

Micros, Peripherals & Interfaces Assignment Submission

by Sam Peppard

Submission date: 14-Jan-2025 12:44PM (UTC+0000)

Submission ID: 248406296

File name:

52079_Sam_Peppard_Micros__Peripherals__Interfaces_Assignment_Submission_621295_1645524176.docx
(728.35K)

Word count: 2782

Character count: 12737



Using & programming the 7 Segment display prototype Circuit

*L4 – Micros Peripherals & Interfaces
ACDA4004
Embedded Electronics Systems
Samuel Peppard
2308344*

Contents

List of Figures	3
Section 1- Creating a Proteus Simulation.....	4
1.1 Proteus Simulation Design Process	4
1.2 Component Information.....	5
Section 2 – How to display hexadecimal on 7-segment display	8
2.1 – How the PIC displays on the 7-segment.....	8
2.2 – How the 7-Segment display works.....	8
2.3 - 1-F in Binary to display on the 7-segment display	8
Section 3	11
3.1 Task 1 – Count from 0 to F in hexadecimal.....	11
3.2 Task 2 – Display L and R when buttons are pressed.....	13
3.3 Task 3 – On and Off using buttons	15
3.4 Task 4 – Counting up and down using buttons	17
References.....	18
Bibliography	19

List of Figures

Figure 1 - Schematic design of Electronic Circuit used in this project	4
Figure 2 - PIC16F84A	5
Figure 3 - 7-segment Display	5
Figure 4 - Crystal Oscillator	5
Figure 5 - an image of a button.....	6
Figure 6 - An image of a Piezo Buzzer.....	6
Figure 7 - Image on an NPN transistor.....	6
Figure 8 - Image of a capacitor	7
Figure 9 - Image of a 7-segment display and letters corresponding to the LED	8
Figure 10 - Flow chart of order of code for task 1	11
Figure 11 - Assembly code for task 1	12
Figure 12 - Flowchart of information for Task 2	13
Figure 13 - Main section of code for Task 2.....	14
Figure 14 - A flowchart for Task 3.....	15
Figure 15 – Main code for Task 3.....	16
Figure 16 - Flowchart for Task 4.....	17

Section 1- Creating a Proteus Simulation

1.1 Proteus Simulation Design Process

When designing the Circuit, you need to take into account multiple considerations.

These include:

1. What to display the circuit on?
2. How you will display from the circuit
3. How to power the circuit
4. Any additional components needed to allow it to perform all tasks you need.

