Lab 10 EEPROM

JCH
11/5/25
6:45-7:00

Goals

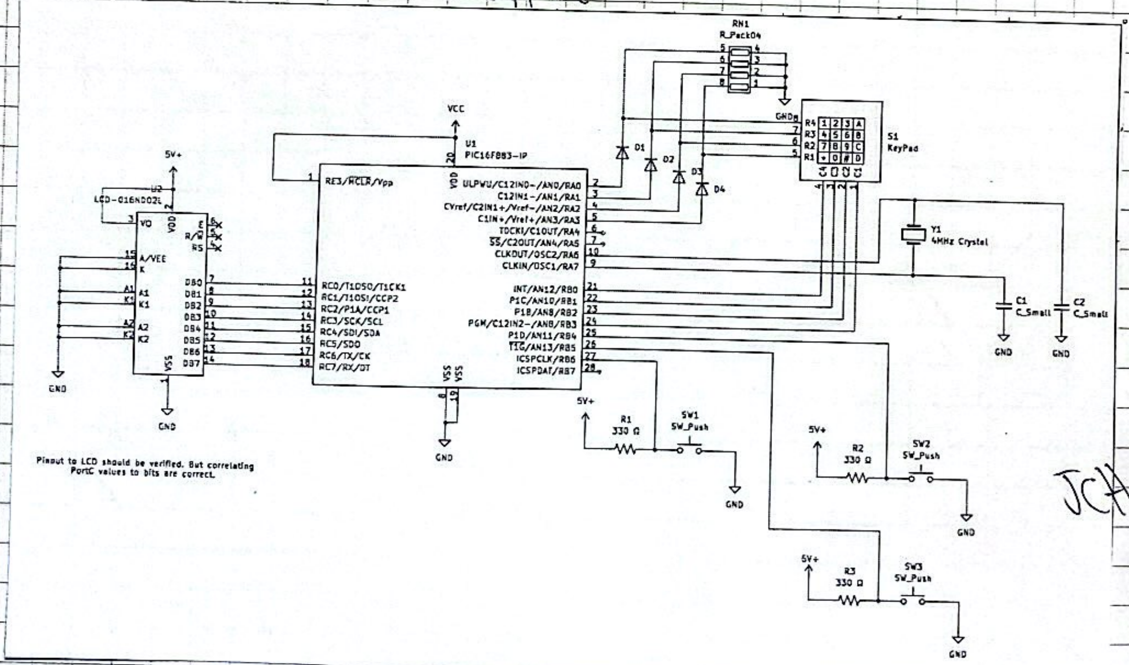
Write and troubleshoot a program that saves data in the EEPROM and then recalls the data and displays it on a dot matrix display.

Objective 1

Write a program that waits for a start recording switch to be pressed and blinks an "S" at one second intervals while it waits. once the record button is pressed any time a key is pressed on the keypad the dot matrix equivalent is recorded into EEPROM. It should record up to 10 values or stop recording when the stop button is pressed. The program should keep track of how many buttons were pressed.

Figure 10-1 Schematic for EEPROM with keypad

JDR
-6-25



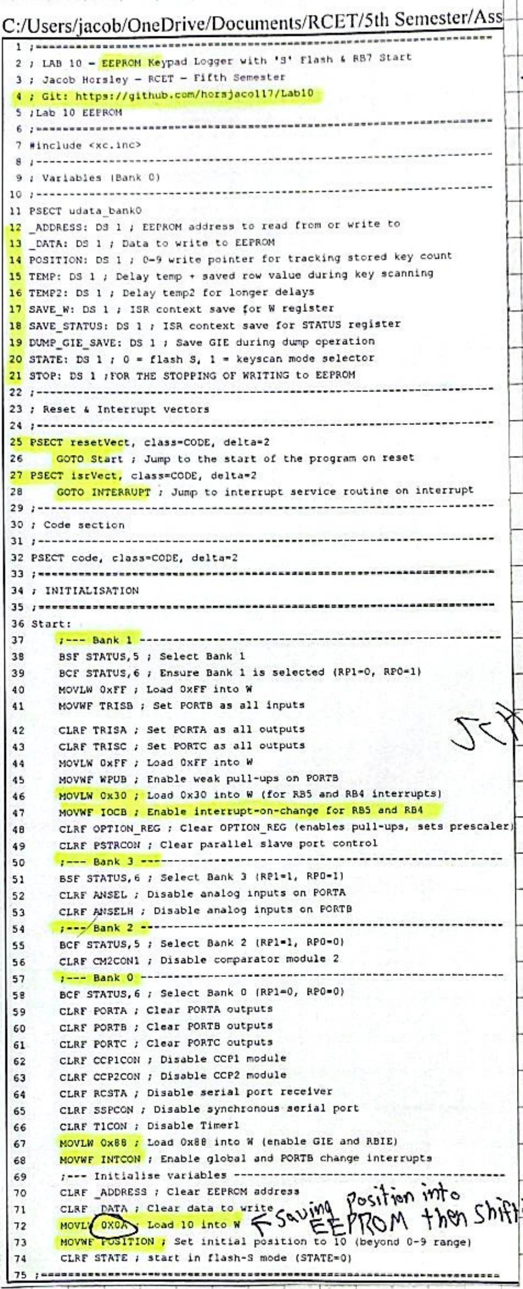
Pinout to LCD should be verified. But correlating PortC values to bits are correct.

