Microprocessor Lab --- Timer function

Using timer function generate wave form

1. Equipment

- M68HC12EVB
- Oscilloscope
- Signal Generator
- Personal Computer

2. Introduction

The purpose of the timer module is to allow for time critical operations to be handled mostly by hardware, instead of entirely in software. For example, generating or measuring waveforms can be done with minimal input from the processor using the timer module.

The 68HC12 Standard Timer Module consists of a 16-bit programmable timer that is driven by a programmable prescaler mechanism. It also has eight 16-bit input capture/output compare channels, and two pulse accumulators. This hardware is explained in your text book and as well as in its datasheet.
(http://www.freescale.com/files/microcontrollers/doc/ref_manual/S12ECT16B8CV1.pdf?srch=1)

Before anything happens in the Standard Timer Module, the software on the CPU must enable the timer system by setting bits in the appropriate registers. The Timer System Control Register, located at register offset $0086, controls the basic behavior of the entire timer module. Bit 7 enables and disables the timer, bits 6 and 5 determine if the module runs during wait and freeze modes, and bit 4 determines if certain flags are automatically cleared after access to their associated registers.

As described previously, the Standard Timer Module has 8 channels which are individually programmed to be input capture or output compare. The function of each channel must be specified in software by writing to the TIOS (Timer Input Capture/Output Compare Select) register, located at register offset $0080. Setting a bit to 1 makes its associated channel function as an output compare, while setting a 0 makes the channel an input capture. 68HC11 users will like this quite a bit, since you can select the function of each pin, instead of hardwiring it.

The prescaler is a useful device that allows you to control the frequency at which the counter increments by dividing MCLK by a constant. On reset, the prescaler defaults to a value of 1, which means the TCNT register is incrementing at the same frequency as MCLK. Typically MCLK = Crystal Frequency / 2, therefore MCLK on a 16-MHz crystal is 8 MHz. (It is possible to change the speed of MCLK by altering the CLKCTL register, but most people don't do this). Using the PR0-PR2 bits of the TMSK2 register, MCLK can be divided by a number up to 32. The chart below shows the period of a 'tick' for an 8 MHz MCLK, as well as the amount of time it takes TCNT to overflow. (i.e. count from $0000 to $FFFF + 1)

Prescaler value selection table

PR2	PR1	PR0	Period (8mhz MCLK)	TCNT overflow	Prescale Factor
0	0	0	125ns	8.192 ms	1
0	0	1	250ns	16.384 ms	2
0	1	0	500ns	32.768 ms	4
0	1	1	1us	65.536 ms	8
1	0	0	2us	131.072 ms	16
1	0	1	4us	262.144 ms	32

(ns = nano-second, us = micro-second, ms = millisecond)

3. Procedure

Use the *HP* signal generator on your desk to generate a 1kHz square wave as the input signal. Use input-capture channel 0 to measure its period as per example 8.2 in your text book. Verify your results by calculating the measured period and comparing it to known input at several different frequencies.

Generate a 1 kHz square wave with a 30% duty cycle from pin PT6 as per example 8.4 in your text book. Use the oscilloscope to display and measure this waveform.

Combine the first two parts of this lab to write a program that outputs a 500 Hz square wave which is high for a period of time equal to the period of the signal present on input-capture channel 0.
Adjust the signal generator to generate waveforms with 75% and 25% duty cycles. Note which input frequencies were required in your lab report.

Show your oscilloscope display to a TA.

4. Lab report

Include your program and screen shot which show the execution results, sketch the waveform you obtained from output-compare channel.

HC12 Port Registers

```
REGBS          EQU $0000    ; DP256 register bank base address
PTT:           EQU REGBS+$240 ;portT data register
DDRT:          EQU REGBS+$242 ;portT direction register

TIOS:        EQU REGBS+$40 ;timer input/output select
CFORC:        EQU REGBS+$41 ;timer compare force
OC7M:        EQU REGBS+$42 ;timer output compare 7 mask
OC7D:        EQU REGBS+$43 ;timer output compare 7 data
TCNT:        EQU REGBS+$44 ;timer counter register hi
*TCNT:        EQU REGBS+$45 ;timer counter register lo
TSCR:        EQU REGBS+$46 ;timer system control register
TTOV:        EQU REGBS+$47 ;reserved
TCTL1:        EQU REGBS+$48 ;timer control register 1
TCTL2:        EQU REGBS+$49 ;timer control register 2
TCTL3:        EQU REGBS+$4A ;timer control register 3
TCTL4:        EQU REGBS+$4B ;timer control register 4
TMSK1:        EQU REGBS+$4C ;timer interrupt mask 1
TMSK2:        EQU REGBS+$4D ;timer interrupt mask 2
TFLG1:        EQU REGBS+$4E ;timer flags 1
TFLG2:        EQU REGBS+$4F ;timer flags 2
TC0:        EQU REGBS+$50 ;timer capture/compare register 0
TC1:        EQU REGBS+$52 ;timer capture/compare register 1
TC2:        EQU REGBS+$54 ;timer capture/compare register 2
TC3:        EQU REGBS+$56 ;timer capture/compare register 3
TC4:        EQU REGBS+$58 ;timer capture/compare register 4
TC5:        EQU REGBS+$5A ;timer capture/compare register 5
TC6:        EQU REGBS+$5C ;timer capture/compare register 6
TC7:        EQU REGBS+$5E ;timer capture/compare register 7
```

Corrected program of Output Compare mode from textbook example

```
TMSK1     equ $4C ;timer interrupt mask 1
TMSK2     equ $4D ;timer interrupt mask 2
TFLG1     equ $4E ;timer flags 1
TFLG2     equ $4F ;timer flags 2
TCNT:     EQU REGBS+$44 ;timer counter register hi
TCNT:     EQU REGBS+$45 ;timer counter register lo
TC0       equ $50 ;timer capture/compare register 0
TC6:      equ $5C ;timer capture/compare register 6
TIOS      equ $40 ;timer input/output select
portt     equ $240

high_time        equ    700
low_time         equ    300
                 org    $1000
                 movb   #$90,TSCR       ; select fast timer flag clear and
enable TCNT
                 bset   TIOS,$40        ; select OC6 function
                 movb   #$03,TMSK2      ; set prescale factor to 1us
                 movb   #$10,TCTL1      ; select toggle as the action on
pin PT6
```

```
                bset    PORTT,$40       ; set PT6 pin to high
                ldd     TCNT
                addd    #high_time
                std     TC6             ; start an OC6 with 700 delay
                movb    #$40,TFLG1      ; clear the C6F flag
high            brclr   TFLG1,$40,high  ; wait until C6F flag is set
                ldd     TC6
                addd    #low_time
                std     TC6
low             brclr   TFLG1,$40,low   ; wait until C6F flag is set
                ldd     TC6
                addd    #high_time
                std     TC6
                bra     high
                end
```