

Micros Assignment - Oscar Smith

by Oscar Smith

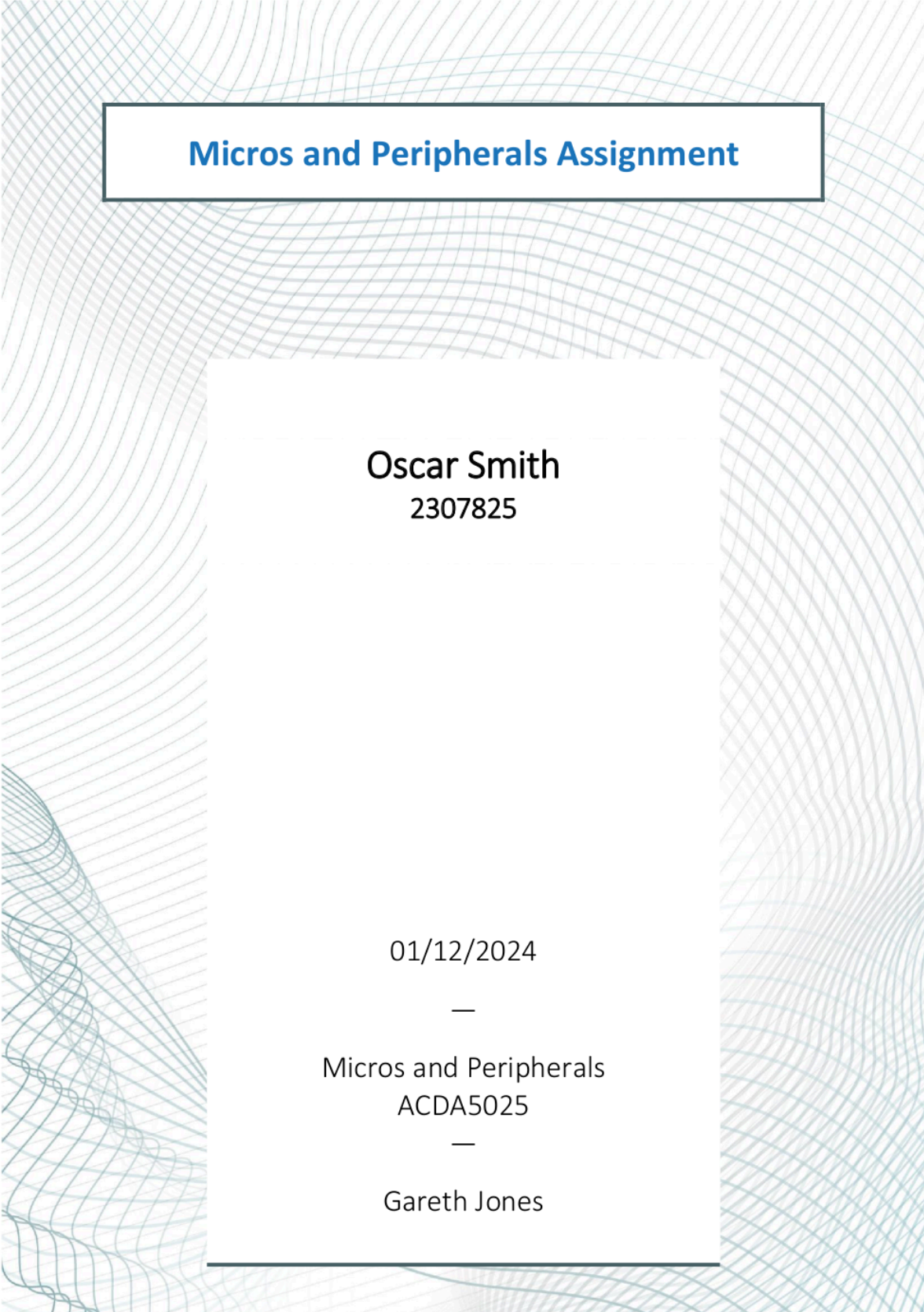
Submission date: 14-Jan-2025 01:44PM (UTC+0000)

Submission ID: 248415910

File name: 51511_Oscar_Smith_Micros_Assignment_-_Oscar_Smith_621295_1441454621.docx (3.81M)

Word count: 1780

Character count: 8612



Micros and Peripherals Assignment

Oscar Smith
2307825

01/12/2024

—

Micros and Peripherals
ACDA5025

—

Gareth Jones

Table of Contents:

1. Introduction.....	
2. Circuit.....	
2.1. Overview of Circuit.....	
2.2. Parts and Specifications.....	
2.3. PIC Overview.....	
3. Code.....	
3.1. 7 Segment Hexadecimal Explanation.....	
3.2. Code segment 1.....	
3.3. Code segment 2.....	
3.4. Code segment 3.....	
3.5. Code segment 4.....	
3.6. Code segment 5.....	
4. References and Figures.....	