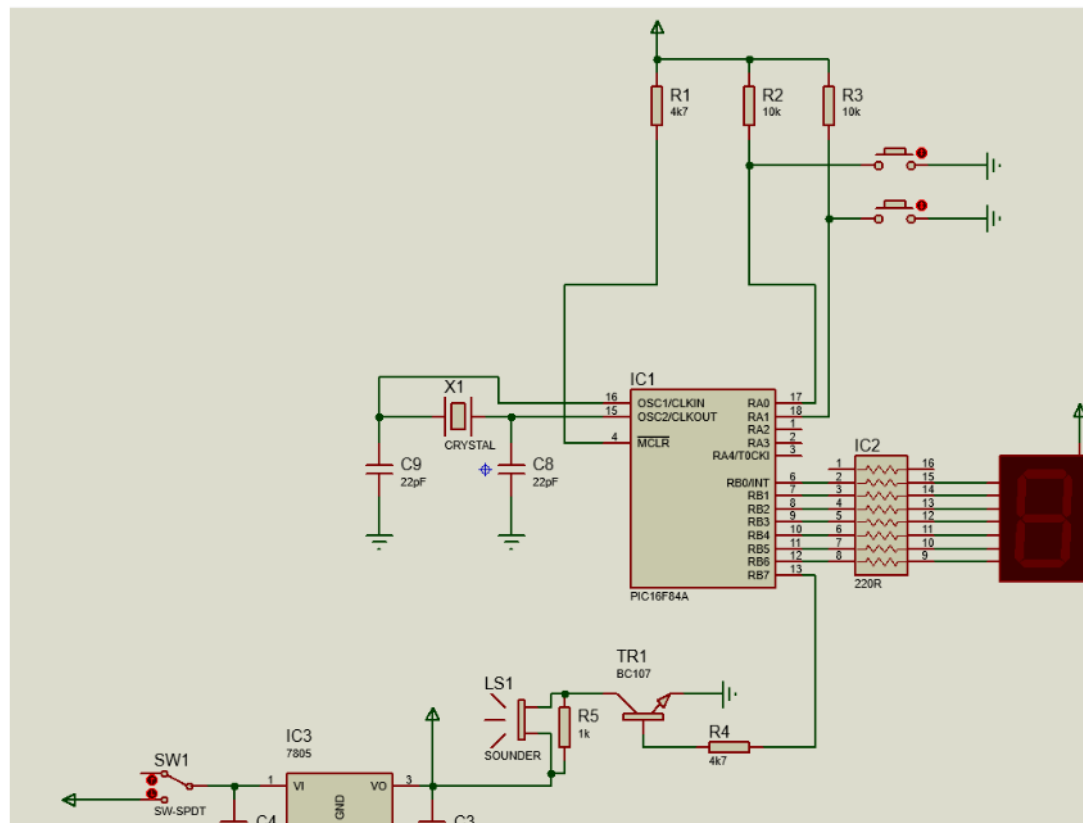


Figure 1 - Schematic design of Electronic Circuit used in this project

Looking at Figure 1 above you can see that all of these considerations have been utilised.

The circuit will display the values on the 7-segment display on the right side. To ensure that the 7-segment display shows the correct values PIC16F84. This PIC will be able to utilise Assembly code to conduct all operations. It works as almost the “brain” of the Circuit.

The next factor to tackle would be the powering within the circuit. The circuit is powered from 2 sections. Firstly, the bottom left where it goes through a switch (SW1) and into a Voltage regulator (IC3) and additionally from the top which then passes through resistors and into the PIC. If the buttons are pressed it can also then go through that junction which can send power into pin 17 and 18, this will be useful coming up to tasks 2 and 3.

The final key consideration is additional components that will need to be used to ensure this circuit can perform all of the tasks it is presented with. To make sure that this is done there is 2 buttons which will be used in tasks 2 and 3 whilst also having a sounder which will be used in tasks 5 and 6.

1.2 Component Information

1.2.1 PIC16F84A

PIC16f84A is an 18-pin Enhanced FLASH/EEPROM 8-Bit Microcontroller [1]. This PIC is a reduced instruction set computer (RISC) with only 35 instructions used within its software. With 13 I/O pins it allows for multiple outputs which can assist the 7-segment display. With 1750 bytes of program memory and speeds of 5 million instructions per second [2]. This PIC will be ample for the simple programs that will be involved within this project.



Figure 2 - PIC16F84A

1.2.2 7-Segment Display

A 7-segment display is 7 LEDs arranged in a rectangular form with a horizontal LED in the center; As shown in Figure 3. This component can allow you to display digits and letters excluding M, V, X and Z. This therefore means that it is possible to display 0-F within hexadecimal on just one 7-segment display common use cases for a 7-segment display can be electronic devices which need to display numbers for example a clock or a counter.

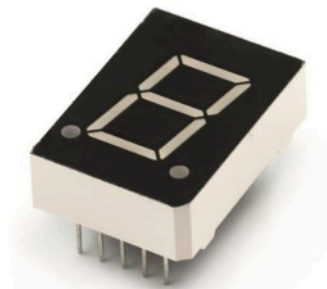


Figure 3 - 7-segment Display

1.2.3 Crystal Oscillator

Crystal oscillators are used in circuits to produce a highly accurate and stable frequency [3]. the component works due to the quartz inside resonating when AC is applied to the circuit. This is primarily due to the Piezoelectric Effect. This therefore means that the circuit is able to make a frequency which then can be fed into a circuit which can assist other components which require frequency. For example, in the circuit that's being used PIC16F84A requires a frequency to be able to effectively run.



