APPLICATION OPERATION

The first application of my Logic Switch is to help prototypes of digital circuit. There are two main functions:

1 – Logic Level Generator
The momentary switch acts as a debounced switch. Each time you press the switch the outputs toggle. There are also two led indicators to show the state of the outputs Q and 'Q.

2 – Clock Generator
To enter or exit this mode you need to press the switch at least 2 seconds. The first time you enter this mode the outputs oscillate at 1 Hz. Pressing again will change frequency to 10 Hz (LED L5 On). Pressing a second time for 100 Hz (LED L2 On) and a third time for 1 KHz (LED L1 On). Pressing the switch again will reset frequency to 1 Hz. Note that duty cycle is 50% for all frequencies.

BLOCK DIAGRAM

PIC508-04/P

Switch (Sw1)

Q ('Q LED)

GP3 GP1 Q (Q LED)

GP0 GP5 L1 (1 KHz)

GP4 L2 (100 Hz)

GP2 L5 (10 Hz)

MICROCHIP HARDWARE
DEVELOPMENT TOOLS USED

Assemble/Complier version:
MPlab 3.22, MPasm 1.5

BILL OF MATERIAL

<table>
<thead>
<tr>
<th>Part</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>Ceramic Capacitor 0.1uF 50v</td>
</tr>
<tr>
<td>R1-R2</td>
<td>680 Ohms resistor 1/4 watt</td>
</tr>
<tr>
<td>R4-R6</td>
<td>4.7 K Ohms resistor 1/4 watt</td>
</tr>
<tr>
<td>L1-L2-L5</td>
<td>Rectangular LED (2mm x 5mm)</td>
</tr>
<tr>
<td>L3-L4</td>
<td>Round LED T1 3/4</td>
</tr>
<tr>
<td>SW1</td>
<td>Momentary Switch E-Switch #520-03-1</td>
</tr>
<tr>
<td>U1</td>
<td>PIC12C508-04/P</td>
</tr>
</tbody>
</table>
APPENDIX A: SOURCE CODE

;***************************************************************;
;   Marc Lemay
;   Electro Technician
;   221 St-Isidore
;   St-Etienne-des-Gres
;   Quebec, Canada
;   G0X 2P0
;   Tel: (819) 535-4117
;   Project: Logic Switch
;   date   : august 23  1997
;***************************************************************

list p=12c508, f=inhx8m ;uC number ;and inhx8m output format file

decfreq    equ 0x07  ;low register use for frequency generation
decfreqh   equ 0x0e  ;high register use for frequency generation
freqscale   equ 0x08  ;low prescale value for frequency generation
freqscaleh  equ 0x0f  ;high prescale value for frequency generation
;1 =  1 Khz
;10 =100 Hz
;100= 10 Hz
;1000=  1 Hz  3e8  hexa

timeswp         equ 0x09  ;switch time pressed
                 ;each unit = 16 msec
swcourte        equ 0x0a  ;1 = switch pressed detected
swprescale      equ 0x0b  ;switch prescale 0 to 31 = 16 msec
funct          equ 0x0c  ;0 = 0 logic output (stable)
                 ;1 = 1 logic output (stable)
                 ;2 = output = freq 1 Hz
                 ;3 = output = freq 10 Hz
                 ;4 = output = freq 100 Hz
                 ;5 = output = freq 1 KHz
functtmp       equ 0x0d  ;temp register
gpio           equ 0x06  ;adr5  io
tmr0           equ 0x01 ;adr5  timer
status         equ 0x03  ;status register adr5
osccal         equ 0x05 ;oscillator calibration register

org 0000
begin init

movwf osccal  ;save oscillator calibration value

movlw 0x08    ;gp0 gp1 gp2 gp4 gp5 output gp3 input
tris 6      

movlw 0xd1    ;prescale =1/4 for RC internal

clrf decfreqh;
clrf decfreq;
clrf freqscaleh;0 = freq. stop
clrf freqscaleh;0 = freq. stop
clrf timeswp ;time switch pressed = 0
clrf swcourte
clrf funct   ;function = 0 = ouput = 0