1. Introduction:

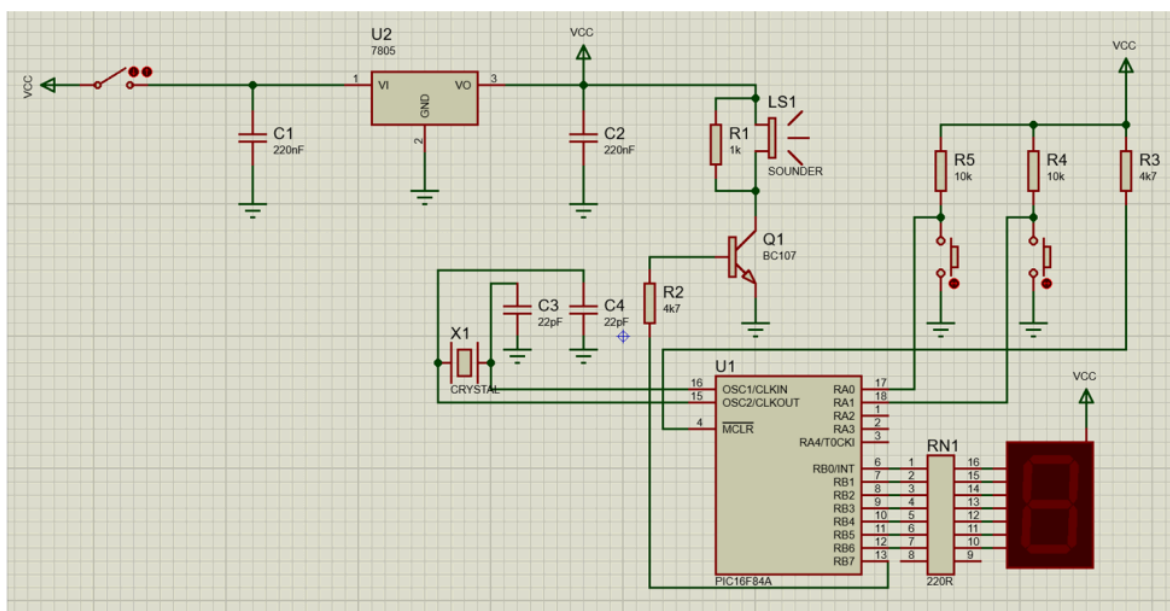
This written report will describe the process of coding a PIC microcontroller circuit in the simulation software proteus.

The report will first provide an overview of the circuit and its components including a brief overview of the PIC16F84A microcontroller that is at the heart of the circuit, it will then cover the process of coding each module for the tasks provided.

The code segments are separated into tasks 1-6 and each section in this report will contain both the annotated assembly code and a flowchart explaining what the code is doing.

2.1. Circuit Overview:

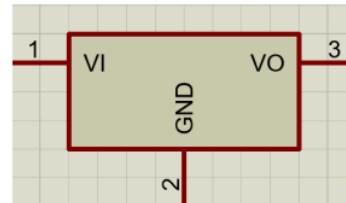
The circuit used in this assignment (shown below), consists of the PIC16F84A microcontroller as the “brain” of the circuit, 2 buttons used as input devices to control various aspects of the circuit and a sounder and 7 segment display used as output devices. This circuit creates a good base to code and test different functions and practice a number of different coding techniques.



2.2. Parts and Specifications:

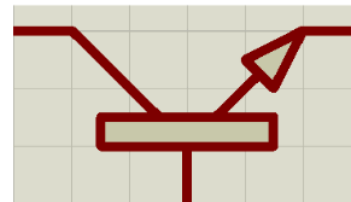
7085 power supply regulator:

This power supply regulator is included in the circuit to ensure that the sensitive microcontroller receives a consistent voltage regardless of any fluctuations in the supply voltage, in this case the voltage regulator can receive between 7V and 25V and will output 5V, with the excess voltage being dissipated as heat. [1] (Fig.1.)



