ME3241

MICROPROCESSOR APPLICATIONS

Part 2 Assignment

Mohamed Azhar (A0110710E)

Seenivasan Sre Vinod (A0108418H)

Yao JingFu (A0125009R)

Introduction

Through this project, we would be developing an embedded system/assembly language program to generate LED light patterns according to a piece of music. We challenged ourselves by using the first 32 seconds of the fast paced Mission Impossible theme song, and generated LED patterns according to the beats of the song.

Algorithm Design and Implementation

Implementing an LED pattern algorithm essentially has 2 components. The first is setting the sequence of desired output (e.g. B'10010010'). The second crucial component is implementing appropriate time delays that would control how fast an output would remain or change.

Output: We broke down the 32s MI music into 6 segments: *Intro, Alarm music, Soft MI theme, Hard MI theme, Transition, Doubt and Ending*. We used a simple music visualizer to identify major changes in beats within each segment and came up with ideas for different LED pattern ideas.

Time Delay: With the music visualizer, we identified how long each beat lasts. These would form the basis of the time delays we would implement in our code. We used both nop and Timer0 to show our understanding of different approaches to implement delay subroutines .

Algorithm

	LED Output	Time delay	Repetition
Intro (0-3 sec)	a)00000000	3s	Nil
Alarm (3-6 sec)	b)10101010	0.1875s	One loop - b,c Loops 7 times
	c)01010101	0.1875s	
Soft MI Theme (6 to 12 sec)	d)11100111	0.1875s	One loop - d,e,d,e,f,g,f,g Loops 4 times
	e)11111111	0.1875s	
	f)11000011	0.1875s	
	g)11111111	0.1875s	
High MI Theme (12-18 sec)	h)11101111	0.15s	One loop - h,i,h,i,j,k,j,k Loops 4 times
	i)111111111	0.15s	
	j)00000000	0.15s	
	k)11111111	0.15s	
Transition (18-24 sec)	l)01111111	0.15s	One loop - i,m,n,o,p,q,r,s Loops 4 times
	m)10111111	0.15s	
	n)11011111	0.15s	
	o)11101111	0.15s	

	p)11110111	0.15s	
	q)11111011	0.15s	
	r)11111101	0.15s	
	s)11111110	0.15s	
Doubt (24-31 sec)	t)01111111	0.3s	One loop - t,u,v loops 3 times
	u)10011111	0.3s	
	v)11100000	2.15s	
Ending (31-32 sec)	w)00000000	0.15s	NIL
	x)11111111	0.15s	
	y)00000000	0.15s	
	z)11111111	0.15s	

Sample calculations

Delay with nop for 1 sec
No of loop counts x 20 x 4 x (1x10^-6) = 1 sec
No of loop counts = 12500
= 50 x 250

Delay for 0.15 sec = Loop 3 times with delay using timer0 for 0.05 sec
No.of clock cycle x 16 x 4 x (1x10^-6) = 0.05 sec
No.of clock cycle = 781 (nearest whole no.)
Value to load into TMR0 = 65535-781
= 64754
= FCF2 in hex

Results

Video link:https://drive.google.com/open?id=0ByhLawVzr8t5TXB2T3E5YVVlbk0
Music track: https://www.youtube.com/watch?v=XAYhNHhxN0A (first 32s)

Discussions including Problems Encountered

Problem	Description	Solution
Hardware	Microprocessor was not responsive	We consulted with the professor and found that the hardware was faulty.
Implementation	Delay using nop was too long than expected	We consulted with the professor and learnt that we are using a wrong frequency of 40MHz in our calculation instead of 1 MHz of the PIC controller.
	Time delay calculated did not sync with music beat	Used trial and error to get the best synchronization