Figure 4 - Crystal Oscillator

1.2.4 Buttons

Buttons are components within a circuit which at a fundamental level is used to momentarily turn on and off a section of the circuit. The way it works is that when a button is pressed it completes the circuit. In context to this specific circuit, it allows you to decide if you want to power pin 17 and pin 18. This can therefore allow us to run specific functions when the button is pressed. for example, in section 2 of the code when one button is pressed the 7-segment display will show the letter L and when the other is pressed it will show the letter R. Additionally with buttons you can make them "latch" in brief this means that when the button is pressed once there will be no change until the circuit receives an additional change.



Figure 5 - an image of a button

1.2.5 Sounder

A sounder does exactly what the name says, it is a component in which produces a sound when voltage is applied. There are two main types of sounders. These are magnetic sounders and piezo sounders. Magnetic sounder works by using alternating current inside a magnetic field in comparison to Piezo sounders which contain materials which change shape when there is an alternating voltage [4]. The sounder in this circuit is specifically a piezo sounder; this is due to its low current and reliance on voltage.



Figure 6 - An image of a Piezo Buzzer

1.2.6 Voltage regulator

Voltage regulators are used within circuits to change the voltage in a circuit. They are mainly used near the input voltage to raise or lower potential difference to assist the desired components within a circuit. This component includes 3 pins, Voltage in, Voltage out and Ground.

1.2.7 Transistor

Transistors are used to control the flow of electric current due to its layered structure. It also amplifies the current whilst acting as a switch. Specifically, in this circuit is an NPN (Negative-Positive-Negative) transistor. This therefore means that current flows from the collector to the emitter. It operates in three key modes these are cut-off (no current flow), saturation (full current flow), and active (current amplification) [5].



Figure 7 - Image on an NPN transistor

1.2.8 Resistor

Resistors in brief terms resist current. They can be used to control the current flow within a circuit by limiting the amount that passes. Resistance and current are inversely proportional to each other. This means that the higher the resistance the lower the current flow will be. Resistors have been used in this circuit primarily to limit the current flowing into the PIC.

1.2.9 Capacitors

Capacitors are components which store electrical charge in the circuit. Capacitors have multiple uses within a circuit. Within this circuit they are used before ground. The reasoning for this is to reduce noise which have the ability to affect the performance of the IC (Integrated circuit). In summary capacitors within this circuit can ensure that the PIC performs as well as it can.



Figure 8 - Image of a capacitor

1.2.10 Switch

A switch is a very simple electronic component. The role of a switch is to change which wires are connected. They are very similar to button but instead of temporarily being on they are permanent until pressed again. The switch in this circuit is used to turn on and off the power to the sounder.

1.2.11 Ground

Ground within a circuit is primarily used as a reference point for the return path to ensure that the circuit creates a loop. Without a ground the circuit will not be complete and can be dangerous. Therefore, ensuring all sections of the circuit is grounded is a crucial step in designing a circuit.

Section 2 – How to display hexadecimal on 7-segment display

2.1 – How the PIC displays on the 7-segment

2.2 – How the 7-Segment display works

When the value on the pic is 0 the pic will light that section so if you in put the binary number 0000000 into the 7-segment display, the whole pic will be lit up. Additionally, if you put the binary number 1111111 into the 7-segment display no lights will be on.

The next point to consider is which digit within the 7-digit binary number corresponds to which line on the 7 Segment display in figure 2 you can see that each segment of the 7-segment display is labelled A to G. Each of these letters represent one of the digits within the 7-digit binary number. The order it goes in is reversed though where G is the first digit and A is the final Digit. Therefore, to display anything on the 7-segmentnt display you must consider the binary number of GFEDCBA where the numbers represent either a 1 or 0 depending on if you want the segment lit or not.