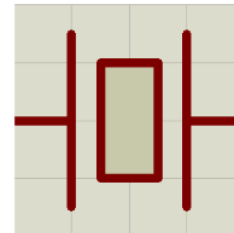
BC107 bipolar transistor:

This bipolar transistor is used for general purpose switching in this circuit, in this case when a current is applied to the base through pin B7 it allows current to flow through the sounder.



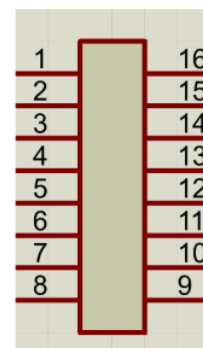
Quartz crystal oscillator:

A crystal oscillator amplifies and feeds back a voltage until a stable oscillation is achieved, in this circuit an oscillator is used to provide a steady clock signal to the microcontroller. [2]



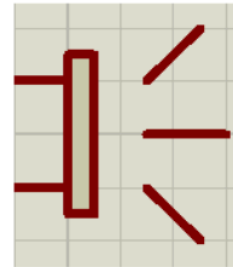
RX8 resistor bank:

The resistor bank is used instead of 8 separate resistors purely for ease of use and to make the circuit easier to read, it serves the same purpose as 8 single resistors.

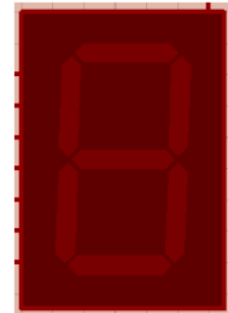


Piezo Sounder:

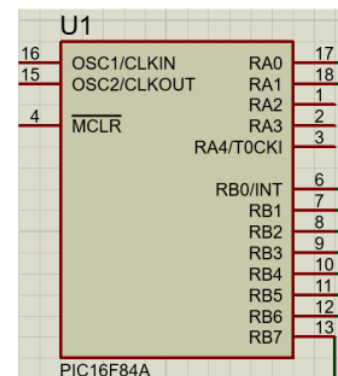
The piezo sounder is a buzzer that works by quickly deforming a piezoelectric material by running a voltage across it, doing this at specific frequencies creates a buzzing sound. In this circuit it is used as an output device.

**7 Seg Com Anode:**

This 7 segment LED display is used in the circuit as an output device, it can be used to display the numbers 0-9 and the letters A-Z.

**2.3. PIC Overview:**

The PIC microcontroller used in this circuit is a PIC16F84A, it is an 18-pin 8-bit microcontroller with 1024 words of program memory, 68 bytes of RAM (or Random Access Memory) and 64 bytes of EEPROM (or Electrically Erasable Read Only Memory). The PIC has 13 I/O pins split into two “ports”, port A and port B. [3]



3.1. 7 Segment Hexadecimal Explanation:

To display the characters 0-F on the 7 segment display it is easiest to first convert them to binary, then they can be forwarded to port B which will display them on the 7 segment display without having to worry about special characters or hexadecimal conversion.

Due to the LED display only having 7 segments, and the way that binary is read from back to front, the first bit in the sequence will always be 1. This means that pin B7, which is not connected to the LED display, will be low. The binary conversions of the characters are shown below:

11000000 = 0
11001111 = 1
10100100 = 2
10110000 = 3
10011001 = 4
10010010 = 5
10000010 = 6
11111000 = 7
10000000 = 8
10011000 = 9
10001000 = A
10000011 = B
11000110 = C
10100001 = D
10000110 = E
10001110 = F

3.2. Code Segment 1, 0-F Count:

The code for the 0-F count is shown in (Fig.2.), the pseudo-code is as follows:

- Start "get-count" function
- Move literal binary for character 0 to the W register
- Move contents of W register to port B which will display on the 7 segment LED
- Call pause subroutine to delay program by 200ms
- Repeat with binary for letters 0-F
- Once all letters have been displayed go to "get-count" to repeat

3.3. Code Segment 2, L-R Buttons:

The code for the L-R buttons is shown in (Fig.3.), the pseudo-code is as follows:

"Main" function:

- Perform a bit test on pin A0 (button 1) to see whether the button is pressed
- If the button is pressed go to "Display_L" function
- If the button is not pressed continue with the "Main" function
- Perform a bit test on pin A1 (button 2) to see whether the button is pressed
- If the button is pressed go to "Display_r" function
- If the button is not pressed continue with the "Main" function
- Move literal binary "1111111" to the W register
- Move the contents of the W register to port B in order to clear the 7 segment display between button presses
- Call pause subroutine to delay program by 200ms
- Go to "Main" function to repeat