JCH
11/6/25
11:00-11:30

Lab 10 EEPROM

JCH
11/6/25
11:30-12:00

Figure 10-2 Configuration For Lab 10 code



C:/Users/jacob/OneDrive/Documents/RCET/5th Semester/Ass

```
;==========================================================================
; LAB 10 - EEPROM Keypad Logger with 'S' Flash & RB7 Start
; Jacob Horsley - RCET - Fifth Semester
; Git: https://github.com/horsjacol17/Lab10
;Lab 10 EEPROM
;==========================================================================
#include <xc.inc>
;--------------------------------------------------------------------------
; Variables (Bank 0)
;--------------------------------------------------------------------------
PSECT udata_bank0
_ADDRESS: DS 1 ; EEPROM address to read from or write to
_DATA: DS 1 ; Data to write to EEPROM
POSITION: DS 1 ; 0-9 write pointer for tracking stored key count
TEMP: DS 1 ; Delay temp + saved row value during key scanning
TEMP2: DS 1 ; Delay temp2 for longer delays
SAVE_W: DS 1 ; ISR context save for W register
SAVE_STATUS: DS 1 ; ISR context save for STATUS register
DUMP_GIE_SAVE: DS 1 ; Save GIE during dump operation
STATE: DS 1 ; 0 = flash S, 1 = keyscan mode selector
STOP: DS 1 ;FOR THE STOPPING OF WRITING to EEPROM
;--------------------------------------------------------------------------
; Reset & Interrupt vectors
;--------------------------------------------------------------------------
PSECT resetVect, class=CODE, delta=2
    GOTO Start ; Jump to the start of the program on reset
PSECT isrVect, class=CODE, delta=2
    GOTO INTERRUPT ; Jump to interrupt service routine on interrupt
;--------------------------------------------------------------------------
; Code section
;--------------------------------------------------------------------------
PSECT code, class=CODE, delta=2
;==========================================================================
; INITIALISATION
;==========================================================================
Start:
    ;--- Bank 1 ------------------------------------------------------------
    BSF STATUS,5 ; Select Bank 1
    BCF STATUS,6 ; Ensure Bank 1 is selected (RP1=0, RP0=1)
    MOVLW 0xFF ; Load 0xFF into W
    MOVWF TRISB ; Set PORTB as all inputs

    CLRF TRISA ; Set PORTA as all outputs
    CLRF TRISC ; Set PORTC as all outputs
    MOVLW 0xFF ; Load 0xFF into W
    MOVWF WPUB ; Enable weak pull-ups on PORTB
    MOVLW 0x30 ; Load 0x30 into W (for RB5 and RB4 interrupts)
    MOVWF IOCB ; Enable interrupt-on-change for RB5 and RB4
    CLRF OPTION_REG ; Clear OPTION_REG (enables pull-ups, sets prescaler)
    CLRF PSTRCON ; Clear parallel slave port control
    ;--- Bank 3 ------------------------------------------------------------
    BSF STATUS,6 ; Select Bank 3 (RP1=1, RP0=1)
    CLRF ANSEL ; Disable analog inputs on PORTA
    CLRF ANSELH ; Disable analog inputs on PORTB
    ;--- Bank 2 ------------------------------------------------------------
    BCF STATUS,5 ; Select Bank 2 (RP1=1, RP0=0)
    CLRF CM2CON1 ; Disable comparator module 2
    ;--- Bank 0 ------------------------------------------------------------
    BCF STATUS,6 ; Select Bank 0 (RP1=0, RP0=0)
    CLRF PORTA ; Clear PORTA outputs
    CLRF PORTB ; Clear PORTB outputs
    CLRF PORTC ; Clear PORTC outputs
    CLRF CCP1CON ; Disable CCP1 module
    CLRF CCP2CON ; Disable CCP2 module
    CLRF RCSTA ; Disable serial port receiver
    CLRF SSPCON ; Disable synchronous serial port
    CLRF T1CON ; Disable Timer1
    MOVLW 0x88 ; Load 0x88 into W (enable GIE and RBIE)
    MOVWF INTCON ; Enable global and PORTB change interrupts
    ;--- Initialise variables ----------------------------------------------
    CLRF _ADDRESS ; Clear EEPROM address
    CLRF _DATA ; Clear data to write
    MOVLW 0X0A ; Load 10 into W
    MOVWF POSITION ; Set initial position to 10 (beyond 0-9 range)
    CLRF STATE ; start in flash-S mode (STATE=0)
;==========================================================================
```