END
movlw 0x1f ;31 decimal
movwf swprescale;init prescale switch pressed to 31
movlw 0x36 ;logic 0 on gpi0 and 1 on gpi1
          ;gp2, gp4 et gp5 at 1 log -> led freq. off
movwf gpio
movlw 0x05 ;init value for timer0
movwf tmr0
;end init

;begin master prog
princ btfss tmr0,07 ;test bit 7 of tmr0
    goto princ ;wait until tmr0 = 128
;each 500 Usec the program will go here
movlw 0x05 ;init value of tmr0 to 5 to give a good 500 usec
          ;for time base
movwf tmr0
movf freqscale,0;mov freqscale in w
btfss status,2;check the z bit in status
    goto freqact ;if freqscale <> 0 the freq. is running
;here the freq. is off and a logic 0 or 1 is steady to the output
suite btfss gpio,3 ;test switch pressed=1 log open=0 log.
    goto open ;
;goto here if switch pressed --- gp3 = 1 log.
decfsz swprescale,1;var decrement from 31 to 0
    goto princ ;if <> 0 we return to princ
;here the switch is pressed since 16 msec
movlw 0x1f ;value 31 decimal
movwf swprescale;re-init swprescale to 31
incf timeswp ;inc value time switch pressed
btfsc timeswp,2;test if timeswp = 4 (64 msec pressed ?)
    goto rendu4 ;branch if timeswp = 4 (from 4 to 7 it’s ok)
;here we check the 2 second switch pressed
movlw 0x80 ;w <-- 128 for 2 sec.
subwf timeswp,0;compare w and timeswp
btfss status,2;skip if z bit = 1
    goto princ ;if z bit = 0 the switch not pressed for 2 second
;here the switch is pressed since 2 second
movf freqscale,1;move to affect the z flag
btfss status,2;skip if z flag is 1
    goto tologic;if z is 0 then freqscale<>0 then freq is running
;here the logic mode is on... we stop it to make a freq. running
bcf gpio,5 ;led 1 KHz on
movlw 0x05 ;5 = 1 KHz
movwf funct ;funct = 5 = 1 KHz
movlw 0xc8 ;c8 = 200 dec. prescale before output toggle
movwf freqscale
movwf decfreq
clr freqscaleh
clr decfreqh
scansw
btfsc gpio,3 ;test if switch open
    goto scansw ;scan while sw not open
movlw 0x1f ;31 decimal
movwf swprescale;re-init prescale
clr timeswp ;time switch pressed
    goto princ ;return to princ to scan tmr0
to logic

; here the freq. will be stopped and
; a steady logic 0 will be at the output
clr funct; function = 0
clr freqscale; deactive the frequency
clr freqscaleh;
movlw 0x36 ; output gpi0 = 0 gpi1 (inverse) = 1
; and frequency led OFF
goto princ; return to master program to scan tmr0

rendu4 ; branch here when the switch is pressed for 64 msec
movlw 0x01 ;
movwf swcourte; init variable swcourte to 1 (the switch is good)
goto princ; return to master program to scan tmr0

;------------------------------------------------------------------------------
open ; jump here when the switch is not pressed
movlw 0x1f ; 31 decimal
movwf swprescale; re-init prescale for switch
clr timeswp ; reset timeswp because switch is open
movf swcourte,1; check if swcourte = 0
btfsc status,2; test le z flag si 0 logique
goto princ; branch if z flag equal 1 log.