"Display_L" function:

- Move literal binary for the letter L to the W register
- Move the contents of the W register to port B to display "L"
- Call pause subroutine to delay the program by 200ms
- Go to "Main" function

"Display_r" function:

- Move literal binary for the letter R to the W register
- Move the contents of the W register to port B to display "r"
- Call pause subroutine to delay the program by 200ms
- Go to "Main" function

3.4. Code Segment 3, On-Off Buttons:

The code for the On-Off buttons is shown in (Fig.4.), the pseudo-code is as follows:

“Main” function:

- Perform a bit test on pin A0 (button 1) to see whether the button is pressed
- If the button is pressed go to “ON” function
- If the button is not pressed continue with the “Main” function
- Perform a bit test on pin A1 (button 2) to see whether the button is pressed
- If the button is pressed go to “OFF” function
- If the button is not pressed continue with the “Main” function
- Go to “Main” function to repeat

“ON” function:

- Move literal binary “10000000” (which lights all of the 7 LEDs) to the W register
- Move the contents of the W register to port B to turn LEDs on
- Perform a bit test on pin A1 (button 2) to see whether the button is pressed
- If the button is pressed go to “Main” function

“OFF” function:

- Move literal binary “11111111” (which turns off all of the 7 LEDs) to the W register
- Move the contents of the W register to port B to turn LEDs off
- Perform a bit test on pin A0 (button 1) to see whether the button is pressed
- If the button is pressed go to “Main” function

3.5. Code Segment 4, Button Counter:

The code for the button counter is shown in (Fig.5.), the pseudo-code is as follows:

For space saving purposes I will do pseudo-code for the display4, 5 and 6 functions, they all follow the same principle

“display4” function:

- Move literal binary for the number 4 to the W register
- Move the contents of the W register to port B to display the number 4
- Call the pause subroutine twice to delay for 400ms
- Perform a bit test on pin A0 (button 1) to see whether the button is pressed
- If the button is pressed go to “display3” function
- If the button is not pressed continue with the “display4” function
- Perform a bit test on pin A1 (button 2) to see whether the button is pressed
- If the button is pressed go to “display5” function
- If not pressed go to “display4” function to continue displaying 4 and repeat the function

“display5” function:

- Move literal binary for the number 5 to the W register
- Move the contents of the W register to port B to display the number 5
- Call the pause subroutine twice to delay for 400ms
- Perform a bit test on pin A0 (button 1) to see whether the button is pressed
- If the button is pressed go to “display4” function
- If the button is not pressed continue with the “display5” function
- Perform a bit test on pin A1 (button 2) to see whether the button is pressed
- If the button is pressed go to “display6” function
- If not pressed go to “display5” function to continue displaying 5 and repeat the function

“display6” function:

- Move literal binary for the number 6 to the W register
- Move the contents of the W register to port B to display the number 6
- Call the pause subroutine twice to delay for 400ms
- Perform a bit test on pin A0 (button 1) to see whether the button is pressed
- If the button is pressed go to “display5” function
- If the button is not pressed continue with the “display6” function
- Perform a bit test on pin A1 (button 2) to see whether the button is pressed
- If the button is pressed go to “display7” function
- If not pressed go to “display6” function to continue displaying 6 and repeat the function

3.6. Code Segment 5, Sounder:

The code for the Sounder is shown in (Fig.6.), the pseudo-code is as follows:

“loop” function:

- Perform a bit test on pin A0 to see whether the button is pressed
- If the button is pressed move literal binary “01111111” to the W register
- Move the contents of the W register to port B to set pin B7 (the pin connected to the sounder) high
- Call the pause subroutine twice to delay for 400ms
- Go to “loop” function to repeat

4. Figures:

(Fig.1.) Datasheet for 7805 Voltage Regulator showing the min-max operating voltages and the output voltage:

recommended operating conditions

		MIN	MAX	UNIT
V_I Input voltage	$\mu A7805C$	7	25	V
	$\mu A7808C$	10.5	25	
	$\mu A7810C$	12.5	28	
	$\mu A7812C$	14.5	30	
	$\mu A7815C$	17.5	30	
	$\mu A7824C$	27	38	
I_O Output current			1.5	A
T_J Operating virtual junction temperature	$\mu A7800C$ series	0	125	$^{\circ}C$

electrical characteristics at specified virtual junction temperature, $V_I = 10\text{ V}$, $I_O = 500\text{ mA}$ (unless otherwise noted)