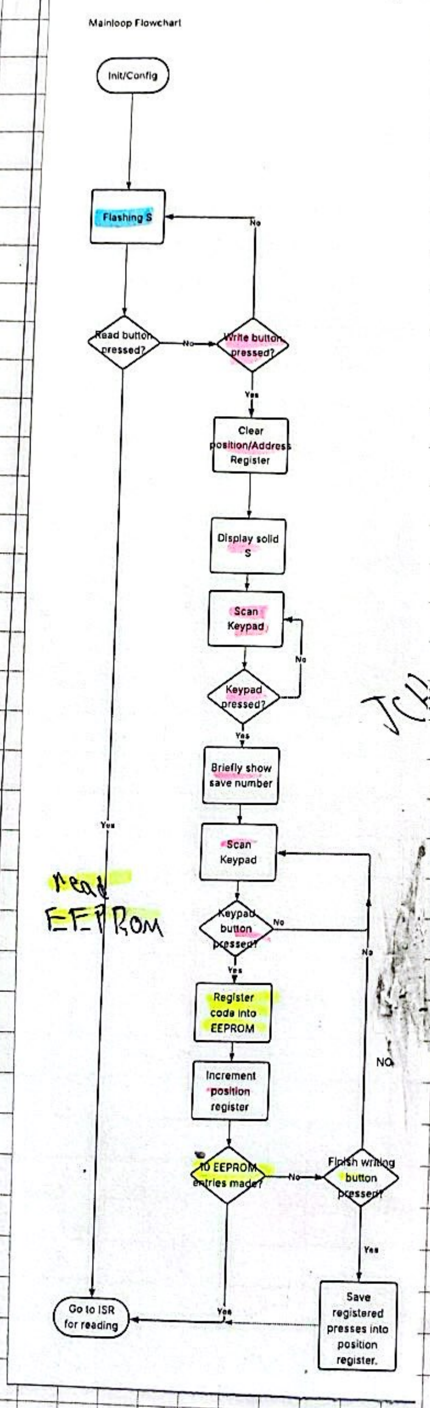
JCH

← saving position into EEPROM then shifting it into position works

This configuration primarily configures the Ports. Port C goes to the Dot Matrix display. Ports A and B are a mix of inputs and outputs for the keypad logic.