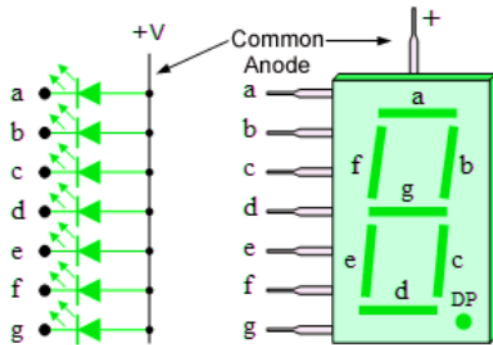
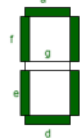
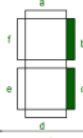
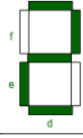
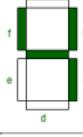
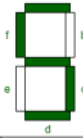
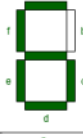
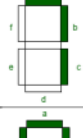


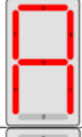
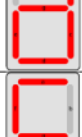
Figure 9 - Image of a 7-segment display and letters corresponding to the LED

2.3 - 1-F in Binary to display on the 7-segment display

Now that the 7-segment display is understood the next consideration is what binary value needs to be entered to display the correct hexadecimal number. As you can see below in table 1 the correct binary and decimal values have been evaluated.

Table 1 - Values that display values from 0-F in hexadecimal

Hex display Value	Binary value to display on 7 Segment display	Decimal equivalent	Hex equivalent	Visual representation of 7 segment display
0	1000000	64	40	
1	1111001	121	79	
2	0100100	36	24	
3	0110000	48	30	
4	0011001	25	19	
5	0010010	18	12	
6	0000010	02	02	
7	1111000	120	78	
8	0000000	00	00	

9	0010000	10	10	
A	0001000	08	08	
B	0000011	03	3	
C	1000110	70	46	
D	0100001	33	21	
E	0000110	06	6	
F	0001110	14	E	

Section 3

3.1 Task 1 – Count from 0 to F in hexadecimal

For this task the PIC is wanting to count in hexadecimal from 0 to F with a pause of 0.2 seconds. The initial plan is to create a flow chart so that it is clear what steps the code needs to take to ensure the task is conducted properly. Below in figure 10 this can be seen.

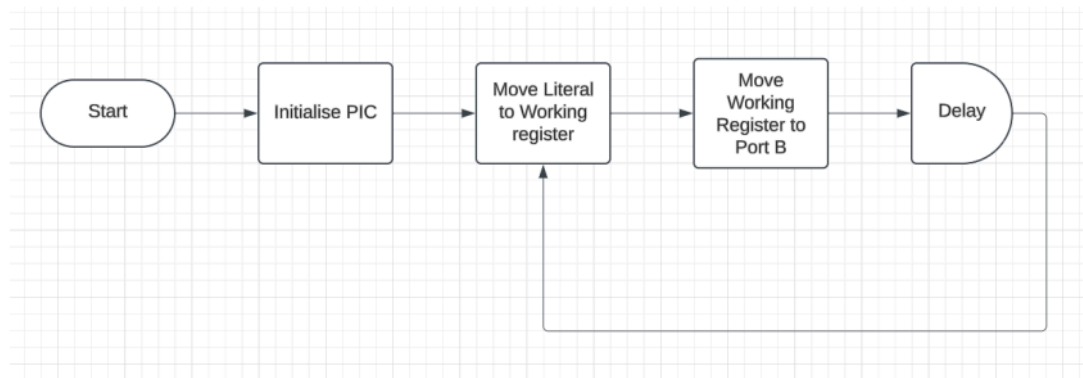


Figure 10 - Flow chart of order of code for task 1

As you can see the initial task that needs to be conducted is initialising the PIC. This prepares the PIC ready to conduct Task 1.

