```
;Include the PIC file and Configure the system
list       P=PIC12F752
#include "p12F752.inc"
__config (_CP_OFF & _WDT_OFF & _BOREN_EN & _PWRTE_ON & _FOSC_INT & _MCLRE_OFF &
_CKLOUTEN_OFF)

;VARIABLE DEFINITIONS
;Pound Defines Pins
#define   COLOR_OUTPUT_PIN                2                              ;PIN 2
#define   PAIRED_OUTPUT_PIN               1                              ;PIN 1
#define   TIME_OUTPUT_PIN                         0                              ;PIN 0
#define   COLOR_INPUT_PIN                 3                              ;PIN 3 - If set we are
blue
#define   PAIRED_INPUT_PIN                4                              ;PIN 4 - If set we
should set the servo to be in the paired position
#define   TIME_INPUT_PIN                  5                              ;PIN 5 - Every pulse
indicates we should tick the time servo
#define           TESTING_OUTPUT_PIN              5

;Temporary Variables for File Registers
W_TEMP                      equ             0x71
STATUS_TEMP         equ             0x72
PCLATH_TEMP                 equ             0x73

;Additional File Regsiters
PORT_A_READ         equ             0x74

;Depending on which bits are set in PWM_Status, we know what point we are in our PWM
PWM_STATUS          equ             0x75
#define                     PAIRED_PWM_BIT          1
#define                     TIME_PWM_BIT            2

;Compare Values
T2_COMPARE_PAIR                     equ     0x76
T2_COMPARE_TIME                     equ     0x77
T2_COMPARE_INTERMEDIATE             equ     0x78

TIMER_COUNTER                       equ     0x79

Timer_Counter_Reset                 equ             9

;Timer Constants - Note these need to actually be determined
T2_Prescalar                equ b'00000010'         ;Prescaler of 16
T2_Postscalar               equ     b'00001000'         ;Postscaler of 2
```

```
Unpaired_Period                 equ d'10'                       ;Match value
of 50
Paired_Period                   equ d'5'                        ;Match value of 50

Intermediate_Period             equ d'200'

;Org
;Set org to start at 0 and go to port initialization but goto ISR aterwards
        org             0
        goto    InitPort
        org             4
        goto    ISR
        org             5


;Pin Values Table that the motor will step through
Timer_Periods:
        addwf   PCL, f
  retlw         d'4'
        retlw   d'5'
  retlw         d'6'
  retlw         d'8'
  retlw         d'9'            ;45 second position
        retlw   d'11'
        retlw   d'12'
        retlw   d'14'
        retlw   d'15'
        retlw   d'16'           ;0 Position
EndTable: ;use this label to verify that the end is less than 0xff


;Initialize the PORT
InitPort:
        ;Select PORT A (ANSELA) and clear it to initialize
        banksel         ANSELA
        clrf    ANSELA

        ;Select TRISA and clear it which will set everything to an output
        banksel         TRISA
        clrf    TRISA

        ;Set the bits of all of our inputs
        bsf             TRISA, COLOR_INPUT_PIN
```

```
        bsf             TRISA, PAIRED_INPUT_PIN
        bsf             TRISA, TIME_INPUT_PIN

        BANKSEL     OPTION_REG
        BCF         OPTION_REG, NOT_GPPU

        ;Clear the Actual Port so nothing is Turned On
        banksel             LATA
        clrf        LATA

InitVariables:
        clrf        PWM_STATUS
        bsf             PWM_STATUS, PAIRED_PWM_BIT

        ;Reset The Timer Counter
        movlw       Timer_Counter_Reset
        movwf       TIMER_COUNTER

        ;Call the Table to get the last value
        movf        TIMER_COUNTER, w
        call        Timer_Periods

        ;Set the compare time eqeual to the last value of the table
        movwf       T2_COMPARE_TIME

        banksel             LATA
        bcf             LATA, COLOR_OUTPUT_PIN

;Initialize the Interrupts
InitInterrupt:
        ;initialize the Capture Interrupt
        banksel             IOCAN                       ;select the register for
capture interrupts (IOCAN)
        bsf             IOCAN, TIME_INPUT_PIN   ;Set the bit for the IOCAN register
for the receiving pin (this implies negative edge)

        ;Initialize the Timer Interrupt for Timer 2
        banksel             T2CON                       ;Bankselect Timer 2
        clrf        T2CON                   ;Clear
        movlw       T2_Prescalar        ;Set the prescaler
        iorwf       T2CON, f
