；\*\*\*\*\*\*\*\*\*\*\*\*\*pins\*\*\*\*\*\*\*\*\*\*\*\*\*

;PORT A pin 0:control pot(input)

;PORT B pin 0-3:LED(output)

;PORT C pin 0:green button(input)

;pin 1:red button(input)

;PORT D pin 0:main TIP(output)

;pin 1:reduced TIP(output)

;pin 2 :sensor(input)

;PORT E pin 0-2:octal switch(input)

#include <P16F747.INC>

Title “Case study three”

\_\_CONFIG \_CONFIG1, \_FOSC\_HS & \_CP\_OFF & \_DEBUG\_OFF & \_VBOR\_2\_0 & \_BOREN\_0 & \_MCLR\_ON & \_PWRTE\_ON & \_WDT\_OFF

\_\_CONFIG \_CONFIG2, \_BORSEN\_0 & \_IESO\_OFF & \_FCMEN\_OFF

Count equ 20h

Temp equ 21h

State equ 22h

ADvalue equ 23h

Timer2 equ 24h

Timer1 equ 25h

Timer0 equ 26h

Mode equ 27h

org 00h

goto init

org 04h

goto isrService

org 15h

；\*\*\*\*\*\*\*\*\*\*\*\*\*init\*\*\*\*\*\*\*\*\*\*\*\*\*

init

clrf PORTB

clrf PORTC

clrf PORTD

clrf PORTE

bsf STATUS,RP0 ;Set bit in STATUS register for bank 1

movlw B'11110000';

movwf TRISB ;set Port B pin 0-3 as all outputs for LEDs

;set unused pins as inputs

movlw B'11111100';

movwf TRISD ;Port D pin0:main TIP(output)

;Port D pin1:reduced TIP(output)

;Port D pin2:an input pin used for sensor

;set unused pins as inputs

movlw B'11111111'

movwf TRISC ;Configure Port C as all inputs

;Port C pin 0:green button

;Port C pin 1:red button

movlw B'1111'

movwf TRISE ;Configure Port E pin 0,1,2 as inputs from octal switch

;set unused pins as outputs

bcf STATUS,RP0 ;Set bit in STATUS register for bank 0

clrf Count ;Zero the counter

waitPress

btfsc PORTC,0

goto GreenPress

goto waitPress

GreenPress

btfss PORTC,0 ;See if green button still Pressed

goto waitPress

GreenRelease

btfsc PORTC,0 ;See if green button released

goto GreenRelease

call SwitchDelay

Classify

comf PORTE,w ;compliment PORTE and store the result in w

andlw B'00000111' ;only need last three numbers to indicate mode

movwf State ;The value of State is the mode we choose

bcf STATUS,Z

xorwf 01h ;xor w with 1

btfsc STATUS,Z ;if w=1

goto initPortMode1 ;execute mode1

bcf STATUS,Z

xorwf,02h ;xor w with 2

btfsc STATUS,Z ;if w=2

goto initPortMode2 ;execute mode2

bcf STATUS,Z

xorwf,03h ;xor w with 3

btfsc STATUS,Z ;if w=3

goto initPortMode3 ;execute mode3

bcf STATUS,Z

xorwf,04h ;xor w with 4

btfsc STATUS,Z ;if w=4

goto initPortMode4 ;execute mode4

goto initError ;if the mode isn't 1,2,3 or 4,that'll be an error.

；\*\*\*\*\*\*\*\*\*\*\*\*\*mode1\*\*\*\*\*\*\*\*\*\*\*\*\*

initPortMode1

;bsf Mode,1 ;make the pin1 of Mode to be 1(others are 0)

clrf PORTB

bsf PORTB,0 ; make LEDs show mode 1

waitPress1

btfsc PORTC,0

goto GreenPress1

btfsc PORTC,1

goto RedPress1

goto waitPress1

GreenPress1

btfss PORTC,0 ;See if green button still Pressed

goto waitPress1

GreenRelease1

btfsc PORTC,0 ;See if green button released

goto GreenRelease1

call SwitchDelay

;movlw 0h

;movwf PORTB ;turn off the LEDs？

goto Classify

RedPress1

btfss PORTC,1 ;See if red button still Pressed(avoid noise)

goto waitPress1

RedRelease1

btfsc PORTC,1 ;See if red button is released

goto RedRelease1 ;wait until it's released

call SwitchDelay ;let switch debounce

movlw 01h ;w=1

bcf STATUS,Z ;clear Z before xor

xorwf Count,f ;if Count is 1, then it’ll be 0. If it’s 0,then it’ll be 1. Store the result in Count.