Lab 10 EEPROM

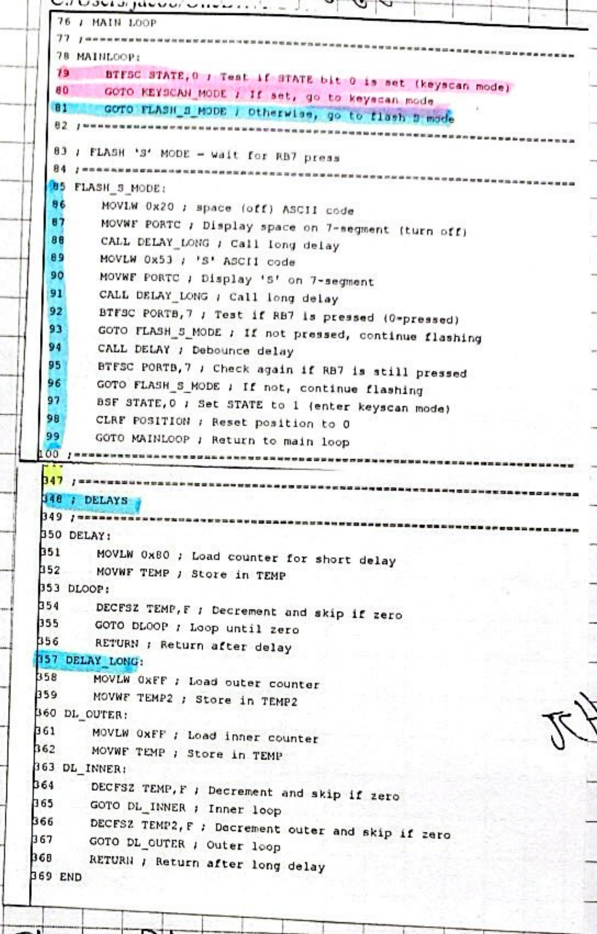
Figure 10-3 Flow chart for Mainloop



read EEPROM

JCH

Figure 10-4 Main flashing S code



```
; MAIN LOOP
;==================================================================
MAINLOOP:
    BTFSC STATE,0 ; Test if STATE bit 0 is set (keyscan mode)
    GOTO KEYSCAN_MODE ; If set, go to keyscan mode
    GOTO FLASH_S_MODE ; Otherwise, go to flash S mode
;==================================================================
; FLASH 'S' MODE - wait for RB7 press
;==================================================================
FLASH_S_MODE:
    MOVLW 0x20 ; space (off) ASCII code
    MOVWF PORTC ; Display space on 7-segment (turn off)
    CALL DELAY_LONG ; Call long delay
    MOVLW 0x53 ; 'S' ASCII code
    MOVWF PORTC ; Display 'S' on 7-segment
    CALL DELAY_LONG ; Call long delay
    BTFSC PORTB,7 ; Test if RB7 is pressed (0=pressed)
    GOTO FLASH_S_MODE ; If not pressed, continue flashing
    CALL DELAY ; Debounce delay
    BTFSC PORTB,7 ; Check again if RB7 is still pressed
    GOTO FLASH_S_MODE ; If not, continue flashing
    BSF STATE,0 ; Set STATE to 1 (enter keyscan mode)
    CLRF POSITION ; Reset position to 0
    GOTO MAINLOOP ; Return to main loop
;==================================================================
;==================================================================
; DELAYS
;==================================================================
DELAY:
    MOVLW 0x80 ; Load counter for short delay
    MOVWF TEMP ; Store in TEMP
DLOOP:
    DECFSZ TEMP,F ; Decrement and skip if zero
    GOTO DLOOP ; Loop until zero
    RETURN ; Return after delay
DELAY_LONG:
    MOVLW 0xFF ; Load outer counter
    MOVWF TEMP2 ; Store in TEMP2
DL_OUTER:
    MOVLW 0xFF ; Load inner counter
    MOVWF TEMP ; Store in TEMP
DL_INNER:
    DECFSZ TEMP,F ; Decrement and skip if zero
    GOTO DL_INNER ; Inner loop
    DECFSZ TEMP2,F ; Decrement outer and skip if zero
    GOTO DL_OUTER ; Outer loop
    RETURN ; Return after long delay
END
```

JCH

The flashing main loop is really simple and just flashes an a ASCII S. There are buttons that trigger a read and write function. The read also doubles as a stop button