        movlw       T2_Postscalar       ;set the postscaler
        iorwf       T2CON, f
```

```
        banksel         PR2                                     ;make sure we are in
PR_2
        movlw       T2_COMPARE_PAIR             ;set the compare value
        movwf       PR2                                 ;move working register back
to PR_2

        bsf             T2CON, TMR2ON           ;Turn on Timer 2

        ;Enable/Disable Interrupts
        banksel         INTCON
        bsf             INTCON, PEIE  ;Enable Peripheral Interrupt
        bsf             INTCON, GPIE  ;Enable Interrupt on Change (IOCIE --> GPIE)
        bsf             INTCON, GIE             ;Enable Global Interrupt

        banksel         PIE1
        bsf             PIE1, TMR2IE  ;Enable Timer2 Interrupt

;Event Checker Loop to Idle in Unless we Receive and Interrupt
EventChecker:
        ;Read in the Most Recent Port A
        banksel         PORTA
        movfw       PORTA
        movwf       PORT_A_READ

        ;Check for Paired Byte
        btfss       PORT_A_READ, PAIRED_INPUT_PIN  ;if the Paired Bit is clear then we can
go back to simply waiting for an interupt/pairing as nothing will have changed
        goto        PairBitCleared
        call        PairBitSet

        ;Check for Color
        btfss       PORT_A_READ, COLOR_INPUT_PIN        ;if the Paired Bit is clear then
we can go back to simply waiting for an interupt/pairing as nothing will have changed
        goto        SetColorRed
        goto        SetColorBlue


;Go to this function when we discover the pair bit was clear
PairBitCleared:
        ;Set the Period of the Pair Compare value to be the paired period
        movlw       Unpaired_Period
        movwf       T2_COMPARE_PAIR

        ;Reset the Timing Servo so that it is at the intial counter of 10
```

```
        movlw       Timer_Counter_Reset
        movwf       TIMER_COUNTER

        ;Call the Table to get the last value
        movf        TIMER_COUNTER, w
        call        Timer_Periods

        ;Set the compare time eqeual to the last value of the table
        movwf       T2_COMPARE_TIME

        goto        EventChecker        ;loop back to event checker / waiting for interrupt
state

;Call this Function When we discover the pair bit was set
PairBitSet:
        ;Set the Period of the Pair Compare value to be the paired period
        movlw       Paired_Period
        movwf       T2_COMPARE_PAIR

        return;

;Set the Color to Blue
SetColorBlue:
        banksel             LATA
        bsf                 LATA, COLOR_OUTPUT_PIN
        goto        EventChecker        ;loop back to event checker / waiting for interrupt
state

;Set the Color to Red
SetColorRed:
        banksel             LATA
        bcf                 LATA, COLOR_OUTPUT_PIN
        goto        EventChecker        ;loop back to event checker / waiting for interrupt
state


;interrupt
ISR:
        call        Push                            ;Call Push

        banksel             INTCON                              ;Disable Global
Interrupts
        bcf                 INTCON, GIE
```

```
        ;if the interrupt was caused by timer 2 then go to timer 2 action
        banksel         PIR1
        btfsc           PIR1, TMR2IF            ;check if timer 2 flag arises
        goto            Timer2Interrupt         ;go to timer 2 interrupt

;Capture Interrupt
CaptureInterrupt:
        banksel         IOCAF
        bcf             IOCAF, TIME_INPUT_PIN   ;Clear Capture Flag
        banksel         INTCON
        bcf             INTCON, RAIF            ;Clear Capture Flag     in INTCON
(IOCIF i.e. RAIF)

        ;Decrement the Timer Counter
        decf            TIMER_COUNTER
        movf            TIMER_COUNTER, w

        ;Call the Table to get the last value
        call            Timer_Periods

        ;Set the compare time eqeual to the last value of the table
        movwf           T2_COMPARE_TIME

        goto            Pop                     ;Goto Pop

;Timer2Interrupt i.e. time to PWM
Timer2Interrupt:
        banksel         PIR1
        bcf             PIR1, TMR2IF            ;Clear Timer Interrupt Flag

        ;First Check which stage we are in
        banksel         PWM_STATUS
        btfsc           PWM_STATUS, PAIRED_PWM_BIT  ;if the pair bit is clear then skip
        goto            PairedPWM

        btfsc           PWM_STATUS, TIME_PWM_BIT    ;if the time bit is clear then skip
        goto            TimePWM

