Embedded Project

NAME: YOUANA WAGEH FOUAD

BN: 79

TASK (1):

a) Describe all the pins of PIC16f877A. After that, your colleagues would have enough information once they need to interface the PIC16f877A with other hardware.

ANSWER:

The PIC16F877A is a powerful 8-bit microcontroller from the PIC (Peripheral Interface Controller) family, developed by Microchip Technology. It has a total of 40 pins, which are described below:

1. VSS: Ground pin, used to provide the reference potential for the microcontroller.

2. VDD: Power supply pin, used to provide the voltage supply to the microcontroller.

3. RA0/AN0: Analog input or digital I/O pin, depending on the configuration.

4. RA1/AN1: Analog input or digital I/O pin, depending on the configuration.

5. RA2/AN2: Analog input or digital I/O pin, depending on the configuration.

6. RA3/AN3: Analog input or digital I/O pin, depending on the configuration.

7. RA4/T0CKI: Timer0 clock input or digital I/O pin, depending on the configuration.

8. RA5/AN4: Analog input or digital I/O pin, depending on the configuration.

9. RE0/RD/AN5: Analog input or digital I/O pin, depending on the configuration.

10. RE1/WR/AN6: Analog input or digital I/O pin, depending on the configuration.

11. RE2/CS/AN7: Analog input or digital I/O pin, depending on the configuration.

12. VPP/MCLR: Programming voltage or Master Clear Reset input, used for programming the microcontroller or resetting it.

13. RB0/INT: External interrupt input or digital I/O pin, depending on the configuration.

14. RB1: Digital I/O pin.

15. RB2: Digital I/O pin.

16. RB3: Digital I/O pin.

17. RB4: Digital I/O pin.

18. RB5: Digital I/O pin.

19. RB6/PGC: Programming clock input or digital I/O pin, depending on the configuration.

20. RB7/PGD: Programming data input or digital I/O pin, depending on the configuration.

21. RC0/T1OSO/T1CKI: Timer1 oscillator output or digital I/O pin, depending on the configuration.

22. RC1/T1OSI/CCP2: Timer1 oscillator input or digital I/O pin, depending on the configuration.

23. RC2/CCP1: Capture/Compare/PWM (pulse-width modulation) module or digital I/O pin, depending on the configuration.

24. RC3/SCL: I2C bus clock input or digital I/O pin, depending on the configuration.

25. RC4/SDA: I2C bus data input or digital I/O pin, depending on the configuration.

26. RC5/TX/CK: USART (universal synchronous/asynchronous receiver/transmitter) transmit or digital I/O pin, depending on the configuration.

27. RC6/RX/DT: USART receive or digital I/O pin, depending on the configuration.

28. RC7/CCP3: Capture/Compare/PWM module or digital I/O pin, depending on the configuration.

29. OSC1/CLKI: External oscillator or clock input.

30. OSC2/CLKO: External oscillator or clock output.

31. VREF-/AN2: Negative reference voltage for the ADC (analog-to-digital converter) or analog input pin, depending on the configuration.

32. VREF+/AN3: Positive reference voltage for the ADC or analog input pin, depending on the configuration.

33. AN4: Analog input pin.

34. AN5: Analog input pin.

35. AN6: Analog input pin.

36. AN7: Analog input pin.

37. AVSS: Analog ground pin, used as the reference potential for the analog inputs.

38. AVDD: Analog power supply pin, used to provide the voltage supply to the analog inputs.

39. RB7: Digital I/O pin.

40. RB6: Digital I/O pin.

These pins can be configured as digital I/O pins or as analog inputs, depending on the specific requirements of the application. The microcontroller can be interfaced with other hardware using these pins, by configuring them appropriately and using the appropriate communication protocols.

b) Explain to your colleagues the functions of the main blocks in PIC16f877A : ALU, Status and Control, Program Counter, Flash Program Memory, Instruction Register, Instruction Decoder.

ANSWER:

1. ALU (Arithmetic Logic Unit): The ALU is the part of the microcontroller that performs arithmetic and logical operations on the data. It can perform operations such as addition, subtraction, multiplication,

AND, OR, XOR, and shift operations. The ALU operates on binary data and produces a result that is stored in a register.