JCH
10/5/25
~~12:00~~
1-1:30

Lab 10 EEPROM

JCH
10/6/25
1:30-2:00

Figure 10-5 Keyscan Code and logic

```
150 ; KEY HANDLERS - One per key
151 ;=====================================================================
152 DISP_1:
153     MOVLW 0x31 ; ASCII '1'
154     GOTO HANDLE_KEY ; Go to key handler
155 DISP_2:
156     MOVLW 0x32 ; ASCII '2'
157     GOTO HANDLE_KEY ; Go to key handler
158 DISP_3:
159     MOVLW 0x33 ; ASCII '3'
160     GOTO HANDLE_KEY ; Go to key handler
161 DISP_4:
162     MOVLW 0x34 ; ASCII '4'
163     GOTO HANDLE_KEY ; Go to key handler
164 DISP_5:

165     MOVLW 0x35 ; ASCII '5'
166     GOTO HANDLE_KEY ; Go to key handler
167 DISP_6:
168     MOVLW 0x36 ; ASCII '6'
169     GOTO HANDLE_KEY ; Go to key handler
170 DISP_7:
171     MOVLW 0x37 ; ASCII '7'
172     GOTO HANDLE_KEY ; Go to key handler
173 DISP_8:
174     MOVLW 0x38 ; ASCII '8'
175     GOTO HANDLE_KEY ; Go to key handler
176 DISP_9:
177     MOVLW 0x39 ; ASCII '9'
178     GOTO HANDLE_KEY ; Go to key handler
179
180 DISP_A: MOVLW 0x0A ; Code for 'A'
                GOTO HANDLE_KEY ; Go to key handler
181
182 DISP_B: MOVLW 0x0B ; Code for 'B'
                GOTO HANDLE_KEY ; Go to key handler
183
184 DISP_C: MOVLW 0x0C ; Code for 'C'
                GOTO HANDLE_KEY ; Go to key handler
185
186 DISP_D: MOVLW 0x0D ; Code for 'D' (though not used in scan, included)
                GOTO HANDLE_KEY ; Go to key handler
187
188
189 DISP_S:
190     CLRF STOP ; Clear STOP register
191     MOVLW 0X53 ; ASCII 'S'
192     GOTO HANDLE_KEY ; Go to key handler
193 ;=====================================================================
194 ; HANDLE_KEY - Write once, wait for release
195 ;=====================================================================
196 HANDLE_KEY:
197     MOVWF _DATA ; Store key value in _DATA
198     MOVF POSITION,W ; Load current position
199     MOVWF _ADDRESS ; Set EEPROM address to position
200     CALL WRITE_EEPROM ; Write data to EEPROM
201     CALL READ_EEPROM ; Read back and display on PORTC
202     INCF POSITION,F ; Increment position
203     MOVF POSITION,W ; Load position
204     XORLW 0x0A ; Compare to 10
205     BTFSS STATUS,2 ; If not 10, skip

206     GOTO NO_DUMP2 ; Continue without dump
207     CALL DUMP ; Dump contents if 10 keys stored
208     BCF STATE,0 ; Reset to flash S mode
209     GOTO MAINLOOP ; Return to main loop
210 NO_DUMP2:
211 ; CLRF STOP ; Commented: Clear STOP
212     CALL WAIT_KEY_RELEASE ; Wait for key release (debounce)
213     GOTO MAINLOOP ; Return to main loop
214 ;=====================================================================
215 ; WAIT_KEY_RELEASE - Wait until key is released
216 ;=====================================================================
217 WAIT_KEY_RELEASE:
218     CALL DELAY ; Initial debounce delay
219 RELEASE_LOOP:
220     MOVF TEMP,W ; Restore saved row
221     MOVWF PORTA ; Re-select row
222     CALL DELAY ; Delay for settling
223     MOVF PORTB,W ; Read PORTB
224     ANDLW 0x0E ; Mask RB1,RB2,RB3 (columns)
```

JCH

```
101 ; KEYSCAN MODE
102 ;=====================================================================
103 KEYSCAN_MODE:
104     BCF PORTA,5 ; Clear RA5 (possibly for LED or indicator)
105     ;--- Row 3 ---------------------------------------------------------
106     MOVLW 0x06 ; Row 3 select value
107     MOVWF PORTA ; Select row 3 on PORTA
108     MOVWF TEMP ; Save row value in TEMP
109     CALL DELAY ; Delay for settling
110     BTFSS PORTB,3 ; Check column 3 (RB3)
111     GOTO DISP_9 ; If pressed, handle '9'
112     BTFSS PORTB,2 ; Check column 2 (RB2)
113     GOTO DISP_8 ; If pressed, handle '8'
114     BTFSS PORTB,1 ; Check column 1 (RB1)
115     GOTO DISP_7 ; If pressed, handle '7'
116     BTFSS PORTB,0 ; Check column 0 (RB0)
117     GOTO DISP_C ; If pressed, handle 'C'
118
119     ;--- Row 2 ---------------------------------------------------------
120     MOVLW 0x05 ; Row 2 select value
121     MOVWF PORTA ; Select row 2 on PORTA
122     MOVWF TEMP ; Save row value in TEMP
123     CALL DELAY ; Delay for settling

124     BTFSS PORTB,3 ; Check column 3
125     GOTO DISP_6 ; If pressed, handle '6'
126     BTFSS PORTB,2 ; Check column 2
127     GOTO DISP_5 ; If pressed, handle '5'
128     BTFSS PORTB,1 ; Check column 1
129     GOTO DISP_4 ; If pressed, handle '4'
130     BTFSS PORTB,0 ; Check column 0
131     GOTO DISP_B ,; If pressed, handle 'B'
132     ;--- Row 1 ---------------------------------------------------------
133     MOVLW 0x03 ; Row 1 select value
134     MOVWF PORTA ; Select row 1 on PORTA
135     MOVWF TEMP ; Save row value in TEMP
136     CALL DELAY ; Delay for settling
137     BTFSS PORTB,3 ; Check column 3
138     GOTO DISP_3 ; If pressed, handle '3'
139     BTFSS PORTB,2 ; Check column 2
140     GOTO DISP_2 ; If pressed, handle '2'
141     BTFSS PORTB,1 ; Check column 1
142     GOTO DISP_1 ; If pressed, handle '1'
143     BTFSS PORTB,0 ; Check column 0
144     GOTO DISP_A ; If pressed, handle 'A'
145
146     BTFSC STOP, 1 ; Check if STOP bit 1 is set
147     GOTO DISP_S ; If set, handle 'S'
148     GOTO KEYSCAN_MODE ; No key pressed, loop back
149 ;=====================================================================
```

