ECE 3731: **Introduction to Microprocessors**

Lab 6: Timer Subsystem -Flag Polling and Interrupts

Fall, 2017

Honor Code:

I have neither given nor received unauthorized assistance on this graded report.

X Hussein El-Souri

Contents

Objective

- Generate a tone (any frequency) on the speaker.
- Maintain a 4-bit counter on the Port B LED's
- Use an output compare channel in interrupt mode to update the value of the counter
- Update counter every second.

Equipment Used

- Codewarrior
- HCS12 Microprocessor
- Notepad++ to import code with format
- https://www.lucidchart.com/ to easily make flowcharts.

Flowchart

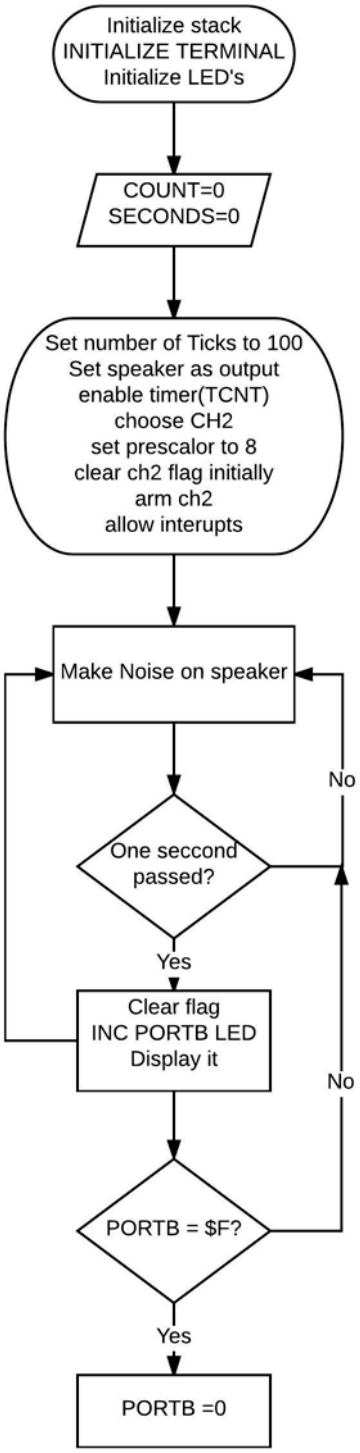

Figure 1: Flow

Procedure

The main loop makes a sound on the speaker of any frequency and if one second passes the interrupt service takes place. The interrupt service increments the value of PORTB LED and displays it. If the value reaches $F it rolls over to 0. As Mentioned in the flowchart the speaker must set as output and the timer must be enabled with a pre-scale value of 8 and appropriate flags must be cleared. Stetting the prescaler to 8 allows for 10ms per tick and total of 100 ticks allows for 1 second calculation.

Code

```
;---------------------------------------------------
; Insert your code here
;---------------------------------------------------
        lds   #ROMStart ; load stack pointer
*SET UP THE (interrupt) SERVICE & INITIALIZE
        JSR   TermInit  ; Initialize Serial Port (for simulation)
        JSR   led_enable
       CLR   COUNT
        CLR   SECONDS
        CLR   MINUTES
        CLR   HOURS
        MOVB  #100,NUMTICKS  ; number of ticks (interrupts) for 1 second
        bset  DDRT,%00100000 ; PT5 (spkr) is output
        bset  TSCR1,TEN     ; enable TCNT
        bset  TIOS,IOS2      ; choose OC2 for timer CH. 2
        movb  #$03,TSCR2     ; set prescaler to 8
        movb  #C2F,TFLG1    ; clear  C2F flag initially
        bset  TIE,C2I     ; arm OC2
        cli               ; allow interupts
; main program loop follows
LOOP
    BRCLR TFLG2, #%10000000,LOOP
    LDAA PTT
    EORA #%00100000
    STAA PTT
    MOVB  #%10000000, TFLG2
    bra LOOP
       *============= SERVICE PROCESS
OC2ISR
        MOVB   #C2F,TFLG1   ; clear flag
        LDD    TC2  ; schedule next interrupt
        ADDD   #30000  ; 30000 cycles = 10ms
        STD    TC2     ; .....
        INC    COUNT   ; one more interrupt interval counted
        LDAB   COUNT
        CMPB   NUMTICKS ; has count reached amount for 1 second?
        BNE    DONE  ; not one second yet so return
        CLR    COUNT
        JSR    ONE.SECOND  ; one second has elapsed
DONE    RTI
*============= END OF SERVICE ROUTINE
* subroutines follow this
; ONE.SECOND:
; what to do every second
ONE.SECOND
        inc PORTB
        bclr PORTB,%11110000
        INC   SECONDS
        LDAA  SECONDS
        CMPA  #60
        BEQ   ONE.MINUTE ; need to update minutes if >= 60 seconds`
        BRA   UpdateDone
UpdateDone  ; return from subrouitne
        RTS
```

Conclusion

We learned how devices (that are not connected to the internet) keep track of time using the frequency of the processor. And how to effectively uses that frequency to do events at specific time stamps.