outCount

btfss STATUS,Z ;if Count=0

call SolenoidEngaged ;make solenoid engaged

btfsc STATUS,Z ;if Count=1

call SolenoidDis ;make solenoid disengaged

goto waitPress1

；\*\*\*\*\*\*\*\*\*\*\*\*\*mode2\*\*\*\*\*\*\*\*\*\*\*\*\*

initPortMode2

bsf Mode,2 ;make the pin2 of Mode to be 1(others are 0)

movlw 02h

movwf PORTB; make LEDs show mode 2

waitPress2

btfsc PORTC,0

goto GreenPress2

btfsc PORTC,1

goto RedPress2

goto waitPress2

GreenPress2

btfss PORTC,0 ;See if green button still Pressed

goto waitPress2

GreenRelease2

btfsc PORTC,0 ;See if green button released

goto GreenRelease2

call SwitchDelay

;movlw 0h

;movwf PORTB ;turn off the LEDs??

goto Classify

RedPress2

btfss PORTC,1 ;See if red button still Pressed

goto waitPress2

RedRelease2

btfsc PORTC,1 ;See if red button is released

goto RedRelease2

call SwitchDelay ;let switch debounce

initAD

bsf STATUS,RP0

movlw B'00000100'

movwf ADCON1

bcf STATUS,RP0

movlw B'01000001'

movwf ADCON0

call SetupDelay

bsf ADCON0,GO

waitloop

btfsc ADCON0,GO ;check if A/D is finished

goto waitloop

;After A/D finished, get AD value

btfsc ADCON0,GO ;make sure A/D finished

goto waitloop ;if not,continiue to wait

movf ADRESH,W ;get the value

movwf ADvalue ;put the value into the variable,ADvalue

ADvalueZero

movwf 0h ;w=0

bcf STATUS,Z ;clear Z before xor

xorlw ADvalue ;xor ADvalue with w(0)

btfsc STATUS,Z ;if z=1,i.e.ADvalue = 0

goto initError ;ADvalue can't be 0

call SolenoidEngaged

initoneforthsecond2

;initialize the time variables.

;One forth of 333,333 is 83333,which is 14585 in hex.

movlw 02h

movwf Timer2 ;get the most significant value+1

movlw 45h

movwf Timer1

movlw 85h

movwf Timer0

oneforthvalue2

;make the solenoid engage for ¼ the value

;of the control pot in seconds.

movlw 0h

bcf STATUS,Z

decf ADvalue,F ;ADvalue=ADvalue-1,until ADvalue=0

xor ADvalue,W

btfsc STATUS,Z

call SolenoidDis

btfsc STATUS,Z

goto waitPress2

;every time before lighting it for one forth second,

;check whether red button is pressed again.

btfsc PORTC,1 ;See if red button Pressed

goto RedAgain ;deal with the suitation that red button is pressed before the timing finishes.

call Timedelay ;delay one forth second

goto initoneforthsecond2

RedAgain

btfss PORTC,1 ;See if red button still Pressed(avoid noise).

return ;if it's pressed by noise,then keep finishing timing.

goto RedRelease2

；\*\*\*\*\*\*\*\*\*\*\*\*\*mode3\*\*\*\*\*\*\*\*\*\*\*\*\*

initPortMode3

bsf Mode,3 ;make the pin3 of Mode to be 1(others are 0)

movlw 03h

movwf PORTB; make LEDs show mode 3

waitPress3

btfsc PORTC,0

goto GreenPress3

btfsc PORTC,1

goto RedPress3

goto waitPress3

GreenPress3

btfss PORTC,0 ;See if green button still Pressed

goto waitPress3

GreenRelease3

btfsc PORTC,0 ;See if green button released

goto GreenRelease3

call SwitchDelay

;movlw 0h

;movwf PORTB ;turn off the LEDs??