Secondly, the Code will need to move the Literal value (which will be defined) on the 7-segment display (as worked out in section 2 of the report).

After this the code will need to move the value from the working register to port B. this therefore will output the binary value out of all pins associated with port B. therefore displaying the Hexadecimal value on the 7-Segment display.

Finally, there will need to be a delay. This is so that we have time to read the value on the 7-segment display.

After understanding the order, the code needs to work, it was time to write the code. As seen below in figure 11.

```
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

get_count

    movlw 0x40        ; Load 0
    movwf portB       ; move W to port B to update the display

    call pause        ;0.2 ms pause to show updated number

    movlw 0x79        ; Load 1
    movwf portB       ; move W to port B to update the display

    call pause        ;0.2 ms pause to show updated number
```

Figure 11 - Assembly code for task 1

As you can see on the code. The first section is the PIC initialisation. This is in brief clearing all the registers and ensuring there is nothing previously saved which may cause errors with the new code.

After this the line:

`"movlw 0x40"`

Allows us to move the value of 40 (Hexadecimal) to the working register.

After this the code will move from the working register to port B by using the line of code

`"movwf portB"`

Then a pause is created by calling the pause function. These three lines are then repeated for all 16 hexadecimal units.

If you are interested more in the code, then feel free to look into it more in the bibliography.

3.2 Task 2 – Display L and R when buttons are pressed

Task 2 was to make the circuit display an L if one button was pressed and R with the second button. If no buttons are pressed, then nothing will be displayed.

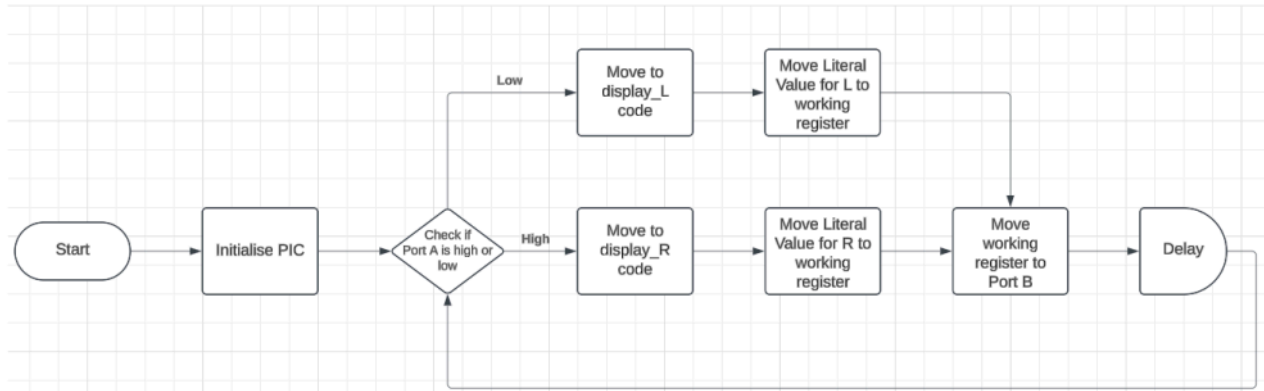


Figure 12 - Flowchart of information for Task 2

As you may be able to tell, there is a lot of similar processes that will occur in task 2 as they did in task 1. There are some additional steps involved. Initially, again, the PIC will be initialised.

After this the PIC will decide if port A is high or Low. This can tell us if button 1 or button 2 has been pressed. If its low the top button (Button 1) has been pressed, if it is high then the bottom button has been pressed (button 2) this will then send us to different sections of code to ensure the correct letter will be displayed.

After this the code follows a similar process to task 1, it will move the literal value for either L or R to the working register which then will move the value to port B which will then display the code on the 7-Segment display.

