

Micros Assignment Bailey Freeman

by Bailey Freeman

Submission date: 13-Jan-2025 04:34PM (UTC+0000)

Submission ID: 248296280

File name: 51535_Bailey_Freeman_Micros_Assignment_Bailey_Freeman_621295_1040417754.docx (160.84K)

Word count: 975

Character count: 4995

Micros, Peripherals and Interfaces

**Using and programming the 7-segment display
prototype circuit**

BEng Embedded Electronics systems

Module Code: ACCA4004

Name: Bailey Freeman

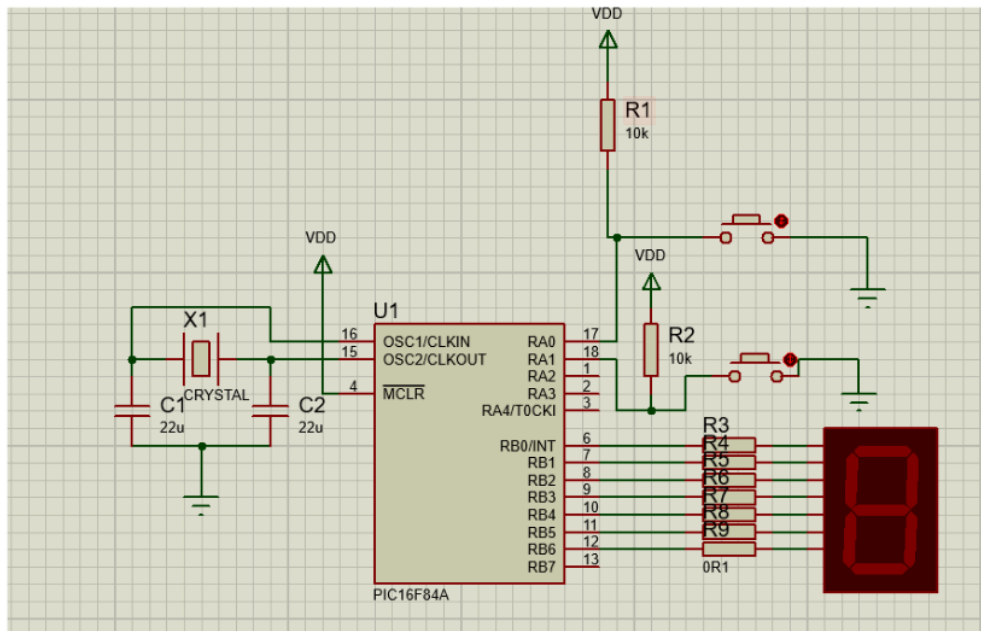
Date: 13/01/2025

Contents

The circuit.....	3
The programme plan.....	5
The programme.....	8

The Circuit

The circuit used for this assignment mainly focuses around a PIC16F84A which is an 8-bit microcontroller that can be used to control many different things but, in this case, it is being used to take inputs and operate a seven segment LED display. A circuit diagram is shown below.



It is important to understand the individual components in the circuit in order to understand the whole circuit, so the following section is a breakdown of each component in the circuit.

The Microcontroller:

The PIC16F84A microcontroller (U1) is the brain of the circuit. It operates by executing a pre-written programme stored in its memory, allowing it to process inputs and control outputs.

This microcontroller includes two ports:

Port A (RA0-RA3) is configured as input for receiving user signals via push-button switches.

Port B (RB0-RB7) is configured as output, driving the 7-segment display to show numerical or symbolic results.

The microcontroller relies on a clock signal to execute instructions sequentially. Its performance depends on the external oscillator connected to its OSC1 and OSC2 pins, which ensures a consistent operational frequency.

Clock Circuit:

The clock circuit consists of a crystal oscillator (X1) and two capacitors (C1, C2). The crystal oscillator turns the natural vibration of a quartz crystal into an electrical square wave which can be used to set the clock speed of a microcontroller. The frequency of the oscillator determines how fast the microcontroller processes instructions, which is essential for precise timing in applications such as counters or timers. Faster clock speeds will in turn allow processors to process more data quickly but the speed at which processors can run at is dependant on the quality and specifications of the processor as well as the cooling precautions in place

The capacitors stabilise the oscillations generated by the crystal by filtering electrical noise. Without these capacitors, the oscillator's performance could become erratic, causing timing errors or inconsistent microcontroller behaviour.

Buttons:

The circuit includes two push-button switches connected to input pins on Port A (RA0, RA1). These switches allow user inputs into the system. Each switch is paired with a 10 k Ω pull-down resistor (R1, R2) to ensure the corresponding input pin reads a HIGH (logic 1) signal when the switch is not pressed. When a switch is pressed, it creates a direct connection to ground, making the pin read a LOW (logic 0) signal. This provides binary input signals that the microcontroller can process.

