EE322: EMBEDDED SYSTEMS

DESIGN

FERNANDO P.D.R.

E/16/103

GROUP 10

SEMESTER 5

LAB 3 PART 2 : PROBLEM

The problem with the submitted code is that when the voltage level given to the Vss is changed the voltage level given to R0 will be changed according to each switch. Practically this cannot be done since the Vss might get be changed with time. From –Vref to +Vref analog to digital converter will quantize 2^10 levels. For different voltages for a threshold of (+Vref--Vref)/2^9 it gives different binary numbers.

SOLUTION

We can add a subroutine to calibrate the voltage values given by different buttons once and store it in the EEPROM memory and save the corresponding Vss in the EEPROM memory. When program is started check if the preciously saved Vss value is the same with the current value. If true we don’t have to caliberate.In contrast we can load the previously saved data and use it for the program to compare the R0 value with them. If Vss value has changed we can call the caliberate subroutine to save the new button voltages to EEPROM. The Code I implemented to do this but failed is

processor 16f877a

#include<p16f877a.inc>

org 0x00

TEMP equ 0x21

COUNT1 equ 0x20

COUNT2 equ 0x22

ADDR equ 0x23

VALUE equ 0x24

BUTTON1 equ 0x55

BUTTON2 equ 0x56

BUTTON3 equ 0x57

BUTTON4 equ 0x58

BUTTON5 equ 0x59

BUTTON6 equ 0x60

CHECK equ 0x31

numberONE equ 0x32

numberTWO equ 0x33

INTCON equ 0Bh

STATUS equ 03h

TRISA equ 85h

PORTA equ 05h

TRISD equ 88h

PORTD equ 08h

goto Main

Main:

bsf STATUS,5

bcf STATUS,6 ;selecting bank 1

movlw b'00000000' ;setting port d as output

movwf TRISD

movlw b'11111' ;setting port A as input

movwf TRISA

bcf STATUS,5 ;moving back to bank 0

clrf ADCON1

bsf ADCON1,3 ;configure pins in port A to 1000,adfm

bsf ADCON1,7

movlw b'11000001' ;set A/D coversion on,set clock to Focs8

movwf ADCON0

CALL caliberation

loop1

call adc ;READING THE VALUE AND ASSIGNING TO NUMBERONE

movf TEMP,0

MOVWF numberONE

MOVF BUTTON1,0 ;READING DATA FROM THE STORED BUTTON VALUES

MOVWF ADDR

CALL readdata

MOVWF numberTWO ;ASSIGN IT TO NUMBER TWO

CALL compare ;number

btfsc CHECK,0 ;if no1=no2 btfsc=1 which means go to next step

call switch1

;if button 1 val != the analog input

call adc ;READING THE VALUE AND ASSIGNING TO NUMBERONE

movf TEMP,0

MOVWF numberONE

MOVF BUTTON2,0 ;READING DATA FROM THE STORED BUTTON VALUES

MOVWF ADDR

CALL readdata

MOVWF numberTWO ;ASSIGN IT TO NUMBER TWO

CALL compare ;number

btfsc CHECK,0 ;if no1=no2 btfsc=1 which means go to next step

call switch2

call adc ;READING THE VALUE AND ASSIGNING TO NUMBERONE

movf TEMP,0

MOVWF numberONE

MOVF BUTTON3,0 ;READING DATA FROM THE STORED BUTTON VALUES

MOVWF ADDR

CALL readdata

MOVWF numberTWO ;ASSIGN IT TO NUMBER TWO

CALL compare ;number

btfsc CHECK,0 ;if no1=no2 btfsc=1 which means go to next step

call switch3

call adc ;READING THE VALUE AND ASSIGNING TO NUMBERONE

movf TEMP,0

MOVWF numberONE

MOVF BUTTON4,0 ;READING DATA FROM THE STORED BUTTON VALUES

MOVWF ADDR

CALL readdata

MOVWF numberTWO ;ASSIGN IT TO NUMBER TWO

CALL compare ;number

btfsc CHECK,0 ;if no1=no2 btfsc=1 which means go to next step

call switch4

call adc ;READING THE VALUE AND ASSIGNING TO NUMBERONE

movf TEMP,0

MOVWF numberONE

MOVF BUTTON5,0 ;READING DATA FROM THE STORED BUTTON VALUES

MOVWF ADDR

CALL readdata

MOVWF numberTWO ;ASSIGN IT TO NUMBER TWO

CALL compare ;number

btfsc CHECK,0 ;if no1=no2 btfsc=1 which means go to next step

call switch5

call switch6

goto loop1

adc

bcf ADCON0,3 ;setting R0 as input

call delay

bsf ADCON0,2 ;starting the a/d onversion

wait

btfsc ADCON1,2 ;when conv over ,goes to adresh

goto wait

movf ADRESL,0 ;move adresh to portd

movwf TEMP

call delay

return

delay

loop2

decfsz COUNT1,1

goto loop2

decfsz COUNT2,1

goto loop2

return

writedata

BSF STATUS,6 ;

BSF STATUS,5 ;Bank 3

BTFSC EECON1,1 ;Wait for

GOTO $-1 ;write to finish

BCF STATUS,5 ;Bank 2

MOVF ADDR,0 ;Address to

MOVWF EEADR ;write to

MOVF VALUE,0 ;Data to

MOVWF EEDATA ;write

BSF STATUS,5 ;Bank 3

BCF EECON1,7 ;Point to Data memory

BSF EECON1,2 ;Enable writes Only disable interrupts

BCF INTCON,7 ;if already enabled otherwise discard

MOVLW 0x55 ;Write 55h to

MOVWF EECON2 ;EECON2

MOVLW 0xAA ;Write AAh to

MOVWF EECON2 ;EECON2

BSF EECON1,1 ;Start write operation Only enable interrupts

BSF INTCON,7 ;if using interrupts,otherwise discard

BCF EECON1,2 ;Disable writes

return

caliberation

call delay

movlw b'11111111' ;indicating to start caliberate by lighting all

movwf PORTD ;the leds

call delay

movlw b'00000000'

movwf PORTD

call adc ;caliberating for button1 which is 0 V ,bottom

movf TEMP,0 ;one

movwf VALUE ;loading the data from analog read

movf BUTTON1,0 ;moving address to ADDR

movwf ADDR

call writedata

call switch1

call adc

movf TEMP,0

movwf VALUE

movf BUTTON2,0

movwf ADDR

call writedata

call switch2

call adc

movf TEMP,0

movwf VALUE

movf BUTTON3,0

movwf ADDR

call writedata

call switch3

call adc

movf TEMP,0

movwf VALUE

movf BUTTON4,0

movwf ADDR

call writedata

call switch4

call adc

movf TEMP,0

movwf VALUE

movf BUTTON5,0

movwf ADDR

call writedata

call switch5

call adc

movf TEMP,0

movwf VALUE

movf BUTTON6,0

movwf ADDR

call writedata

call switch6

return

compare

clrf CHECK ;CLEAR CHECK

bcf STATUS,0

bcf STATUS,2

movf numberTWO,0

subwf numberONE,0

btfsc STATUS,2

bsf CHECK,0 ;set 0th bit of check to 1 if no1=no2

btfss STATUS,0

bsf CHECK,1 ;set 1st bit of check to 1 if no2>no1

btfsc STATUS,0

bsf CHECK,2 ;set 2nd bit of check to 1 if no1>no2

bcf STATUS,0

bcf STATUS,2

return

readdata

BSF STATUS,6 ;

BCF STATUS,5 ;Bank 2

MOVF ADDR,0 ;Write address

MOVWF EEADR ;to read from

BSF STATUS,5 ;Bank 3

BCF EECON1,7 ;Point to Data memory

BSF EECON1,0 ;Start read operation

BCF STATUS,5 ;Bank 2

MOVF EEDATA,0 ;W = EEDATA

return

switch1

movlw b'10000000'

movwf PORTD

call delay

movlw b'00000000'

movwf PORTD

return

switch2

movlw b'01000000' ;led

movwf PORTD

call delay

movlw b'00000000'

movwf PORTD

return

switch3

movlw b'00100000' ;led

movwf PORTD

call delay

movlw b'00000000'

movwf PORTD

return

switch4

movlw b'00010000' ;led

movwf PORTD

call delay

movlw b'00000000'

movwf PORTD

return

switch5

movlw b'00001000' ;led

movwf PORTD

call delay

movlw b'00000000'

movwf PORTD

return

switch6

movlw b'00000100' ;led

movwf PORTD

call delay

movlw b'00000000'

movwf PORTD

return

end