JCH

2 1 of 5

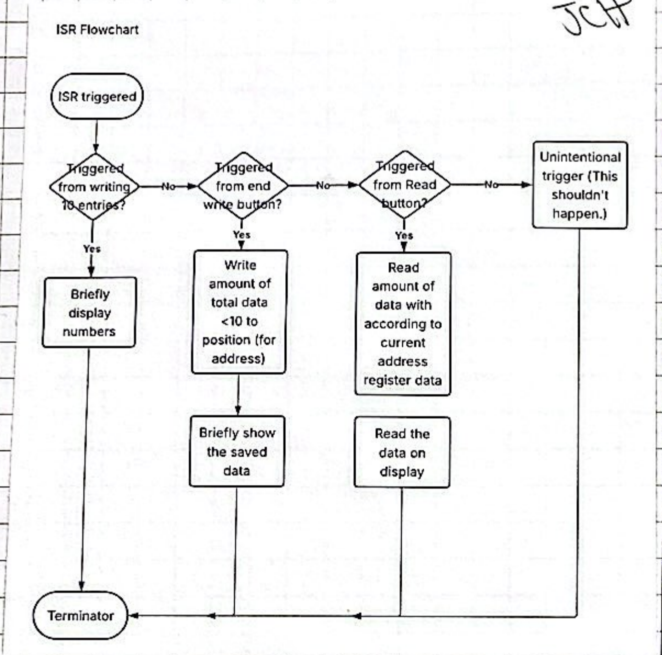
```
5       XORLW 0x0E ; Check if all high (no press)
26      BTFSS STATUS,2 ; If not all high, loop
27      GOTO RELEASE_LOOP ; Continue waiting
28      CALL DELAY ; Final debounce
29      RETURN ; Return when released
```

Take note of * line #'s
(Swap)