2. Status and Control: The Status and Control block is responsible for managing the status of the microcontroller, such as the carry and overflow flags, and for controlling the execution of instructions.

It includes a number of registers that store the status of the microcontroller, such as the C (carry) flag, the Z (zero) flag, and the DC (decimal carry) flag.

3. Program Counter: The Program Counter is a register that keeps track of the address of the next instruction to be executed. It is automatically incremented after each instruction is executed, so that the microcontroller can fetch and execute the next instruction in sequence.

4. Flash Program Memory: The Flash Program Memory is where the program code is stored. It is non-volatile, which means that the program code remains stored even when the power is turned off. The microcontroller can read instructions from this memory and execute them.

5. Instruction Register: The Instruction Register is a register that holds the current instruction being executed. The microcontroller fetches the instruction from the Flash Program Memory and stores it in the Instruction Register, where it is decoded and executed.

6. Instruction Decoder: The Instruction Decoder is responsible for decoding the instruction stored in the Instruction Register and determining the appropriate action to take. It determines the operation to be performed, the operands involved, and the mode of addressing. It then sends signals to the appropriate blocks in the microcontroller to perform the required operation.

Taken together, these blocks form the core of the PIC16f877A microcontroller and enable it to execute instructions and perform a wide range of tasks. By understanding the functions of these blocks, colleagues can design and develop software and hardware interfaces that take full advantage of the capabilities of the microcontroller.

c)Examine the reasons why a led, which is connected to RA4 for flashing prepose not working probably.

ANSWER:

As pins4 in portA is open drain that mean when value equal 0 the n-MOS transistor is acts as short circuit and the output equal zero ,and when the value equal 1 the n-MOS acts as open circuit and the output will be unknown.

d) ATMega328P [2] is also an 8-bit but AVR microcontroller. Evaluate the characteristics of ATMega328P versus PIC16f877A, by comparing the memory size, the power consumption, pin count… of those two MCUs. Give 2 examples of embedded systems where ATMega328P is a better choice than PIC16f877A.

ANSWER:

Here is a comparison of the characteristics of the ATMega328P and the PIC16f877A microcontrollers:

1. Memory size: The ATMega328P has 32KB of Flash memory, 2KB of SRAM, and 1KB of EEPROM, while the PIC16f877A has 14KB of Flash memory, 368 bytes of SRAM, and no EEPROM. This means that the ATMega328P has more memory for storing program code and data

2. Power consumption: The ATMega328P has a lower power consumption than the PIC16f877A, with a typical current consumption of 1.5mA at 1MHz and 5V, compared to 15mA for the PIC16f877A at the same frequency and voltage. This makes the ATMega328P a better choice for battery-powered applications or other low-power systems.

3. Pin count: The ATMega328P has 28 pins, while the PIC16f877A has 40 pins. This means that the PIC16f877A has more I/O pins available for interfacing with other hardware.

Based on these characteristics, the ATMega328P may be a better choice than the PIC16f877A for certain embedded systems. **Here are two examples:**

1. Battery-powered systems: The lower power consumption of the ATMega328P makes it a better choice for battery-powered systems, such as portable devices or remote sensors. The larger amount of memory available on the ATMega328P also makes it easier to store data in these systems.

2. Small embedded systems: The smaller pin count of the ATMega328P makes it a better choice for small embedded systems where space is at a premium. Examples of such systems include wearable devices, small robots, or home automation systems where space is limited and only a few I/O pins are needed.

In summary, while both the ATMega328P and the PIC16f877A are 8-bit microcontrollers, they have different characteristics that make them better suited for different types of embedded systems. The ATMega328P has more memory, lower power consumption, and a smaller pin count, making it a better choice for battery-powered and small embedded systems.