; here swcourte = 1 log. then switch good
clr swcourte; reset swcourte to 0
movf funct,0 ; move funct in w register
movwf functtmp; put a copy of funct in functtmp
btfsc status,2; check the z flag
goto funct1; if z = 1 then branch to funct1
decf functtmp,1
btfsc status,2; check z flag
goto funct0; if z = 1 then branch to funct0
decf functtmp,1
btfsc status,2; check z flag
goto funct3; if z = 1 then branch to funct3
decf functtmp,1
btfsc status,2; check z flag
goto funct4; if z = 1 then branch to funct4
decf functtmp,1
btfsc status,2; check z flag
goto funct5; if z = 1 then branch to funct5

; here the function if 5

movlw 0x02
movwf funct; function = 2 now
movlw 0xe8 ; 3e8 = 1000 dec
movwf freqscale
movwf decfreq
movlw 0x03
movwf freqscaleh
movwf decfreqh;
bsf gpio,5 ; led 500-1000 Hz off. no led for 1 Hz rate
goto princ; return to main program to scan tmr0

funct1 ; here the active function will be 1
incf funct,1 ; funct = 1
movlw 0x35 ; output = 1 and leds off
movwf gpio;
goto princ; return to main program to scan tmr0

funct0 ; here the active function will be 0
clrf        funct ; funct = 0
movlw      0x36 ; output q = 0
movwf      gpio ;
goto       princ ; return to main program to scan tmr0

funct3 ; here the active function will be 3   10 Hz
incf       funct ; increment funct --> 3
movlw      0x64 ; 64 = 100 dec
movwf      freqscale
movwf      decfreq
clrf        freqscaleh
clrf        decfreqh ; reset high byte
bcf         gpio,2 ; led 10 Hz on
movwf       decfreq ; re-init decfreq for next toggle
movwf       freqscaleh
movwf       gpio ; clock out
xorlw      0x03 ; 2 last bits to toggle
movwf      gpio
movwf      freqscale ; freqscale --> w
movwf      decfreq ; re-init decfreq for next toggle
movwf      freqscaleh,0
movwf      decfreqh ; init byte high
goto        princ ; return to main program to scan tmr0

funct4 ; here the active function will be 4   100Hz
incf       funct ; increment funct --> 4
movlw      0x0a ; 0a = 10 dec
movwf      freqscale
movwf      decfreq
clrf        freqscaleh
clrf        decfreqh ; reset high byte
bsf         gpio,2 ; led 10 Hz off
bcf         gpio,4 ; led 100 Hz on
movwf       decfreq ; re-init decfreq for next toggle
movwf       freqscaleh
movwf       gpio
xorlw      0x03 ; 2 last bits to toggle
movwf      gpio
movwf      freqscale
movwf      decfreq
movwf      freqscaleh
movwf      decfreqh
bcf         gpio,4 ; led 100 Hz off
bcf         gpio,5 ; led 1000 Hz on
goto        princ ; return to main program to scan tmr0

funct5 ; here the active function will be 5   1000 Hz
incf       funct ; increment funct --> 5
movlw      0x01 ; 01 = 1 dec = 1 KHz
movwf      freqscale
movwf      decfreq
clrf        freqscaleh
clrf        decfreqh ; reset high byte
bsf         gpio,4 ; led 100 Hz off
bcf         gpio,5 ; led 1000 Hz on
movwf       decfreq ; re-init decfreq for next toggle
movwf       freqscaleh
movwf       gpio
xorlw      0x03 ; 2 last bits to toggle
movwf      gpio
movwf      freqscale
movwf      decfreq
movwf      freqscaleh
movwf      decfreqh
bcf         gpio,4 ; led 100 Hz off
bcf         gpio,5 ; led 1000 Hz on
goto        princ ; return to main program to scan tmr0

;----------------------------------------------------------------------

freqact ; here frequency mode is active
decf        decfreq,1 ; decrement decfreq
btfss       status,2 ; test z flag
goto        suite ; branch if z <> 0
movf        decfreqh,0 ; check high byte if = 0
btfss       status,2 ; check z flag
goto        decfreqhigh ; branch if > 0

; here we toggle the output
movf        gpio,0 ; load w with gpio : clock out
xorlw      0x03 ; 2 last bits to toggle
movwf      gpio
movwf      freqscale,0 ; freqscale --> w
movwf      decfreq ; re-init decfreq for next toggle
movwf      freqscaleh,0
movwf      decfreqh ; init byte high
goto        suite

decfreqhigh
deef        decfreqh,1 ; decrement byte high
goto        suite

end