; Program for converting 16 trigger inputs (active low) to Midi Note-On messages. The 16 triggers are polled one at a time. If a trigger input is high and its current state register shows a low, send Midi Note-On. If the trigger input is low and its current state register shows a high, send Midi Note-On with velocity zero to turn the note off.

; RA0-RA3 outputs, 16 trigger select addresses
; RA4, input, trigger 32/10
; RB0-RB3 inputs, Midi Channel number
; RB7, output, Midi serial output

; Midi Note-On Bytes - 0x9(channel), 0(key#), 0(velocity)

;-------------------------------------------------------------------------------
;                  list     p=16c84
;                   radix    hex
;-------------------------------------------------------------------------------
; CPU EQUATES
;-------------------------------------------------------------------------------
portb   equ  0x06
porta   equ  0x05
w       equ  0
f       equ  1
fselect equ 0x04
fpoint  equ 0x00
status  equ 0x03
count   equ 0x0c  ; trigger number
nstat   equ 0x0d  ; midi note-on status byte with channel#
bitcnt  equ 0x0e  ; counter used in midi send subroutine
send    equ 0x0f  ; folds midi byte in midi send subroutine

;-------------------------------------------------------------------------------
; PROGRAM EQUATES
;-------------------------------------------------------------------------------
noteon  equ 0x90  ; midi status for note-on
vel     equ 0x64  ; velocity value used in midi note-on
baseky  equ 0x20  ; lowest of 16 key numbers used in midi note-on
and associated with each of 16 trigger inputs
basef   equ 0x10  ; lowest of 16 registers holding the current
state (note-on/note-off) of 16 triggers

;-------------------------------------------------------------------------------
; PROGRAM INITIALIZATION
;-------------------------------------------------------------------------------
org     0x00
init    movlw 0x00
        option  ; option register
        movlw 0x00
        movwf 0x0b  ; interrupt register, disable all
        movlw 0x10
        tris    porta ; porta, bits0-3 outputs, bit4 input
        movlw 0x0f
        tris    portb ; portb, bits0-3 inputs, bits4-7 outputs
        bsf    portb,7 ; initialize midi out line high
movlw 0x0f
movwf count ;start the counter at 16
loop to clear trigger current state registers 0x10 to ox1f
loop
movf count,w ;move current count to w
addlw basef ;calculate number of next register to clear
movwf fselect ;mov register number to file select reg
clf fpoint ;clear the register pointed to
decf count,f ;decrement count register
movlw 0xff
subwf count,w ;count minus ff, to test for end of count=ff
btfss status,2 ;test the 2 bit in status register
go loop ;continue loop if count not finished

loop to clear trigger current state registers 0x10 to ox1f
loop
movf count,w ;move current count to w
addlw basef ;calculate number of next register to clear
movwf fselect ;mov register number to file select reg
clf fpoint ;clear the register pointed to
decf count,f ;decrement count register
movlw 0xff
subwf count,w ;count minus ff, to test for end of count=ff
btfss status,2 ;test the 2 bit in status register
go loop ;continue loop if count not finished

;-MAIN PROGRAM LOOP

decrement the counter, modulo 16
nloop decaf ;decrement counter (if 00, then it goes to ff)
movfw count
btfsc count,7 ;test for counter high bit (ff)
andlw 0x0f ;if high bit high, then clear high nibble (0f)
movwf count ;new count value

load trigger address outputs for polling next trigger input
movwf porta ;low nibble addresses one of 16 trigger inputs

let trigger address settle while setting up Midi Note-On status byte
clrwdt ;reset the watchdog timer
movfw portb ;get midi channel number from portb low nibble
andlw 0x0f ;put midi channel number in low nibble
iorlw noteon ;put midi note-on status in high nibble
movwf nstat ;store midi note-on status byte in register nstat

Read trigger input.  If high, branch to trgon.  If low, branch to trgoff.
btfss porta,4 ;read trigger input from porta bit4
go to trgoff ;if high continue to trgon, else goto trgoff

TRIGGER ON ROUTINE

trgon movfw count
addlw basef
movwf fselect ;address trigger's current state register
btfss fpoint,0  ;read current state of trigger in LSB
goto mloop    ;if note is already on, continue to next trigger
movlw 0xff    ;else send Midi Note-On Message
movwf fpoint
movwf nstat
call midi     ;send Midi Note-On status byte
movfw count
addlw baseky
call midi     ;send Midi Note-On key data
movlw vel
movfvel
addlw baseky
call midi     ;send Midi Note-On velocity data
movlw vel

; TRIGGER OFF ROUTINE

trgoff movfw count
addlw basef
movwf fselect ;address trigger's current state register
btfss fpoint,0 ;read current state of trigger in LSB
goto mloop    ;if note is already off, continue to next trigger
clrw           ;else send Midi Note-On Message
movwf fpoint
movfw nstat
call midi     ;send Midi Note-On status byte
movfw count
addlw baseky
call midi     ;send Midi Note-On key data
clrw
call midi     ;send Note-On velocity=0 for a Note-Off
movlw 0

; MIDI SEND SUBROUTINE

midi movwf send ;load byte into send register
bcf portb,7    ;send low midi start bit out portb bit7
movlw 0x09
movwf bitcnt    ;set up the bit counter to shift out 8 bits
contin
nop

;midicontin
nop
nop
nop
nop
nop
nop
decf bitcnt,f   ;decrement counter
btfsc status,2   ;test Z for bitcnt=0
goto endbit     ;if zero go to send endbit
rrf send,f       ;shift next bit into status carry
btfss status,0   ;check carry
goto send0
goto send1
send0 nop
bcf portb,7 ;if carry is low, send midi low
goto contin
send1 bsf portb,7 ;else send midi high
goto contin
endi nop
nop
nop
bsf portb,7 ;send high midi stop bit
movlw 0x04 ;delay before sending next midi byte
movwf bitcnt
decfsz bitcnt,f
goto delay

return ;return from MIDI Send subroutine

;------------------------------------------------------------------------------------------------------------------
;THE END
;------------------------------------------------------------------------------------------------------------------