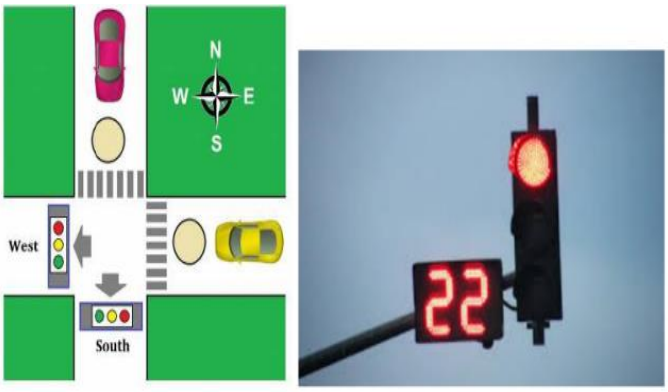
TASK (2):

we will build a traffic light controller, to get familiar with PIC environment:

we will make two cases (Automatic & Manual)

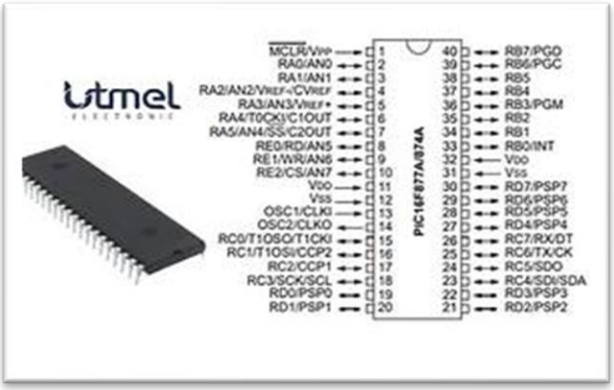
The timing in Automatic mode: West street (15s Red, 3s Yellow, 20s Green), South street (23s Red, 3s Yellow, 12s Green).

AND in Manual mode we will apply 3s for yellow in two streets.



FIRST we will make an overview on PIC16F877A:

OVERVIEW ON PIC16F877A:



The PIC16F877A is a popular 8-bit microcontroller manufactured by Microchip Technology. It is a member of the PIC16F family of microcontrollers and is widely used in various applications such as industrial automation, home automation, robotics, and automotive systems.

The PIC16F877A has a Harvard architecture with a 14-bit instruction word and a 8-bit data path. It has 8K of flash memory for program storage, 368 bytes of RAM for data storage, and 256 bytes of EEPROM for non-volatile data storage. It also has a wide range of peripheral features, including timers, USARTs, SPI, I2C, ADC, and PWM.

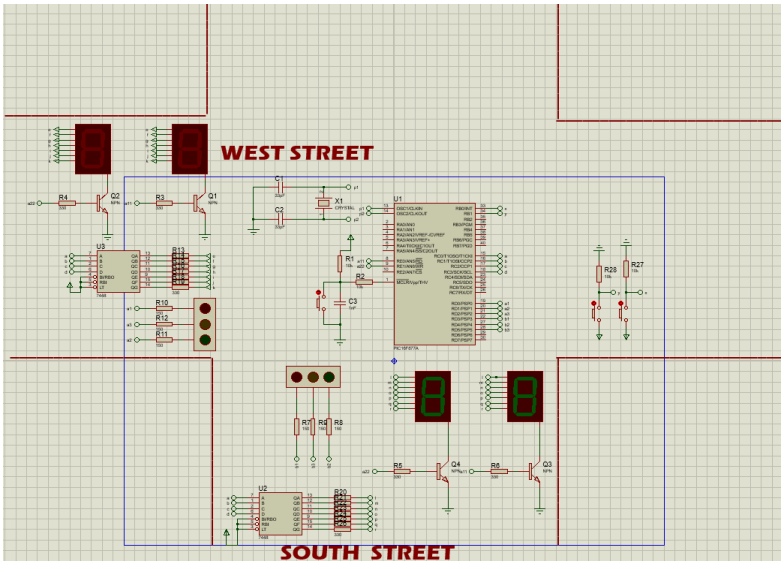
The microcontroller operates at a maximum frequency of 20 MHz and has a wide operating voltage range of 2.0V to 5.5V. It also has low power consumption features, making it suitable for battery-powered applications.

The PIC16F877A is programmed using the MPLAB Integrated Development Environment (IDE) and can be programmed in C, Assembly, or other high-level languages using a variety of compilers. It is also supported by a wide range of development tools and accessories, including development boards, programmers, and debuggers.