Source Code

```
#include <p18F4520.inc>
            radix             dec             ;  input variables will be in decimal unless stated

lp_cnt1    equu             0x21            ; Assign file register 0x21 to lp_cnt1
lp_cnt2    equ              0x22            ; Assign file register 0x22 to lp_cnt2
lp_cnt3    equ              0x23            ; Assign file register 0x23 to lp_cnt3
lp_cnt4    equ              0x24            ; Assign file register 0x24 to lp_cnt4

dup_nop macro kk      ; duplicate nop instruction for kk times

          variable i
i = 0
          while i<kk
          nop
i=i+1
          endw
          endm

          org               0x00
          goto    start
          org               0x08
          retfie
          org               0x18
          retfie

start
          movlw    B'00000000'
          movwf    TRISD          ;  set PORTD as output
          movlw    B'11111111'   ;   All lights off
          movwf    PORTD
          call        delay_3s      ; LED off for 3sec to prepare us to sync music when LED lights up
          movlw    B'00000000' ; Light up all LED - trigger to play music
          movwf    PORTD
          call         delay_3s     ; LED stay lit for 3s

          movlw    D'7' ;
          movwf    0x30, A        ; assign decimal 7 to file register 0x30


loopAlarm3sec                    ; LEDs alternate 7 times over 3s
          movlw    B'10101010'; set LED 7,5,3,1lit up
          movwf    PORTD
          call        delay_01875s         ; delay for 0.1875sec
          movlw    B'01010101' LED 8,6,4,2 lit up
          movwf    PORTD
          call    delay_01875s   ; delay for 0.1875sec
          decfsz    0x30,F,A      ; decrement of 0x30 (value of 7) and skip if equal 0
          bra       loopAlarm3sec ; branch to loopAlarm3sec
```

```
            movlw    D'4'
            movwf    0x31,A ;move 4 to file register 0x31

loopSoftMI          ; next soft MI tune for 6s
            movlw    B'11100111' ; LED 4,3 lit up
            movwf    PORTD
            call     delay_01875s ; delay 0.1875 sec
            movlw    B'11111111' ; all LED off
            movwf    PORTD
            call    delay_01875s
            movlw    B'11100111' ; LED 5,4 lit up
            movwf    PORTD
            call     delay_01875s
            movlw    B'11111111' ; all LED off
            movwf    PORTD
            call    delay_01875s
            movlw    B'11000011' ; LED 6,5,4,3 lit up
            movwf    PORTD
            call    delay_01875s
            movlw    B'11111111'  ; all LED off
            movwf    PORTD
            call     delay_01875s
            movlw    B'11000011'
            movwf    PORTD
            call     delay_01875s
            movlw    B'11111111'
            movwf    PORTD
            call     delay_01875s
            decfsz   0x31,F,A ; decrement 0x31 (4) and skip if equal to 0
                              ; pattern loops 4 times
            bra      loopSoftMI ; branch to loopSoftMI


           movlw    D'6'
           movwf    0x32,A ; assign decimal 6 to 0x32
                            ; next soft MI tune for 6s
loopHardMI
            movlw    B'11101111' ; LED 4 lit ip
            movwf    PORTD
            call     delay_015s ; delay for 0.15 sec
            movlw    B'11111111' ;All LED lit up
            movwf    PORTD
            call    delay_015s
            movlw    B'11101111'
            movwf    PORTD
            call   delay_015s
            movlw    B'11111111'
            movwf    PORTD
            call     delay_015s
            movlw    B'00000000' ; All LED lit up
            movwf    PORTD
            call   delay_015s ; delay 0.15sec
            movlw    B'11111111' ; All LED off
            movwf    PORTD
            call        delay_015s
            movlw    B'00000000'
            movwf    PORTD
            call        delay_015s
            movlw    B'11111111'
            movwf    PORTD
            call        delay_015s
            decfsz    0x32,F,A ; decrement of 0x32 (6) and skip if equal to 0
                                        ; pattern loops 6 times
            bra        loopHardMI    ; branch to loopHardMI


            movlw    D'4'
            movwf    0x33,A ; Assign decimal 4 into file register 0x33

loopTransition
            movlw    B'01111111'  ; LED 8 lit up
            movwf    PORTD
            call        delay_015s  ; delay 0.15sec
            movlw    B'10111111'  ;LED 7 lit up
            movwf    PORTD
            call        delay_015s
            movlw    B'11011111'  ; LED 6 lit up
            movwf    PORTD
            call        delay_015s
            movlw    B'11101111'  ; LED 5 lit up
            movwf    PORTD
            call        delay_015s
            movlw    B'11110111'  ; LED 4 lit up