The next step is to write the code as seen below in figure 12.

```
main
    btfss portA, 0      ;checking if Port A is low
    goto display_L

    btfss portA, 1      ;Checking if Port A is high
    goto display_R

    movlw 0x7F          ;move binary code to W register
    movwf portB

    goto main           ;allows for code to loop on main section until one of the button is pressed

display_L              ;section of code which accounts for the L display
    movlw 0xC7          ;move binary code to W register
    movwf portB         ;move hex number to Port B
    call pause          ;pause to allow code to display
    goto main           ; return back to look for next button press

display_R              ;repeat of display_L code to account for R
    movlw 0xCE          ;move binary code to W register
    movwf portB         ;move hex number to Port B
    call pause          ;pause to allow code to display
    goto main           ;return back to look for next button press
```

Figure 13 - Main section of code for Task 2

The first line in the code is “btfss portA, 0” this is a branching command which is asking if the input = 0 (low). The next line after this is “goto display_L” which tells the code to go the display_L section of the code. Within the display_L section it is a repeat of task 1 where it is moving the Literal to the working register and then from the working register to port B allowing it to display.

There is also the same principle for R except the code asks if the value = 1 (high) instead of 0.

3.3 Task 3 – On and Off using buttons

Task 3 is very similar to task 2, when you press one button the lights will all turn on within the 7-segment display. When you press the other button all the lights will turn off. The main difference between task 2 and task 3 is that the buttons are latching. This means there will be no change until you press another button.

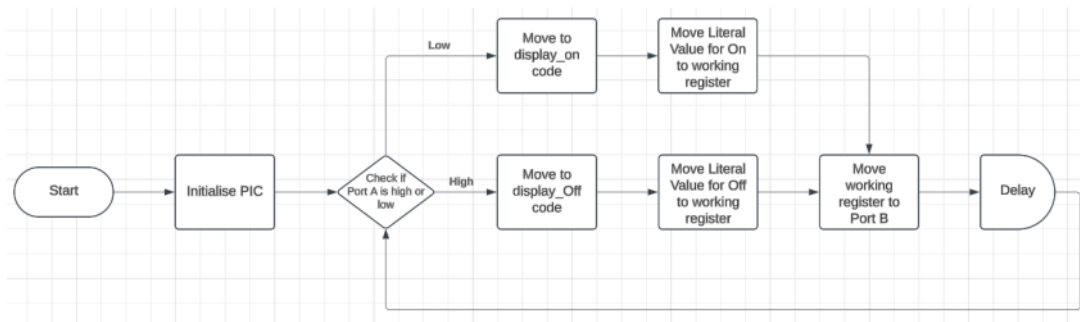


Figure 14 - A flowchart for Task 3

The flowchart for task 3 is identical to the one for Task 2.

although the Flowchart is the same the code does have some slight differences. As seen below.

```
main
    btfss portA, 0          ;checking if Port A is Low
    goto display_on

    btfss portA, 1          ;Checking if Port A is high
    goto display_off

    goto main

display_on
    movlw 0x7F              ;move binary code to W register
    movwf portB             ;move hex number to Port B
    call pause              ;pause to allow code to display
    goto main               ;return back to look for next button press

display_off                 ;repeat of display_L code to account for R
    movlw 0x00
    movwf portB
    call pause
    goto main
```

Figure 15 – Main code for Task 3

The key difference between the code in task 2 and task 3 is the fact that two lines have been removed. These lines are:

```
"movlw 0x7F
movwf portB"
```

These two lines are the lines within task 2 that resets the 7-segment display to be blank. Therefore, to make the lights stay on removing these lines will cause this to happen.

3.4 Task 4 – Counting up and down using buttons

For task 4, the circuit needed to start at the number 5 and be able to increase the value by 1 when one button is pressed and count down when the second button is pressed. The flowchart for this is much larger than the previous task, this is due to there being more steps in the process.