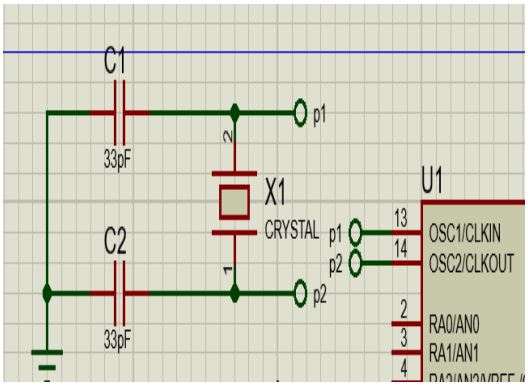
Overall, the PIC16F877A is a versatile and widely used microcontroller with a wide range of features and applications. Its popularity can be attributed to its ease of use, low cost, and availability of a wide range of development tools and resources.

The Circuit Of Traffic Light:

We wil use Proteus for simulation the circuit.



FIRST PART:



The PIC16F877A is a microcontroller from Microchip Technology that is widely used in embedded system designs. One of the features of this microcontroller is its ability to use an external crystal oscillator to generate a precise clock signal for timing and synchronization of the microcontroller's operations. The PIC16F877A is a microcontroller from Microchip Technology that is widely used in embedded system designs. One of the features of this microcontroller is its ability to use an external crystal oscillator to generate a precise clock signal for timing and synchronization of its operations.

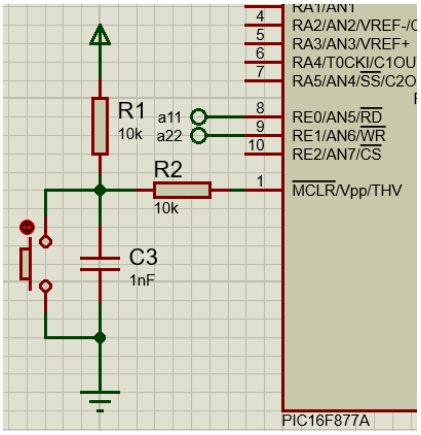
The crystal oscillator is used to provide a stable clock signal to the microcontroller, which is required for the proper functioning of its internal peripherals such as timers, UARTs, SPI, ADC, etc. The microcontroller uses this clock signal to synchronize the execution of instructions and to perform real-time operations.

The PIC16F877A can be configured to work with a range of crystal frequencies, typically from a few kilohertz up to 20 MHz or higher, depending on the specific requirements of the application.

In summary, the function of a crystal in PIC16F877A is to provide a stable and accurate clock signal for timing and synchronization of the microcontroller's operations.

SECOND PART:

The MCLR (Master Clear) button in PIC16F877A microcontroller is used to reset the device to its initial state. The MCLR button is connected to the MCLR pin of the microcontroller, which is an input pin. When the MCLR button is pressed, it pulls the MCLR pin low, which causes the microcontroller to reset.



THE SOUTH STREET:

The timing in Automatic mode: South street (23s Red, 3s Yellow, 12s Green).

The pins of 7segments(a,b,c,d,e,f,g) is connected to decoder

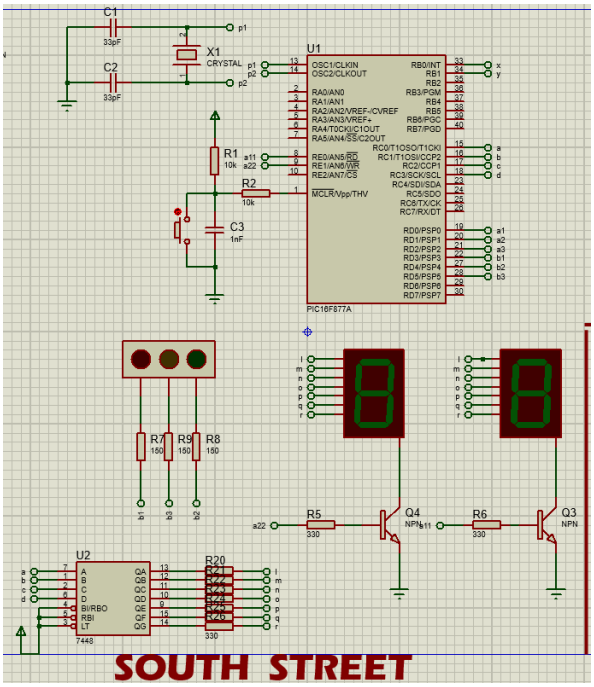
The function of decoder:

A decoder is a combinational logic circuit that is used to convert binary or BCD (binary coded decimal) input signals into a specific output pattern. In a 7-segment display circuit, a decoder is used to convert a binary or BCD input signal into the appropriate segment control signals that are required to display a specific digit or character on the 7-segment display.