            movwf    PORTD
            call        delay_015s
            movlw    B'11111011'  ; LED 3 lit up
            movwf    PORTD
            call        delay_015s
            movlw    B'11111101'  ; LED 2 lit up
            movwf    PORTD
            call        delay_015s
            movlw    B'11111110'   ;LED 1 lit up
            movwf    PORTD
            call        delay_015s
            decfsz    0x33,F,A        ;decrement of 0x33 (4) and skip if equal to 0
                                          ; LED pattern loops 4 times
            bra        loopTransition
            movlw    D'3'
            movwf    0x34           ; assign decimal 3 to to 0x34

loopDoubt
             movlw    B'01111111' ; LED 8 lit up
             movwf    PORTD
             call    delay_015s         ; delay 0.15sec
             movlw    B'10011111'           ; LED 7,6 lit up
             movwf    PORTD
            call    delay_015s
            call    delay_015s      ;delay 0.15+0.15 = 0.3 sec
            movlw    B'11100000'; LED 5,4,3,2,1
            movwf    PORTD
            call    delay_1s
            call    delay_1s
            call    delay_015s      ; delay 1+1+0.15 = 2.15 sec
            decfsz    0x34,F,A      ;decrement of 0x34 (3) and skip if equal to 0
            bra       loopDoubt      ; branch to loopDoubt

            movlw    B'11111111'; all LED off
            movwf    PORTD
            call         delay_015s  ; delay 0.15sec
            movlw    B'00000000'; all LED lit
            movwf    PORTD
            call         delay_015s
            movlw    B'11111111'
            movwf    PORTD
            call         delay_015s
            movlw    B'00000000'
            movwf    PORTD
            call         delay_3s      ;delay 3sec
forever  movlw    B'11111111' ; all LED off forever
            movwf    PORTD
            bra forever


                                         ; delay subroutines

Delay_3s                              ; 3sec delay subroutine
            movlw    175
            movwf    lp_cnt1,A     ;assign 175 to lp_cnt1
loop1    movlw    250
            movwf    lp_cnt2,A     ;assign 250 to lp_cnt2
loop2    dup_nop 17               ;17 instruction cycles
            decfsz    lp_cnt2,F,A  ; decrement of lp_cnt2 ( 1 cycle) and skip if equal 0 (2 cycle)
            bra       loop2              ; branch to loop2
            decfsz    lp_cnt1,F,A  ; decrement of lp_cnt1 (1 cycle) and skip if equal 0 (2 cycle)
            bra       loop1              ; branch to loop1
            return

Delay_1s ; 1sec delay subroutine
            movlw    50
            movwf    lp_cnt3,A     : assign 50 to lp_cnt3
loop3    movlw    250
            movwf    lp_cnt4,A     ;assign 250 to lp_cnt4
loop4    dup_nop 17
            decfsz    lp_cnt4,F,A  ;decrement of lp_cnt4 ( 1 cycle) and skip if equal 0 (2 cycle)
            bra       loop4              ;branch to loop4
            decfsz    lp_cnt3,F,A  ; decrement of lp_cnt3 ( 1 cycle) and skip if equal 0 (2 cycle)
```

```
            bra       loop3              ;branch to loop3
            return

delay_01875s                             ; 0.1875sec delay subroutine
            movlw D'15'
            movwf PRODL          ;assign decimal 15 to PRODL
delay                            ;delay of 0.0125sec
            movlw 0x83           ; enable TMR0, select internal clock
            movwf T0CON,A        ; set prescaler to 16
loopd       movlw 0xFF
            movwf TMR0H          ;assign hex FF to TMR0H
            movlw 0x3C
            movwf TMR0L,A         ; assign hex 3C to TMR0L
            bcf INTCON,TMR0IF,A          ; clear the TMR0IF flag
wait        btfss INTCON,TMR0IF,A
                 bra wait                   ; wait until 50 ms is over
                 decfsz PRODL,F,A     ; decrement of PRODL and skip if equal to 0
                 bra loopd                  ;  branch to loopd
                 return

delay_015s                               ; 0.15sec delay subroutine
           movlw D'3'
            movwf PRODL          ; assign decimal 3 to PRODL
delay_50ms ; delay for 0.05 sec
            movlw 0x83            ; enable TMR0, select internal clock
            movwf T0CON,A        ; set prescaler to 16
loopd1      movlw 0xFC
            movwf TMR0H           ; assign hex FC to TMR0H
            movlw 0xF2
            movwf TMR0L,A                ; assign hex F2 to TMR0L
            bcf INTCON,TMR0IF,A          ; clear the TMR0IF flag
wait1       btfss INTCON,TMR0IF,A ;
                 bra wait1                  ; wait until 50 ms is over
                 decfsz PRODL,F,A     ;decrement of PRODL and skip if equal to 0
                 bra loopd1                  ; branch to loopd1
                 return


            end
```