Lab 10 EEPROM

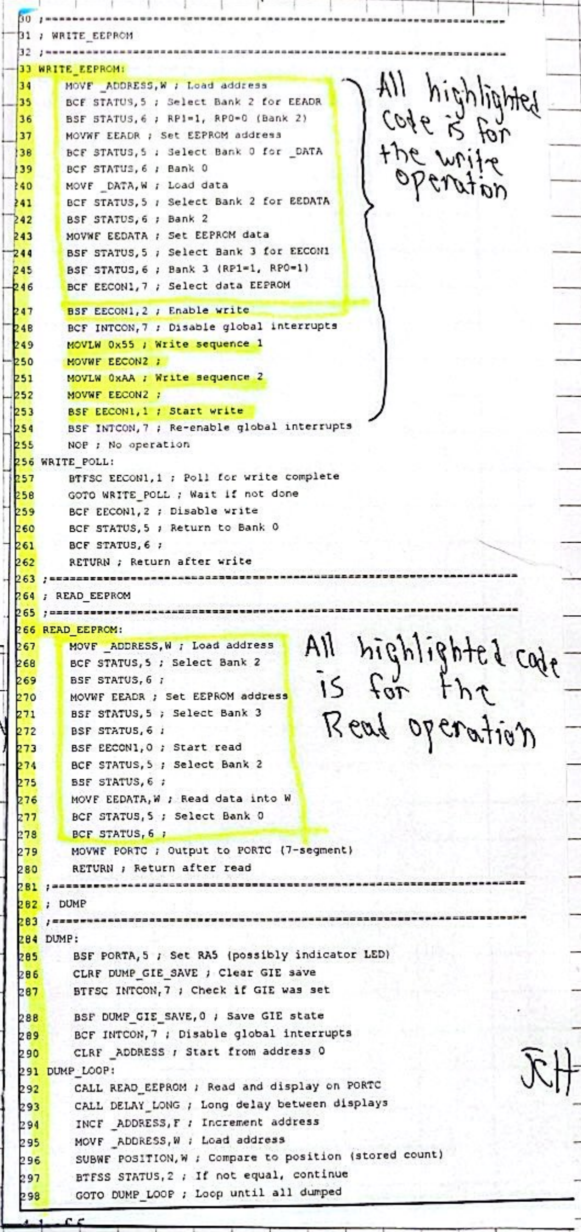
JCH 11/6/25 7:30-8:00

Figure 10-6 Flowchart for ISR


ISR Flowchart
ISR triggered
Triggered from writing 10 entries?
No
Yes
Briefly display numbers
Triggered from end write button?
Yes
Write amount of total data <10 to position (for address)
Briefly show the saved data
Triggered from Read button?
Yes
Read amount of data with according to current address register data
Read the data on display
Unintentional trigger (This shouldn't happen.)
Terminator

JCH

Figure 10-7 ISR Code w/ Read and write Part 1



```
;==========================================================================
; WRITE_EEPROM
;==========================================================================
WRITE_EEPROM:
    MOVF _ADDRESS,W ; Load address
    BCF STATUS,5 ; Select Bank 2 for EEADR
    BSF STATUS,6 ; RP1=1, RP0=0 (Bank 2)
    MOVWF EEADR ; Set EEPROM address
    BCF STATUS,5 ; Select Bank 0 for _DATA
    BCF STATUS,6 ; Bank 0
    MOVF _DATA,W ; Load data
    BCF STATUS,5 ; Select Bank 2 for EEDATA
    BSF STATUS,6 ; Bank 2
    MOVWF EEDATA ; Set EEPROM data
    BSF STATUS,5 ; Select Bank 3 for EECON1
    BSF STATUS,6 ; Bank 3 (RP1=1, RP0=1)
    BCF EECON1,7 ; Select data EEPROM
    BSF EECON1,2 ; Enable write
    BCF INTCON,7 ; Disable global interrupts
    MOVLW 0x55 ; Write sequence 1
    MOVWF EECON2 ;
    MOVLW 0xAA ; Write sequence 2
    MOVWF EECON2 ;
    BSF EECON1,1 ; Start write
    BSF INTCON,7 ; Re-enable global interrupts
    NOP ; No operation
WRITE_POLL:
    BTFSC EECON1,1 ; Poll for write complete
    GOTO WRITE_POLL ; Wait if not done
    BCF EECON1,2 ; Disable write
    BCF STATUS,5 ; Return to Bank 0
    BCF STATUS,6 ;
    RETURN ; Return after write
;==========================================================================
; READ_EEPROM
;==========================================================================
READ_EEPROM:
    MOVF _ADDRESS,W ; Load address
    BCF STATUS,5 ; Select Bank 2
    BSF STATUS,6 ;
    MOVWF EEADR ; Set EEPROM address
    BSF STATUS,5 ; Select Bank 3
    BSF STATUS,6 ;
    BSF EECON1,0 ; Start read
    BCF STATUS,5 ; Select Bank 2
    BSF STATUS,6 ;
    MOVF EEDATA,W ; Read data into W
    BCF STATUS,5 ; Select Bank 0
    BCF STATUS,6 ;
    MOVWF PORTC ; Output to PORTC (7-segment)
    RETURN ; Return after read
;==========================================================================
; DUMP
;==========================================================================
DUMP:
    BSF PORTA,5 ; Set RA5 (possibly indicator LED)
    CLRF DUMP_GIE_SAVE ; Clear GIE save
    BTFSC INTCON,7 ; Check if GIE was set

    BSF DUMP_GIE_SAVE,0 ; Save GIE state
    BCF INTCON,7 ; Disable global interrupts
    CLRF _ADDRESS ; Start from address 0
DUMP_LOOP:
    CALL READ_EEPROM ; Read and display on PORTC
    CALL DELAY_LONG ; Long delay between displays
    INCF _ADDRESS,F ; Increment address
    MOVF _ADDRESS,W ; Load address
    SUBWF POSITION,W ; Compare to position (stored count)
    BTFSS STATUS,2 ; If not equal, continue
    GOTO DUMP_LOOP ; Loop until all dumped
```