In Common Cathode 7segment will connect the pins of 7segment with decoder 7448,

And In Common Anode 7segment will connect to decoder 7447.

IN THIS CIRCUIT will connect to 7447 decoder with common cathode 7 segment.



The Decoder connect to PORT.C in Microcontroller.

We will connect a transistor with 7segment,

The function of transistor:

In a 7-segment display circuit, a transistor is often used to drive the segments of the display. The transistor acts as a switch, controlling the flow of current to each segment of the display.

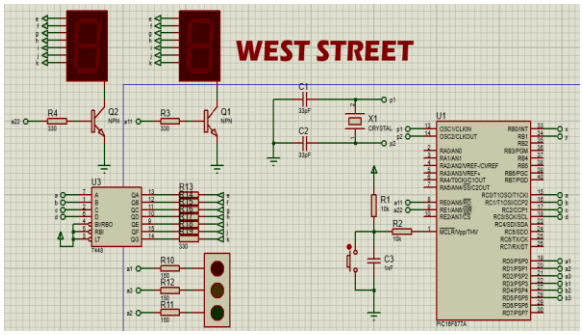
In Common Cathode 7segment will connect NPN transisor, And In Common Anode 7segment will connect to PNP transistor.

Th pins of transistor in PORT.E in Microcontroller.

The Leds of traffic light connect to PORTD(b0,b1,b2).

THE WEST STREET:

The timing in Automatic mode: West street (15s Red, 3s Yellow, 20s Green).



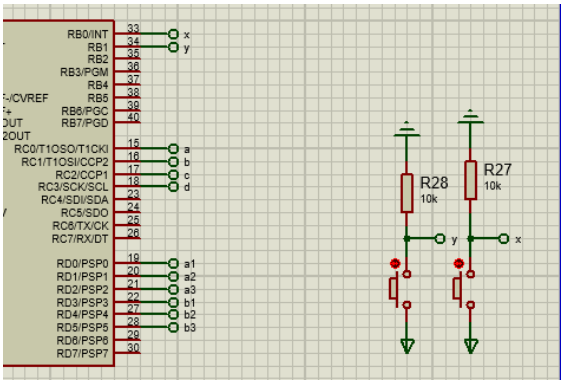
The pins of 7segments(a,b,c,d,e,f,g) is connected to decoder, it's connected to the same pins of the south street in PORT.C in Microcontroller.

The pins of transistor in PORT.E in Microcontroller as same as the pins of south street's pins.

The Leds of traffic light connect to PORTD(b3,b4,b5).

THE SWITCHES:

we will use 2 switches: one for Manual/Automatic and one for switching between 2 streets in Manual mode.



The one who switches between Manual/Automatic in portB.b1

And the one who switches between 2 streets in Manual mode in portB.b0 to make **the interrupt** for coding.

in the PIC16F877A microcontroller, Port B is an 8-bit bidirectional I/O port with internal pull-up resistors. The RB0 pin of Port B can be configured as an external interrupt input.

FOR CODING we will use **MikroC PRO** and will put the coding file in microcontroller in proteus.

In the code will identify the pins that will be used for the circuit and the program to simulate the required purposes.

HOW THIS FUNCTION WORKS:

This function counter used for handle all 7 segments in same port in microcontroller.