goto Classify

RedPress3

btfss PORTC,1 ;See if red button still Pressed

goto waitPress3

RedRelease3

btfsc PORTC,1 ;See if red button is released

goto RedRelease3

call SwitchDelay ;let switch debounce

initAD

bsf STATUS,RP0

movlw B'00000100'

movwf ADCON1

bcf STATUS,RP0

movlw B'01000001'

movwf ADCON0

call SetupDelay

Active

btfsc PORTC,1 ;if red button is pressed

goto waitPress3 ;make control inactive and wait for the button pressed

bsf ADCON0,GO

waitloop

btfsc ADCON0,GO ;check if A/D is finished

goto waitloop

;After A/D finished, get AD value

btfsc ADCON0,GO ;make sure A/D finished

goto waitloop ;if not,continiue to wait

movf ADRESH,W ;get the value

movwf ADvalue ;put the value into the variable,ADvalue

ADvalueZero

movwf 0h ;w=0

bcf STATUS,Z

xorlw ADvalue ;xor ADvalue with w(0)

btfsc STATUS,Z ;if z=1,i.e.ADvalue = 0

goto initError ;ADvalue can't be 0

Compare70h

subwf ADvalue,70h

btfsc STATUS,C ;if C is 1,i.e.the result is positive or zero, AD≥70h,then the solenoid engages

call SolenoidEngaged

btfss STATUS,C ;if the result is negative,then the soleniod disengaged

call SolenoidDis

goto Active ;convert the AD value again and compare it to 70h and do something with solenoid

；\*\*\*\*\*\*\*\*\*\*\*\*\*mode4\*\*\*\*\*\*\*\*\*\*\*\*\*

initPortMode4

bsf Mode,4 ;make the pin4 of Mode to be 1(others are 0)

movlw 04h

movwf PORTB; make LEDs show mode 4

clrf Count

initAD

bsf STATUS,RP0

movlw B'00000100'

movwf ADCON1

bcf STATUS,RP0

movlw B'01000001'

movwf ADCON0

call SetupDelay

bsf ADCON0,GO

waitloop

btfsc ADCON0,GO ;check if A/D is finished

goto waitloop

;After A/D finished, get AD value

btfsc ADCON0,GO ;make sure A/D finished

goto waitloop ;if not,continiue to wait

movf ADRESH,W ;get the value

movwf ADvalue ;put the value into the variable,ADvalue

ADvalueZero

movwf 0h ;w=0

bcf STATUS,Z

xorlw ADvalue ;xor ADvalue with w(0)

btfsc STATUS,Z ;if z=1,i.e.ADvalue = 0

goto initError ;ADvalue can't be 0

waitPress4

btfsc PORTC,0

goto GreenPress4

btfsc PORTC,1

goto RedPress4

goto waitPress4

GreenPress4

btfss PORTC,0 ;See if green button still Pressed

goto waitPress4

GreenRelease4

btfsc PORTC,0 ;See if green button released

goto GreenRelease4

call SwitchDelay

;movlw 0h

;movwf PORTB ;turn off the LEDs??

goto Classify

RedPress4

btfss PORTC,1 ;See if red button still Pressed

goto waitPress4

RedRelease4

btfsc PORTC,1 ;See if red button is released

goto RedRelease4

call SwitchDelay ;let switch debounce

initTenSeconds

movlw 33h

movwf Timer2 ;get the most significant value+1

movlw 0xdc

movwf Timer1

movlw 0xd2

movwf Timer0

MainTIPon

bsf PORTD,0

;If the optical sensor does not indicate that the solenoid has retracted in 10

;seconds, turn off the main transistor and indicate a fault.