All highlighted code is for the write operation

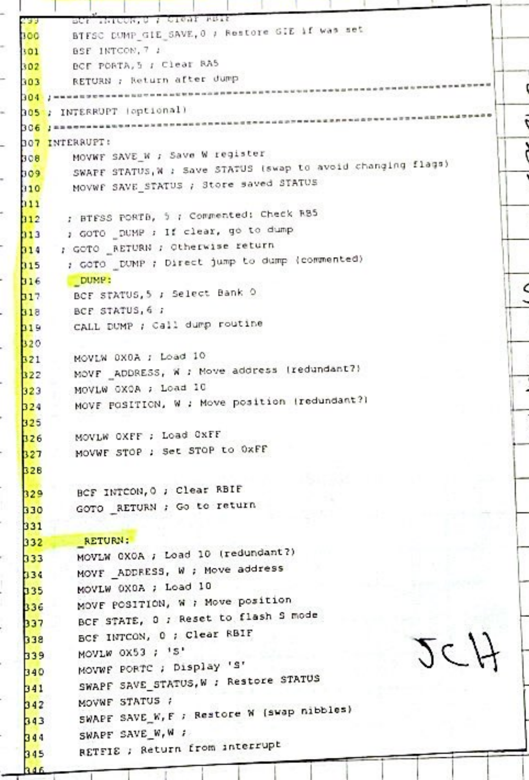
All highlighted code is for the Read operation

JCH

The Interrupt primarily handles the reading and writing to the EEPROM. Each time data is written the position register increments

Lab 10 EEPROM

Figure 10-8 ISR



```
    BCF INTCON,0 ; Clear RBIF
    BTFSC DUMP_GIE_SAVE,0 ; Restore GIE if was set
    BSF INTCON,7 ;
    BCF PORTA,5 ; Clear RA5
    RETURN ; Return after dump
;-----------------------------------------------------------------
; INTERRUPT (optional)
;-----------------------------------------------------------------
INTERRUPT:
    MOVWF SAVE_W ; Save W register
    SWAPF STATUS,W ; Save STATUS (swap to avoid changing flags)
    MOVWF SAVE_STATUS ; Store saved STATUS

   ; BTFSS PORTB, 5 ; Commented: Check RB5
   ; GOTO _DUMP ; If clear, go to dump
  ; GOTO _RETURN ; Otherwise return
   ; GOTO _DUMP ; Direct jump to dump (commented)
    _DUMP:
    BCF STATUS,5 ; Select Bank 0
    BCF STATUS,6 ;
    CALL DUMP ; Call dump routine

    MOVLW 0X0A ; Load 10
    MOVF _ADDRESS, W ; Move address (redundant?)
    MOVLW 0X0A ; Load 10
    MOVF POSITION, W ; Move position (redundant?)

    MOVLW 0XFF ; Load 0xFF
    MOVWF STOP ; Set STOP to 0xFF


    BCF INTCON,0 ; Clear RBIF
    GOTO _RETURN ; Go to return

     _RETURN:
    MOVLW 0X0A ; Load 10 (redundant?)
    MOVF _ADDRESS, W ; Move address
    MOVLW 0X0A ; Load 10
    MOVF POSITION, W ; Move position
    BCF STATE, 0 ; Reset to flash S mode
    BCF INTCON, 0 ; Clear RBIF
    MOVLW 0X53 ; 'S'
    MOVWF PORTC ; Display 'S'
    SWAPF SAVE_STATUS,W ; Restore STATUS
    MOVWF STATUS ;
    SWAPF SAVE_W,F ; Restore W (swap nibbles)
    SWAPF SAVE_W,W ;
    RETFIE ; Return from interrupt
```

JCH

Conclusion

JCH
11/6/25
8:00-8:30

This lab Successfully demonstrates EEPROM operation. So all the data written will be stored until over written when power is completely shut off. The Microchip datasheet can also be referenced to ensure that operation is correct