7 Segment LED:

The output of the circuit is a 7-segment display connected to Port B (RB0 to RB7). Each LED which makes up one segment of the display corresponds to one of the microcontroller's output pins, allowing it to control which segments are lit. By sending appropriate HIGH and LOW signals to these pins, the microcontroller can display numbers or certain letters on the 7-segment display.

To protect the display, current-limiting resistors (R3-R9) are placed between the microcontroller pins and the segments. These resistors prevent excessive current from damaging the LEDs in the display.

Programming the Microcontroller

Now that the circuit and the components within it have been explained, this report can go into the programming. The circuit needs to be programmed to display a count from 0 up to F as hexadecimal and have a delay of roughly 200ms. To do this you'd first have to find the binary output to give the 7-segment display to display each character you want and note this down so you can call it back later. You would then have to initialise the microprocessor and set up its input and output ports. Also, any variables that will be used in the code should be declared. Then you'd need to create a loop for the count to go in which can be created by declaring a function that ends by returning to the top. After that you would need a function that will pass a value to the output and then delay for 200ms. Once you have these things set up you can write code in the loop that will pass the value for which character you want to display and then call the function that outputs that value and delays and then repeat this for each value and allow it to loop back around.

Pseudo code for this programme:

Setup PIC16F84A

Setup portA input

Setup portB output

Count loop

 0 value to active register

 Call print and pause function

 1 value to active register

 Call print and pause function

 2 value to active register

 Call print and pause function

3 value to active register
Call print and pause function
4 value to active register
Call print and pause function
5 value to active register
Call print and pause function
6 value to active register
Call print and pause function
7 value to active register
Call print and pause function
8 value to active register
Call print and pause function
9 value to active register
Call print and pause function
A value to active register
Call print and pause function
B value to active register
Call print and pause function
C value to active register
Call print and pause function
D value to active register
Call print and pause function
E value to active register
Call print and pause function
F value to active register
Call print and pause function
Return to count

```
print and pause function
    output active register value
    delay for 200ms
return
end
```

The assembly programme

This method shown in the pseudo code was used to create a programme with assembly language and is shown in the below screenshots:

```
list    p=16F84      ; select processor type
radix   hex          ; define radix as hexadecimal
; #include "p16f84.inc"
;-----
;      cpu equates (memory map)
portB   equ    0x06   ;port B address
trisB   equ 0x06      ;trisB address in register bank 1
trisa   equ 0x05      ;trisA address in register bank 1

status  equ 0x03      ;status register
RP0     equ    5      ;bit 5 of status register
RP1     equ    6      ;bit 6 of status register

count   equ    0x0c   ; counter 0 to 255
mCount  equ    0x0d   ; pause routine inner loop counter
nCount  equ    0x0e   ; pause routine outer loop counter
;-----
```

```
pause
    movlw 0xff        ; set w = 255 decimal
    movwf mCount      ; mCount = w: outer loop counter

loadN
    movlw 0xff        ; set w = 255 decimal
    movwf nCount      ; nCount = w: inner loop counter

decN
    decfsz nCount, f   ; nCount--
    goto   decN        ; if nCount != 0 then repeat nCount--

    decfsz mCount, f   ; else decrement Count
    goto   loadN       ; if mCount != 0 then
                        ; reload nCount to 255 and decrement
    return            ; else exit subroutine

print_count
    movwf portB        ;moves w value into 7 segment display
    call  pause        ;delay for ~500ms

    return
```

```

count_up
movlw 0x3f      ; change w value to display value of 0
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x06      ; change w value to display value of 1
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x5b      ; change w value to display value of 2
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x4f      ; change w value to display value of 3
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x66      ; change w value to display value of 4
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x6d      ; change w value to display value of 5
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x7d      ; change w value to display value of 6
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x07      ; change w value to display value of 7
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x7f      ; change w value to display value of 8
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x67      ; change w value to display value of 9
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x77      ; change w value to display value of A
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x7c      ; change w value to display value of b
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x39      ; change w value to display value of C
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x5e      ; change w value to display value of d
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x79      ; change w value to display value of E
call  print_count ;updates display and pauses for ~0.2 second

movlw 0x71      ; change w value to display value of F
call  print_count ;updates display and pauses for ~0.2 second

goto  count_up   ; repeat forever

```

This programme produced the desired outcome of the 7-segment display count from 0 up to F with 200ms delay inbetween successfully.

Micros Assignment Bailey Freeman

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

22/100

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

GRADING FORM: GJ_2019

BAILEY FREEMAN

110

ENGAGEMENT WITH LITERATURE

 There are no references supplied **0**

KNOWLEDGE AND UNDERSTANDING

 While some of the information is reasonable, it too limited, doesn't talk about the regulator, the sounder.
The lack of EWL also limits this mark **40**

COGNITIVE AND INTELLECTUAL

 Only one programming task attempted and that uses a simple, brute force approach **10**

PRACTICAL APPLICATION

 Only one task attempted, and the circuit was not fully created. No details on the LED calculations either **20**

TRANSFERABLE SKILLS

 The report is far too brief. The table of contents is limited - needs more segmentation. **40**