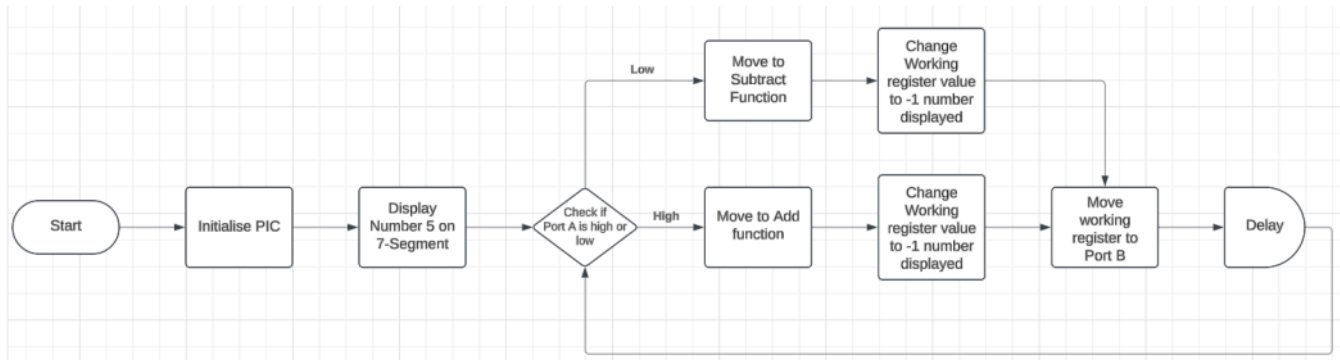


Figure 16 - Flowchart for Task 4

As seen above in Figure 16, the first step is to initialise the PIC, after this the circuit needed to display 5 on the 7-Segment display.

Next the circuit needs to check the input from the buttons by looking if the port is high or low. If high, the circuit needed to move to the add function and then change the value on the 7-segment display, similar logic applies for if Pin A is low; the circuit would decrease the number on the 7-segment display. It will repeat this section allowing you to count up and down.

Unfortunately, the code did not work so there isn't any working code for this. But the theory of how it works would still apply.

References

- [1] Microchip Technology Inc, "PIC16F84A".
- [2] "The PIC16F84A – A Beginners Microcontroller," teach me micro, 15 September 2015. [Online]. Available: <https://www.teachmemicro.com/pic16f84a-beginner-microcontroller/>.
- [3] A. Satheesh, "Crystal Oscillator: Frequency, Circuit and Working," Soldering Mind, 13 June 2023. [Online]. Available: <https://solderingmind.com/crystal-oscillator-frequency-circuit-and-working/>.
- [4] P. Kendall, "How Do Piezo Buzzers Work?," RS, 30 October 2023. [Online]. Available: <https://uk.rs-online.com/web/content/discovery/ideas-and-advice/how-do-piezo-buzzers-work>.
- [5] Electrical 4U, "NPN Transistor: What is it? (Symbol & Working Principle)," Electrical4U, 2 May 2024. [Online].

Bibliography



Assignment_Task
1.pdsprj



Assignment_Task
2.pdsprj



Assignment_Task
3.pdsprj

Micros, Peripherals & Interfaces Assignment Submission

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

54/100

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

GRADING FORM: GJ_2019

SAM PEPPARD**270****ENGAGEMENT WITH LITERATURE**

Only 5 reference sources - should have included more than this so that all/most aspects of the circuit/technologies are covered.
Also - incomplete reference data in places (such as date accessed)

45**KNOWLEDGE AND UNDERSTANDING**

What is written is reasonable, but a higher mark would have been obtained with more detail and more reference to additional reference material.

65**COGNITIVE AND INTELLECTUAL**

A fairly conventional approach to the software problems so a low mark here as this is specifically for 'flair and innovation'

35

PRACTICAL APPLICATION



Circuit implemented ok, but only half the tasks attempted.

Good to see the flowcharts included though.

55

TRANSFERABLE SKILLS



Generally well laid out and professionally presented.

70