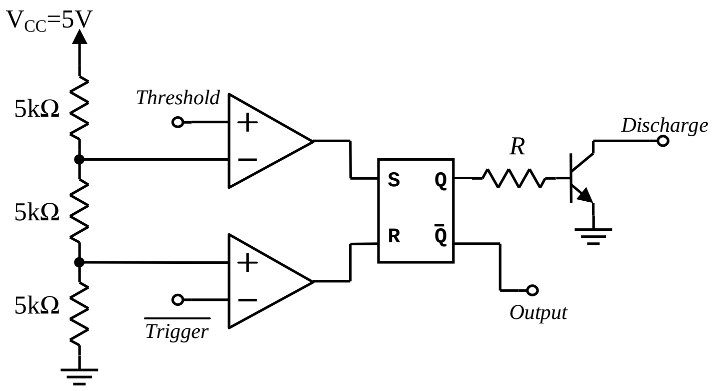
[1] Please notice that this diagram does not include the RESET' pin.

a. There aren't 5 kΩ resistors in your resistor kit. You can use 5.1 kΩ or 4.7 kΩ instead. The actual resistor values are not that critical. What is important in this circuit are the voltage divider values at the '-' and '+' inputs of the two comparators: 2/3 V_{CC} and 1/3 V_{CC} respectively.

b. Use either the LM393 dual comparator or the LM339 quad comparator IC. Remember that these comparator ICs have open collector outputs! Therefore pull-up resistors are needed from the output of each comparator to V_{CC}.

c. Implement the SR flip-flop using either NAND gates (74LS00/74HC00) or NOR gates (74LS02/74HC02). Also remember that the inputs of a RS flip-flop implemented using NAND gates are inverted (R' and S').

d. Choose the base resistor R so that the transistor gets in saturation when the Q output of the RS flip-flop is high (5V). Assume a worst case β=50, and a maximum collector current of 50 mA. Use a 2N3904 or PN2222A *NPN* transistor.



Once you get the circuit working, draw into your notebook the complete astable circuit schematic diagram, as well as the waveforms at the Threshold/Trigger' comparator inputs and the SR flip-flop Q' output. . Compare the actual and theoretical frequency. **Don't take the circuit apart**, as you'll need to demo it to one of your lab TAs by the end of the lab.

2) **Testing C with the F38x board.** The program below prints "Hello, world!" in PUTTY running in a computer throughout the serial port of the C8051F38C/EFM8UB. Copy and paste it to Crosside and save it to 'hello.c'. To compile it under Crosside with C51, click 'Build' → 'Compile/link with C51'. Make sure you set the 'Complete path to C51.exe' correctly. After compiling, load the resulting '.hex' file to the F38x board.

```
#include <C8051f38x.h>
#include <stdio.h>

#define SYSCLK      48000000L  // SYSCLK frequency in Hz
#define BAUDRATE      115200L  // Baud rate of UART in bps

void PORT_Init (void)
{
        P0MDOUT |= 0x10; // Enable UTX as push-pull output
        XBR0     = 0x01; // Enable UART on P0.4(TX) and P0.5(RX)
        XBR1     = 0x40; // Enable crossbar and weak pull-ups
}

void SYSCLK_Init (void)
{
        CLKSEL|=0b_0000_0011; // SYSCLK derived from Internal HF Osc / 1.
        OSCICN |= 0x03;   // Configure internal oscillator for its maximum
        RSTSRC  = 0x04;   // Enable missing clock detector
}

void UART0_Init (void)
{
        SCON0 = 0x10;
        TH1 = 0x10000-((SYSCLK/BAUDRATE)/2L);
        CKCON &= ~0x0B; // T1M = 1; SCA1:0 = xx
        CKCON |=  0x08;
        TL1 = TH1;      // Init Timer1
        TMOD &= ~0xf0;  // TMOD: timer 1 in 8-bit auto-reload
        TMOD |=  0x20;
        TR1 = 1; // START Timer1
        TI = 1;  // Indicate TX0 ready
}

void main (void)
{
        PCA0MD &= ~0x40; // WDTE = 0 (clear watchdog timer enable)
        PORT_Init();     // Initialize Port I/O
        SYSCLK_Init ();  // Initialize Oscillator
        UART0_Init();    // Initialize UART0
        printf( "Hello, world!\r\n" );
}
```

PUTTY is a free Telnet/SSH/Serial terminal that can be downloaded from http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html. It is possible to launch PUTTY directly from CrossIDE by pressing <Control>+T. Configure PUTTY to 115200 baud, 8 bits, parity none, 1 stop bits, and Flow Control 'none'. Also make sure the 'Complete path of PuTTY.exe' field points to a valid location.

3) When used as an astable oscillator, the frequency output of a 555 timer is inversely proportional to the capacitance used in the circuit. If such frequency is measured using a microcomputer system, the capacitor value used in the timer circuit can be determined. Built, either using an actual 555 single timer, a 556 dual timer, or the circuit you assembled in the previous point, a capacitance **meter** that works in the range 1 nF to 1 uF. Use the F38x board and an LCD to measure and display the capacitance. Write the program of your capacitance **meter** using the C programming language. Demonstrate the working system to one of your lab TAs. For your reference, an example on how to measure frequency, 'FreqF38x.c' is provided in the web page of the course.