PARAMETER	TEST CONDITIONS	$T_J^{\dagger}$	$\mu A7805C$			UNIT
			MIN	TYP	MAX	
Output voltage	$I_O = 5\text{ mA to }1\text{ A},$ $P_D \leq 15\text{ W}$	25 $^{\circ}C$	4.8	5	5.2	V
		0 $^{\circ}C$ to 125 $^{\circ}C$	4.75		5.25	

(Fig.2.) Code Segment 1, 0-F count

```

PGM   code
Start
    bcf   status,RP0      ; clear bank selection bits in the status reg
    bcf   status,RP1

    bsf   status,RP0      ; select bank 1
    movlw 0x00            ; load w with 0x00
    movwf trisb           ; load it into trisb to make portb all outputs
    bcf   status,RP0      ; reselect bank 0

    clrf  portB           ; clear all lines low
    count 0               ; initialize counter to 0

get_count
    movf  count, W        ; move count to W

    movlw b'11001111'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10100100'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10110000'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10011001'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10010010'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10000010'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'11111000'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10000000'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10011000'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10001000'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10000011'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'11000110'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10100001'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10000110'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    movlw b'10001110'
    movwf portB           ; move W to port B to update the display
    call  pause           ; delay 200ms by subroutine

    goto  get_count       ; repeat forever

```

(Fig.3.) Code Segment, L-R Buttons

```

PGM   code
Start
    bcf     status,RP0      ; clear bank selection bits in the status reg
    bcf     status,RP1

    bsf     status,RP0      ; select bank 1
    movlw   0x00            ; Load w with 0x00
    movwf   trisb           ; Load it into trisb to make portb all outputs
    bcf     status,RP0      ; reselect bank 0

    clrf    portB           ; clear all lines low
    clrf    count           ; initialize counter to 0

MAIN:
    ; Check if RA0 is Low
    BTFSS   PORTA, 0        ; Check if RA0 is set (high)
    GOTO    DISPLAY_L       ; If RA0 is Low, jump to display 'L'

    BTFSS   PORTA, 1        ; Check if RA1 is set (high)
    GOTO    DISPLAY_r       ; If RA0 is Low, jump to display 'L'

    movlw   b'11111111'     ; Clear screen in between button presses
    movwf   portB
    call    pause
    GOTO    MAIN            ; Loop back to check RA0 again

DISPLAY_L:
    movlw   b'11000111'
    movwf   portB           ; move W to port B to update the display
    call    pause           ; delay 200ms by subroutine
    GOTO    MAIN            ; Loop back to check RA0 again

DISPLAY_r:
    movlw   b'11001110'
    movwf   portB           ; move W to port B to update the display
    call    pause           ; delay 200ms by subroutine
    GOTO    MAIN            ; Loop back to check RA0 again

```

(Fig.4.) Code Segment, On-Off Buttons

```

PGM   code
Start
    bcf     status,RP0      ; clear bank selection bits in the status reg
    bcf     status,RP1

    bsf     status,RP0      ; select bank 1
    movlw   0x00            ; Load w with 0x00
    movwf   trisb           ; Load it into trisb to make portb all outputs
    bcf     status,RP0      ; reselect bank 0

    clrf    portB           ; clear all lines low
    clrf    MAIN            ; initialize counter to 0

MAIN:
    ; Check if RA0 is Low
    BTFSS   PORTA, 0        ; Check if RA0 is set (high)
    GOTO    ON              ; If RA0 is Low, jump to ON

    BTFSS   PORTA, 1        ; Check if RA1 is set (high)
    GOTO    OFF             ; If RA0 is Low, jump to OFF

    GOTO    MAIN            ; Loop back to check RA0 again

ON:
    movlw   b'10000000'
    movwf   portB           ; move W to port B to update the display

    BTFSC   PORTA, 1        ; Check if RA1 is set (high)
    GOTO    MAIN            ; Loop back to check RA0 again

OFF:
    movlw   b'11111111'
    movwf   portB           ; move W to port B to update the display

    BTFSC   PORTA, 0        ; Check if RA0 is set (high)
    GOTO    MAIN            ; Loop back to check RA0 again