Timedelay4

;loop until the time that Timer varibales indicates

btfsc PORTD,2 ;if the solenoid has been engaged

goto ReducedTIPon

decfsz Timer0,F ;

goto Timedelay4

decfsz Timer1,F ;

goto Timedelay4

decfsz Timer2,F ;

goto Timedelay4

goto initError

ReducedTIPon

bsf PORTD,1 ;turn on the reduced TIP

bcf PORTD,0 ;turn off the main TIP

;If the optical sensor indicates that the solenoid has disengaged when

;the reduced transistor in on, restart the whole sequence again (one time).

btfss PORTD,2 ;if the solenoid hasn't been engaged

incf Count ;every time sensor indicates that solenoid hasn't been engaged,Count++

bcfsc Count,2 ;if Count is B'10',it's the second time.

goto initError

btfss PORTD,2 ;if the solenoid hasn't been engaged

goto MainTIPon ;restart the whole sequence

initoneforthsecond4

;initialize the time variables.

;One forth of 333,333 is 83333,which is 14585 in hex.

movlw 02h

movwf Timer2 ;get the most significant value+1

movlw 45

movwf Timer1

movlw 85

movwf Timer0

oneforthvalue4

;make the solenoid engage for ¼ the value

;of the control pot in seconds.

movlw 0h

bcf STATUS,Z

decf ADvalue,F ;ADvalue=ADvalue-1,until ADvalue=0

xor ADvalue,W

;if ADvalue is 0,then disengage the solenoid and go back to waitPress4

btfsc STATUS,Z

call SolenoidDis

btfsc STATUS,Z

goto waitPress4

;if ADvalue doesn't equal to 0, then delay for one forth second

call Timedelay ;delay one forth second

goto initoneforthsecond4

;If the solenoid is turned off and the optical sensor indicates that the solenoid is

;still retracted in 10 seconds, also indicate a fault.

initTenSeconds

movlw 33h

movwf Timer2 ;get the most significant value+1

movlw 0xdc

movwf Timer1

movlw 0xd2

movwf Timer0

call Timedelay ;delay for ten seconds

bcfsc PORTD,2 ;if the solenoid is still engaged

goto initError

goto initPortMode4

；\*\*\*\*\*\*\*\*\*\*\*\*\*kinds of Delay\*\*\*\*\*\*\*\*\*\*\*\*\*

Timedelay

;loop until the time that Timer varibales indicates

decfsz Timer0,F ;

goto Timedelay

decfsz Timer1,F ;

goto Timedelay

decfsz Timer2,F ;

goto Timedelay

return

SwitchDelay

movlw D'20'

movwf Temp

goto delay

SetupDelay

movlw 03h

movwf Temp ;load Temp with hex 3

delay

decfsz Temp,F

goto delay

return

；\*\*\*\*\*\*\*\*\*\*\*\*\*Solenoid\*\*\*\*\*\*\*\*\*\*\*\*\*

SolenoidEngaged

;let the solenoid engages

bsf PORTD,0 ;turn on the main TIP

SolenoidLoop

btfss PORTD,2 ;if the solenoid hasn't been engaged,i.e.PORTD pin2 is 0,then keep waiting

goto SolenoidLoop

bsf PORTD,1 ;turn on the reduced TIP

bcf PORTD,0 ;turn off the main TIP

return

SolenoidDis

;make the solenoid disengaged

bcf PORTD,0 ;turn off the main TIP

bcf PORTD,1 ;turn off the reduced TIP

return

；\*\*\*\*\*\*\*\*\*\*\*\*\*Error\*\*\*\*\*\*\*\*\*\*\*\*\*

initError

movf State,w ;w=State

movwf PORTB ;make the LEDs show the wrong mode we choose

clrf PORTD ;make the solenoid disengaged

Errorloop

bsf PORTB,3 ;turn on the LED

call onesecondTimer ;set Timer2,Timer1,Timer0 equal to one second

bcf PORTB,3;turn off the LED

call onesecondTimer ;set Timer2,Timer1,Timer0 equal to one second

goto Errorloop

onesecondTimer

movlw 06h

movwf Timer2 ;get the most significant value+1

movlw 16h

movwf Timer1

movlw 15h

movwf Timer0

call Timedelay ;loop until the time that Timer varibales indicates

return

isrService

goto isrService

end