        goto            IntermediatePWM


;Set the Paired PWM output high
PairedPWM:
        ;Set the TIME PWM flag so that we go to the Time PWM Next
```

```
        clrf        PWM_STATUS
        bcf             PWM_STATUS, PAIRED_PWM_BIT
        bsf             PWM_STATUS, TIME_PWM_BIT

        ;Set the Paired PWM Pin
        banksel         LATA
        bsf             LATA, PAIRED_OUTPUT_PIN
        bcf             LATA, TIME_OUTPUT_PIN

        ;Reset Compare value
        banksel         PR2                                     ;make sure we are in
PR_2
        movf        T2_COMPARE_PAIR, w                  ;set the compare value
        movwf       PR2                                 ;move working register back
to PR_2

        goto        Pop;                                ;End of the Interrupt

TimePWM:

        ;Clear the Flag so we go to the intermediate PWM next
        clrf        PWM_STATUS
        bcf             PWM_STATUS, PAIRED_PWM_BIT
        bcf             PWM_STATUS, TIME_PWM_BIT

        ;Set the Time PWM Pin
        banksel         LATA
        bcf             LATA, PAIRED_OUTPUT_PIN
        bsf             LATA, TIME_OUTPUT_PIN

        ;Reset Compare value
        banksel         PR2                                     ;make sure we are in
PR_2
        movf        T2_COMPARE_TIME, w                  ;set the compare value
        movwf       PR2                                 ;move working register back
to PR_2

        goto    Pop;                                ;End of the Interrupt

;Set all outputs low
IntermediatePWM:
        ;Set the Compare Vaule for the intermediate timer
        movlw       Intermediate_Period             ;Move the Maximum Period to the
intermediate compare value
```

```
        movwf   T2_COMPARE_INTERMEDIATE         ;set the intermediate
compare value to its maximum

        movf    T2_COMPARE_PAIR, w              ;Move the pair compare
value to the working register
        subwf   T2_COMPARE_INTERMEDIATE, f      ;subtract the working
register from the intermediate file

        movf    T2_COMPARE_TIME, w              ;Move the time compare
value to the working register
        subwf   T2_COMPARE_INTERMEDIATE, f      ;subtract the working
register from the intermediate file

        ;Change the Flag so that we go to the paired PWM Next
        clrf    PWM_STATUS
        bsf             PWM_STATUS, PAIRED_PWM_BIT
        bcf             PWM_STATUS, TIME_PWM_BIT

        ;Clear all Pins
        banksel         LATA
        bcf             LATA, PAIRED_OUTPUT_PIN
        bcf             LATA, TIME_OUTPUT_PIN

        ;Reset Compare value
        banksel         PR2                             ;make sure we are in
PR_2
        movf    T2_COMPARE_INTERMEDIATE, w      ;set the compare value
        movwf   PR2                             ;move working register back
to PR_2

        goto    Pop;                            ;End of the Interrupt



Push:
        movwf   W_TEMP                          ;Store the current w by
moving it directly into the temp file location

        movf    STATUS, W                       ;Store the current status by first
moving the status into W
        clrf    STATUS                          ;Clear the status
        movwf   STATUS_TEMP                     ;Move the W into the tempeorary
status
```

```
        movf    PCLATH, W                       ;Store the current PCLATH by first
moving the PCLATH into W
        clrf    PCLATH                          ;Clear the PCLATH
        movwf   PCLATH_TEMP                     ;Move the PCLATH into the
tempeorary status

        return


Pop:
        clrf    STATUS                          ;clear status

        ;Restore the PCLATH
        movf    PCLATH_TEMP, W
        movwf   PCLATH

        ;Restore the Status
        movf    STATUS_TEMP, W
        movwf   STATUS

        ;Restore the W
        swapf   W_TEMP, F
        swapf   W_TEMP, W                       ;restire WREG keeping status bits

        banksel         INTCON                          ;Turn Global
Interrupts back On
        bsf             INTCON, GIE

        retfie                                  ;return from the ISR

ToggleTest:
        banksel LATA
        bsf     LATA, TESTING_OUTPUT_PIN
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
```

```
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        bcf             LATA, TESTING_OUTPUT_PIN
        return


End ;end of program
```