```

(Fig.5.) Code Segment, Button Counter

```

PGM code
Start
    bcf status,RP0 ; clear bank selection bits in the status reg
    bcf status,RP1

    bsf status,RP0 ; select bank 1
    movlw 0x00 ; load w with 0x00
    movwf trisb ; load it into trisb to make portb all outputs
    bcf status,RP0 ; reselect bank 0

    clrf portB ; clear all lines low
    clrf count ; initialize counter to 0

    goto display5

display0
    movlw b'11000000'
    movwf portB
    call pause
    call pause

    btfss portA, 1
    goto display1
    call pause

    btfss portA, 0
    goto display0
    call pause

    goto display0

display1
    movlw b'11001111'
    movwf portB
    call pause
    call pause

    btfss portA, 1
    goto display2
    call pause

    btfss portA, 0
    goto display0
    call pause

    goto display1

display2
    movlw b'10100100'
    movwf portB
    call pause
    call pause

    btfss portA, 1
    goto display3
    call pause

    btfss portA, 0
    goto display1
    call pause

    goto display2

display3
    movlw b'10110000'
    movwf portB
    call pause
    call pause

    btfss portA, 1
    goto display4
    call pause

    btfss portA, 0
    goto display2
    call pause

    goto display3

display4
    movlw b'10011001'
    movwf portB
    call pause
    call pause

    btfss portA, 1
    goto display5
    call pause

    btfss portA, 0
    goto display3
    call pause

    goto display4

display5
    movlw b'10010010'
    movwf portB
    call pause
    call pause

    btfss portA, 1
    goto display6
    call pause

    btfss portA, 0
    goto display4
    call pause

    goto display5

display6
    movlw b'10000010'
    movwf portB
    call pause
    call pause

    btfss portA, 1
    goto display7
    call pause

    btfss portA, 0
    goto display5
    call pause

    goto display6

display7
    movlw b'11111000'
    movwf portB
    call pause
    call pause

    btfss portA, 1
    goto display8
    call pause

    btfss portA, 0
    goto display6
    call pause

    goto display7

display8
    movlw b'10000000'
    movwf portB
    call pause
    call pause

    btfss portA, 1
    goto display9
    call pause

    btfss portA, 0
    goto display7
    call pause

    goto display8

display9
    movlw b'10011000'
    movwf portB
    call pause
    call pause

    btfss portA, 1
    goto display9
    call pause

    btfss portA, 0
    goto display8
    call pause

    goto display9

```

(Fig.6.) Code Segment, Sounder

```
PGM      code
Start
    bcf    status,RP0      ; clear bank selection bits in the status reg
    bcf    status,RP1

    bsf    status,RP0      ; select bank 1
    movlw  0x00            ; load w with 0x00
    movwf  trisb           ; load it into trisb to make portb all outputs
    bcf    status,RP0      ; reselect bank 0

    clrf   portb           ; clear all lines low
    clrf   count          ; initialize counter to 0

loop
    btfss  portA, 0
    movlw  b'01111111'
    movwf  portb
    call   pause
    call   pause
    goto   loop

;PGM      code
;PGM      code
```

4. References:

- [1] "Positive-Voltage Regulators," [Online]. Available:
<https://cdn.sparkfun.com/assets/1/7/7/3/2/LM7805.pdf>.
- [2] "Quartz Crystal Oscillators," [Online]. Available: <https://www.electronics-tutorials.ws/oscillator/crystal.html>.
- [3] "PIC16F84A Datasheet," [Online]. Available:
<https://ww1.microchip.com/downloads/en/DeviceDoc/35007b.pdf>.

Micros Assignment - Oscar Smith

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

50/100

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

OSCAR SMITH

250

ENGAGEMENT WITH LITERATURE

 Only three sources cited - need to include more than this and so try to cover all relevant aspects

30

KNOWLEDGE AND UNDERSTANDING

 The description of the components while fair and generally without error, could have been done in far more depth by referring to additional research material.

55

COGNITIVE AND INTELLECTUAL

 The programming tasks use a fairly conventional 'brute force' approach, so as this section is for flair and innovation, the mark here is correspondingly low

35

PRACTICAL APPLICATION

 The circuit is simulated fine.
The first tasks look as if they will work, but the

60

sounder in task5 not so much, and task 6 not attempted.

TRANSFERABLE SKILLS



Fairly tidy in its presentation and use of diagrams.

70