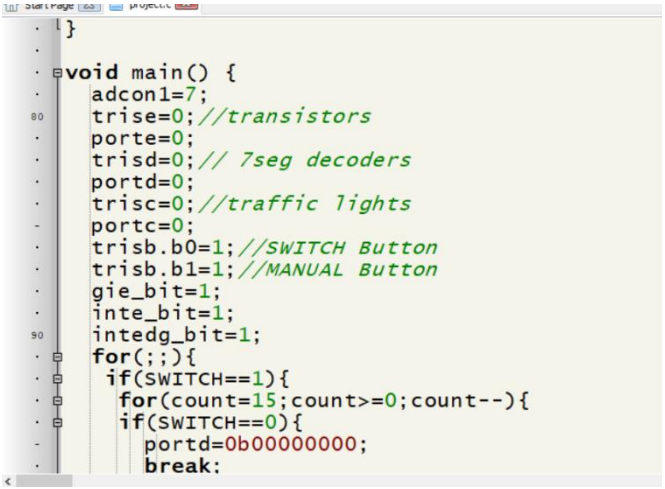
For left and right variables , their function for separate the number that will occur on 7segment to MSD and LSD **.**

```
#define LSD porte.b0
#define MSD porte.b1
#define MANUAL portb.b0
#define SWITCH portb.b1
signed char count;
  unsigned char i;
void counter(signed char count){
    char left=count/10;
    char right=count%10;
    MSD=0; //npn transistor2 off
    LSD=1;// npn transistor1 on
    portc=right;
    delay_ms(10);
    LSD=0; //npn transistor1 off
    MSD=1;  //npn transistor2 on
    portc=left;
    delay_ms(10);
}
int j=0;
void interrupt(){
  if(intf_bit==1){
```

FISRT , we will identify the pins of each port that will be used for simulation.

Transistors ,7 segment decoders and traffic lights will be identified by 0 as there will be outputs

AND for buttons will be identified by 1 as there will be inputs.



```
}

void main() {
  adcon1=7;
  trise=0;//transistors
  porte=0;
  trisd=0;// 7seg decoders
  portd=0;
  trisc=0;//traffic lights
  portc=0;
  trisb.b0=1;//SWITCH Button
  trisb.b1=1;//MANUAL Button
  gie_bit=1;
  inte_bit=1;
  intedg_bit=1;
  for(;;){
   if(SWITCH==1){
    for(count=15;count>=0;count--){
    if(SWITCH==0){
      portd=0b00000000;
      break;
```

HOW THA AUTOMATIC MODE WORKS:

The automatic mode works when SWITCH==1 and make an infinity for loop for work continously

AND make 2 for loops for 15 seconds and 23 seconds to simulate automatically and operate the leds (red for street and green for another street) and in last 3 sec will operate the yellow led for one street of them and another will be red , and after 3sec change to green and another one red.

FOR (SWITCH==0) If I will change to manual mode

```
for(;;){
 if(SWITCH==1){
  for(count=15;count>=0;count--){
  if(SWITCH==0){
    portd=0b00000000;
    break;
   }
    if(count<=3){
      portd=0b00100001;//W.red & S.yellow
    }else{
      portd=0b00010001;//w.red & S.green
    }
    for(i=0;i<50;i++){
      counter(count);
    }

 }
  for(count=23;count>=0;count--){
  if(SWITCH==0){
   portd=0b00000000;
    break;
```

HOW THE MANUAL MODE WORKS:

```
int j=0;
void interrupt(){
  if(intf_bit==1){
    intf_bit=0;
    j++;
  }
  if(j==1){
      portd=0b00010001;
    }
  if(j==2){
       for(count=3;count>=0;count--){
       if(SWITCH==1) break;
        portd=0b00100001;
        for(i=0;i<50;i++){
         char left=count/10;
         char right=count%10;
         MSD=0;
         LSD=1;
         portc=right;
         delay_ms(10);
         LSD=0;
         portc=left;
         delay_ms(10);
  }
  }
  j=3;
}
if(j==3){
   portd=0b00001010;

}if(j==4){
   for(count=3;count>=0;count--){
     if(SWITCH==1)break;
     portd=0b00001100;

     for(i=0;i<50;i++){
        char left=count/10;
        char right=count%10;
        MSD=0;
        LSD=1;
        portc=right;
        delay_ms(10);
        LSD=0;
        MSD=1;
        portc=left;
        delay_ms(10);
  }
  }
  j=1;
  portd=0b00010001;
}if(j==4){
   for(count=3;count>=0;count--){
     if(SWITCH==1)break;
     portd=0b00001100;

     for(i=0;i<50;i++){
        char left=count/10;
        char right=count%10;
        MSD=0;
        LSD=1;
        portc=right;
        delay_ms(10);
        LSD=0;
        MSD=1;
        portc=left;
        delay_ms(10);
  }
  }
  j=1;
  portd=0b00010001;
```

We will use an interrupt for the code of manual mode.

And use a manual switch for each state will operate the required leds, and will use the code of function counter for operating the 7segment in 3 seconds of yellow led.

FLOWCHART: