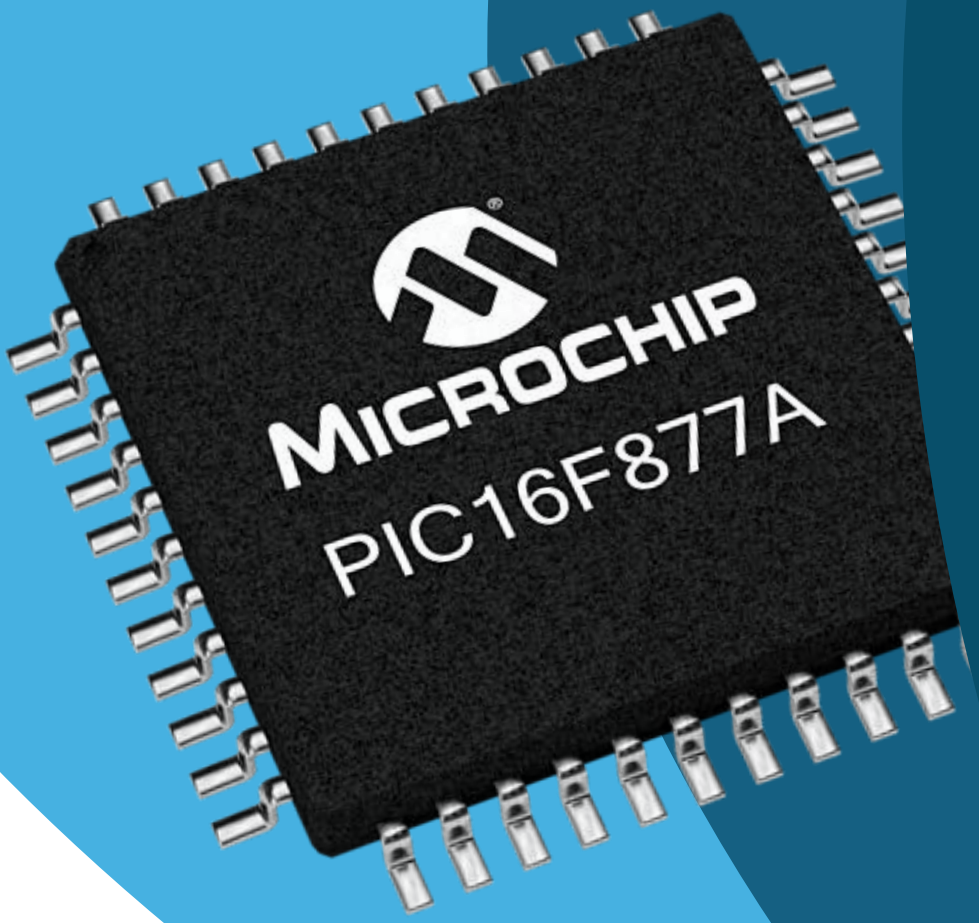


Project: Traffic light controller

جامعة بنها
BENHA UNIVERSITY
كلية الهندسة بشبرا
FACULTY OF ENGINEERING AT SHOUBRA

Work on this research

Name: zeyad bilal gamal elden

Department: 2nd Year Communication and computer engineering

Sec: 2 **B.N**: 26

The Headlines

Task_1:

- **All the pins of PIC16f877A describe.**
- **The functions of the main blocks in PIC16f877A.**
- **A led, which is connected to RA4.**
- **The characteristics of ATMega328P versus PIC16f877A.**
- **2 examples of embedded systems where ATMega328P is a better choice than PIC16f877A.**

Task_2:

- **The circuit for the whole system using electronic circuits simulator program.**
- **The code of the project.**
- **The flowchart for programming.**
- **The component list**

All the pins of PIC16f877A describe:

The PIC16F877A is a popular microcontroller from Microchip Technology. It features 40 pins in a dual in-line package (DIP). Here's a detailed description of each pin:

Power Supply Pins

1. **VDD (Pins 11 and 32):** Positive supply voltage.
2. **VSS (Pins 12 and 31):** Ground reference.

Oscillator Pins

3. **OSC1/CLKIN (Pin 13):** Oscillator crystal or external clock input.
4. **OSC2/CLKOUT (Pin 14):** Oscillator crystal output or clock output.

Reset Pin

5. **MCLR/VPP (Pin 1):** Master Clear (Reset) input or programming voltage input. It is an active-low reset pin.

Port A (RA0-RA5)

6. **RA0/AN0 (Pin 2):** Analog input 0 or digital I/O.
7. **RA1/AN1 (Pin 3):** Analog input 1 or digital I/O.
8. **RA2/AN2/VREF- (Pin 4):** Analog input 2 or voltage reference input (low) or digital I/O.
9. **RA3/AN3/VREF+ (Pin 5):** Analog input 3 or voltage reference input (high) or digital I/O.
10. **RA4/T0CKI (Pin 6):** Digital I/O or Timer0 clock input.
11. **RA5/AN4/SS (Pin 7):** Analog input 4, digital I/O, or SPI Slave Select.

Port B (RB0-RB7)

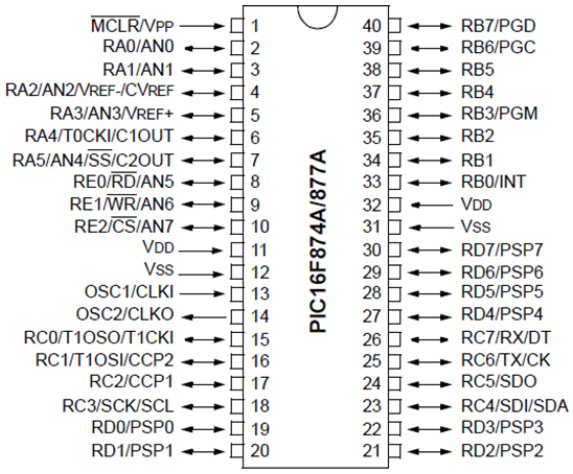
12. **RB0/INT (Pin 33):** External interrupt input or digital I/O.
13. **RB1 (Pin 34):** Digital I/O.
14. **RB2 (Pin 35):** Digital I/O.
15. **RB3/PGM (Pin 36):** Digital I/O or programming mode select.
16. **RB4 (Pin 37):** Digital I/O.
17. **RB5 (Pin 38):** Digital I/O.
18. **RB6/PGC (Pin 39):** Digital I/O or in-circuit debugger and programming clock.
19. **RB7/PGD (Pin 40):** Digital I/O or in-circuit debugger and programming data.

Port C (RC0-RC7)

20. **RC0/T1OSO/T1CKI (Pin 15):** Digital I/O, Timer1 oscillator output, or Timer1 clock input.
21. **RC1/T1OSI (Pin 16):** Digital I/O or Timer1 oscillator input.
22. **RC2/CCP1 (Pin 17):** Digital I/O or Capture/Compare/PWM module 1.
23. **RC3/SCK/SCL (Pin 18):** Digital I/O, SPI clock, or I2C clock.
24. **RC4/SDI/SDA (Pin 23):** Digital I/O, SPI data input, or I2C data.
25. **RC5/SDO (Pin 24):** Digital I/O or SPI data output.
26. **RC6/TX/CK (Pin 25):** Digital I/O, USART transmit, or synchronous clock.
27. **RC7/RX/DT (Pin 26):** Digital I/O, USART receive, or data terminal.

Port D (RD0-RD7)

28. **RD0/PSP0 (Pin 19):** Digital I/O or Parallel Slave Port (PSP) data.
29. **RD1/PSP1 (Pin 20):** Digital I/O or PSP data.
30. **RD2/PSP2 (Pin 21):** Digital I/O or PSP data.
31. **RD3/PSP3 (Pin 22):** Digital I/O or PSP data.
32. **RD4/PSP4 (Pin 27):** Digital I/O or PSP data.
33. **RD5/PSP5 (Pin 28):** Digital I/O or PSP data.
34. **RD6/PSP6 (Pin 29):** Digital I/O or PSP data.
35. **RD7/PSP7 (Pin 30):** Digital I/O or PSP data.



Port E (RE0-RE2)

36. **RE0/RD/AN5 (Pin 8):** Digital I/O, Read control for PSP, or Analog input 5.
37. **RE1/WR/AN6 (Pin 9):** Digital I/O, Write control for PSP, or Analog input 6.
38. **RE2/CS/AN7 (Pin 10):** Digital I/O, Chip Select control for PSP, or Analog input 7.

Special Function Pins

39. **PGM (Pin 36):** Low-voltage programming mode entry.

Each of these pins can serve multiple functions, and their actual use is determined by the configuration of the microcontroller during programming. The versatility of the pins allows the PIC16F877A to be used in a wide variety of applications.

The functions of the main blocks in PIC16f877A:

The PIC16F877A microcontroller contains several key functional blocks that work together to execute instructions and manage operations. Here's an explanation of the main blocks and their functions:

1. Arithmetic Logic Unit (ALU)

- **Function:** The ALU is responsible for performing arithmetic and logical operations. These include addition, subtraction, bitwise AND, OR, XOR, and other mathematical calculations. The ALU processes data from the registers and provides the results back to the registers or memory.
- **Role:** It is essential for performing the calculations required by the microcontroller's programs and making logical decisions based on those calculations.

2. Status and Control Registers

- **Function:** The Status register (STATUS) contains flags that indicate the outcomes of ALU operations, such as Zero (Z), Carry (C), Digit Carry (DC), and Negative (N) flags. Control registers are used to configure the microcontroller's operation modes and control various functions.
- **Role:** The STATUS register helps in decision-making during program execution by indicating the state of the ALU operations. Control registers manage the operation of peripherals, I/O ports, and other internal functions.

3. Program Counter (PC)

- **Function:** The Program Counter holds the address of the next instruction to be executed from the program memory. It increments after each instruction fetch to point to the subsequent instruction.
- **Role:** The PC ensures the sequential execution of instructions stored in the program memory, enabling the microcontroller to follow the program flow.

4. Flash Program Memory

- **Function:** This is a non-volatile memory where the program code is stored. It retains the code even when the power is turned off. The PIC16F877A has 14-bit wide instructions stored in this memory.
- **Role:** Flash Program Memory holds the firmware (the software that controls the hardware) of the microcontroller, providing instructions for execution.

5. Instruction Register (IR)

- **Function:** The Instruction Register temporarily holds the current instruction fetched from the program memory. It is where the opcode of the current instruction is stored before being decoded.

- **Role:** The IR ensures that the instruction is ready and available for the Instruction Decoder to interpret and execute.

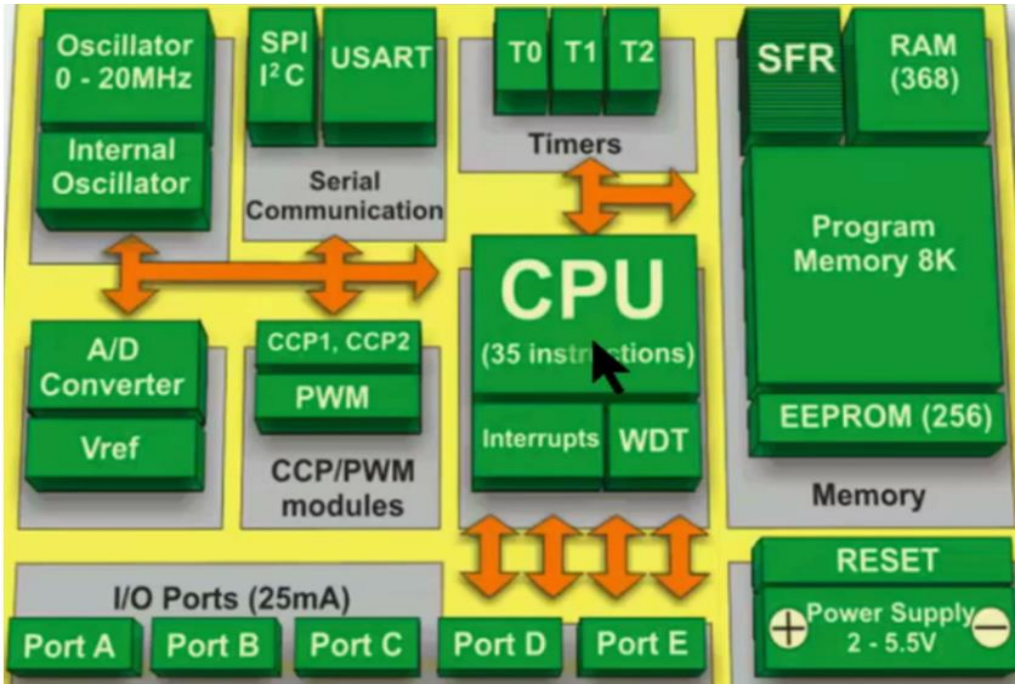
6. Instruction Decoder

- **Function:** The Instruction Decoder interprets the opcode from the Instruction Register and generates the necessary control signals to execute the instruction. It decodes the instruction into specific operations that the ALU and other parts of the microcontroller perform.
- **Role:** The decoder translates the binary instruction into meaningful control signals that orchestrate the microcontroller's operations, directing the ALU, registers, and other components to perform the specified tasks.

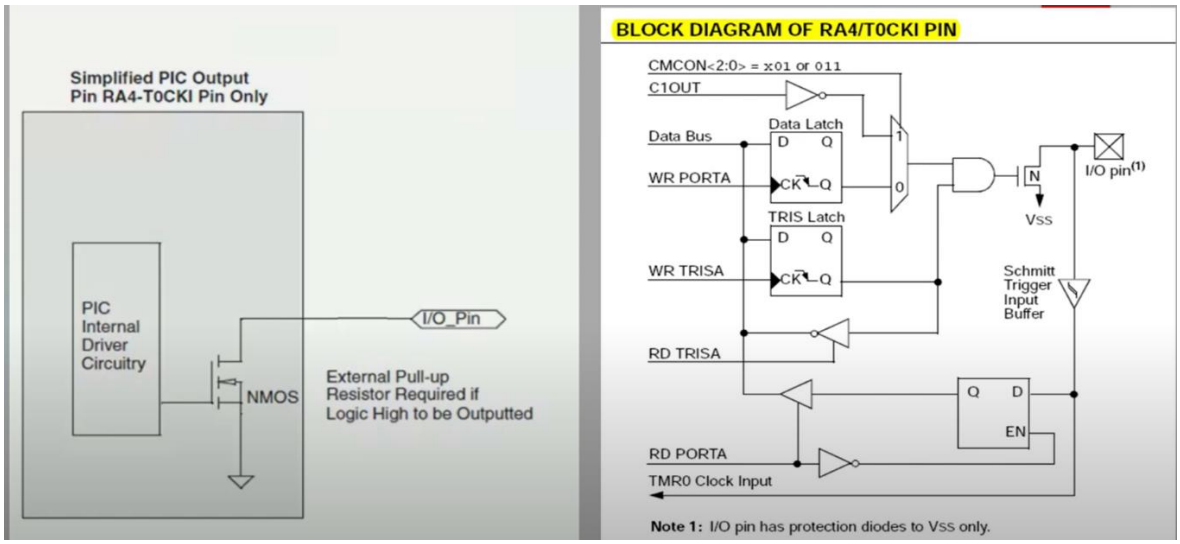
Summary of Interaction

1. **Fetch:** The Program Counter (PC) fetches the next instruction from the Flash Program Memory.
2. **Decode:** The fetched instruction is placed in the Instruction Register (IR), where the Instruction Decoder interprets it.
3. **Execute:** Based on the decoded instruction, the ALU performs the necessary arithmetic or logical operations. The Status register updates flags based on the ALU's output.
4. **Control:** Control registers manage the configuration and operation of various peripherals and I/O ports based on the decoded instructions.

These components work together seamlessly to allow the PIC16F877A to perform a wide range of functions, from simple calculations to complex control tasks in embedded systems.



A led, which is connected to RA4:

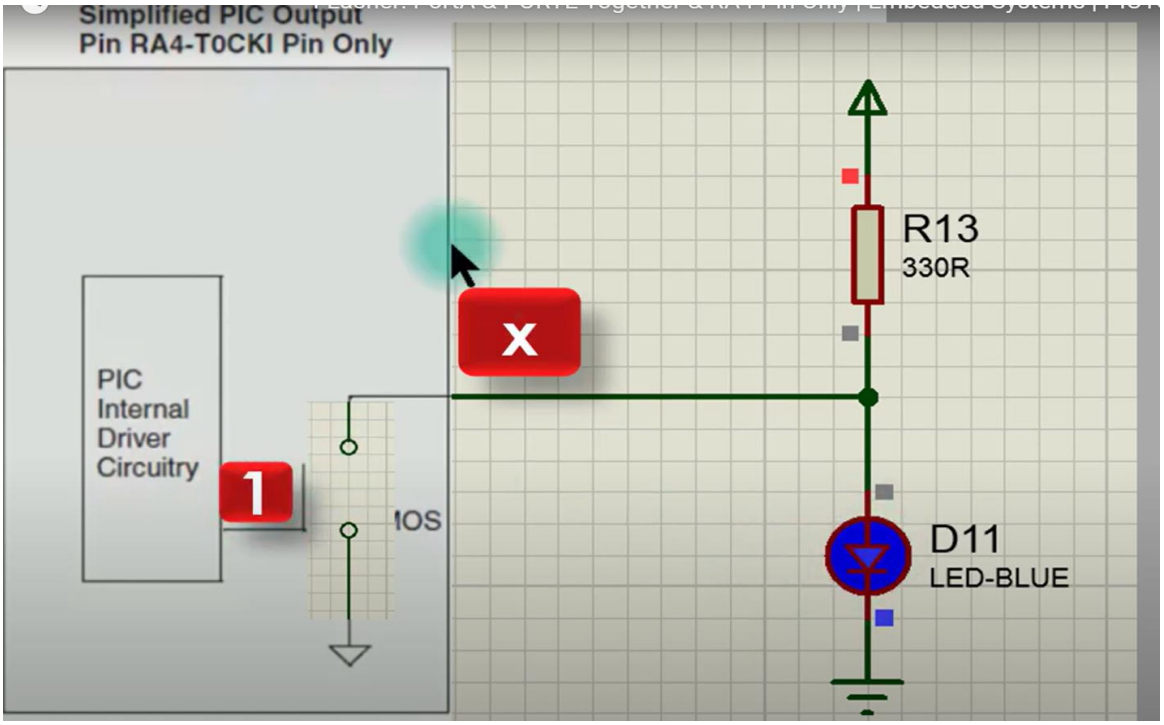


Open-Drain Configuration of RA4

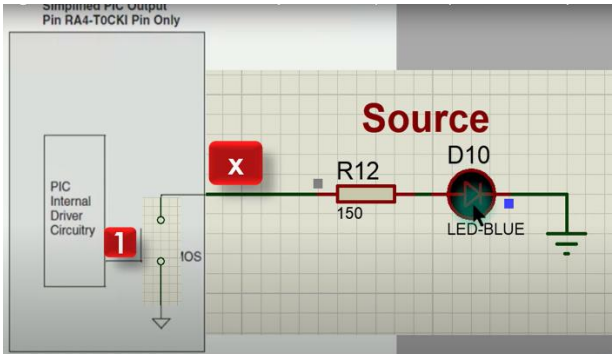
1. **Open-Drain Output:**
    - **Behavior:** An open-drain output can only sink current; it cannot source current. This means when RA4 is set low (logic 0), it can pull the connected circuit to ground. However, when it is set high (logic 1), it essentially becomes an open circuit and does not provide a high voltage level.
    - **Requirement:** To use RA4 to drive an LED or any other load, you need an external pull-up resistor to pull the line high when RA4 is not sinking current.
2. **Standard Digital Output:**
    - **Behavior:** Standard digital output pins can both source current (provide a high voltage level when set to logic 1) and sink current (pull to ground when set to logic 0).
    - **No Additional Components Needed:** These pins do not require an external pull-up resistor to drive a load.

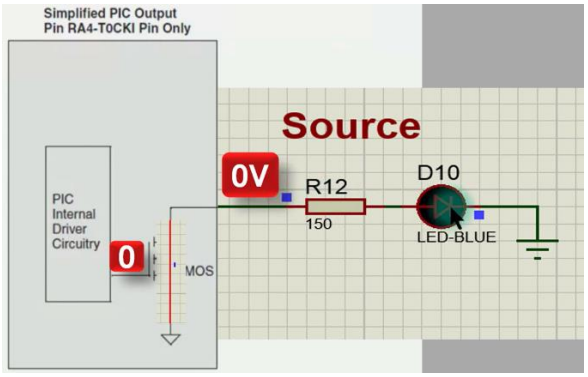
Implications for Driving an LED

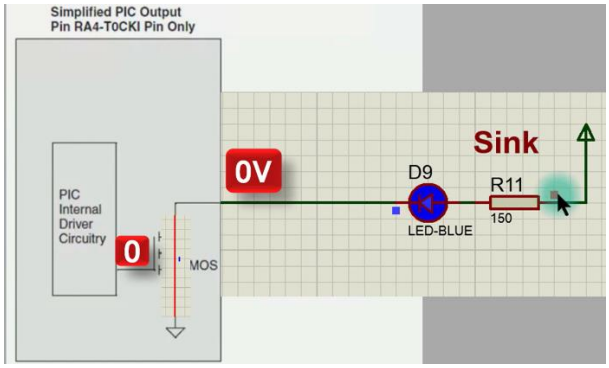
- **RA4 (Open-Drain):**
    - **External Pull-Up Resistor:** You need to connect an external pull-up resistor (e.g., 4.7kΩ to 10kΩ) between RA4 and VDD.
    - **Circuit Example:**

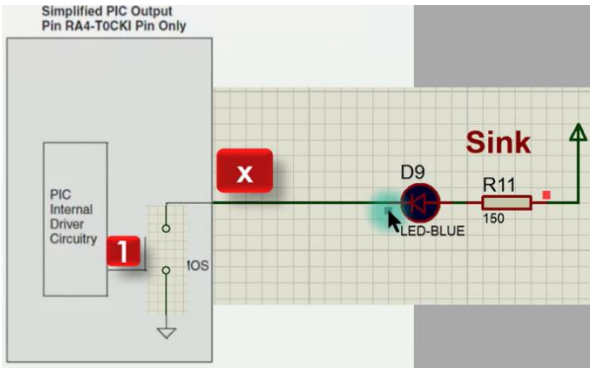


- **Operation:** When RA4 is set low, it pulls the LED cathode to ground, allowing current to flow from VDD through the pull-up resistor and LED, lighting the LED. When RA4 is set high, it is an open circuit, and the pull-up resistor pulls the anode of the LED to VDD, turning the LED off.









Troubleshooting RA4 Issues

If your LED is not working properly when connected to RA4, consider the following steps:

1. **Add a Pull-Up Resistor:**
    - Ensure you have an appropriate pull-up resistor connected between RA4 and VDD.
2. **Check Circuit Configuration:**
    - Verify that the LED and pull-up resistor are connected correctly as per the open-drain configuration.
3. **Verify Code Logic:**
    - Ensure your code correctly toggles RA4. Remember that setting RA4 low will turn on the LED and setting it high will turn off the LED (with the pull-up resistor in place).

The characteristics of ATMega328P versus PIC16f877A:

The ATMega328P and PIC16F877A are both 8-bit microcontrollers but from different families (AVR for ATMega328P and PIC for PIC16F877A). Here's a detailed comparison of their characteristics:

1. Memory Size

Flash Program Memory:

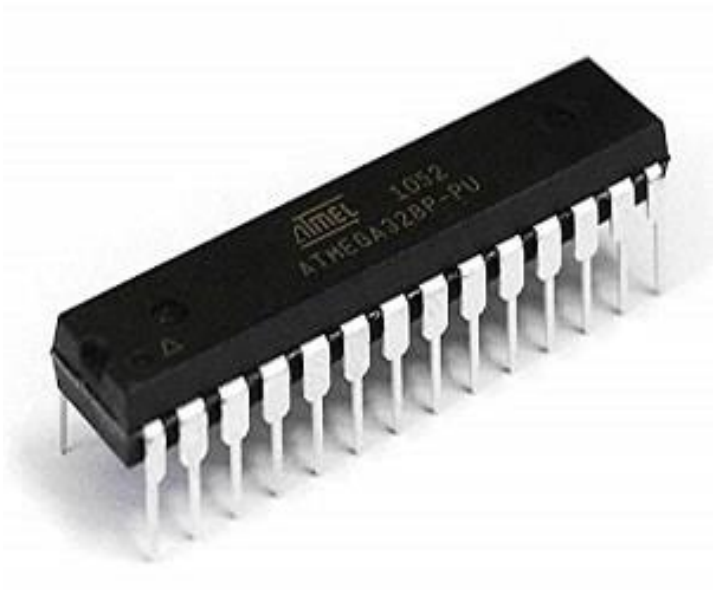
- **ATMega328P:** 32 KB
- **PIC16F877A:** 14 KB

SRAM:

- **ATMega328P:** 2 KB
- **PIC16F877A:** 368 bytes

EEPROM:

- **ATMega328P:** 1 KB
- **PIC16F877A:** 256 bytes



2. Power Consumption

Operating Voltage:

- **ATMega328P:** 1.8V to 5.5V
- **PIC16F877A:** 2.0V to 5.5V

Power Consumption (at 5V):

- **ATMega328P:**
  - Active mode: 0.2 mA at 1 MHz, 0.8 mA at 8 MHz
  - Power-down mode: 0.1 µA
- **PIC16F877A:**
  - Active mode: 2 mA at 4 MHz, 11 mA at 20 MHz
  - Sleep mode: 1 µA

3. Pin Count

- **ATMega328P:** 28 pins (DIP package), also available in 32-pin (TQFP, QFN/MLF)
- **PIC16F877A:** 40 pins (DIP package), also available in 44-pin (TQFP)

4. Clock Speed

Maximum Clock Frequency:

- **ATMega328P:** 20 MHz
- **PIC16F877A:** 20 MHz

5. I/O Ports

Digital I/O Pins:

- **ATMega328P:** 23
- **PIC16F877A:** 33

6. Analog Features

ADC Channels:

- **ATMega328P:** 6 channels (10-bit resolution)
- **PIC16F877A:** 8 channels (10-bit resolution)

7. Timers/Counters

Timers:

- **ATMega328P:**
  - Three timers: Two 8-bit (Timer0, Timer2) and one 16-bit (Timer1)

- **PIC16F877A:**
    - Three timers: One 8-bit (Timer0) and two 16-bit (Timer1, Timer2)

8. Communication Interfaces

UART:

- **ATMega328P:** 1
- **PIC16F877A:** 1

SPI:

- **ATMega328P:** 1
- **PIC16F877A:** 1

I2C:

- **ATMega328P:** 1 (referred to as TWI - Two Wire Interface)
- **PIC16F877A:** 1

9. Other Features

Watchdog Timer:

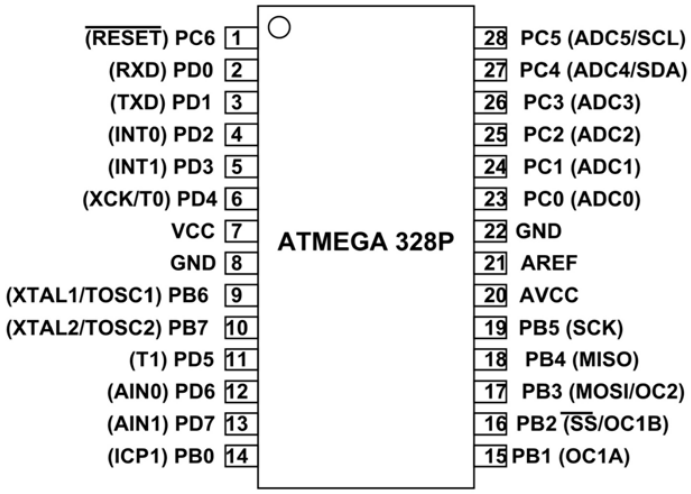
- **ATMega328P:** Yes
- **PIC16F877A:** Yes

PWM Channels:

- **ATMega328P:** 6
- **PIC16F877A:** 2 (via CCP modules)

Brown-out Detection:

- **ATMega328P:** Yes
- **PIC16F877A:** Yes



Summary

ATMega328P:

- Advantages: Larger program memory (Flash), more SRAM, larger EEPROM, lower power consumption in active and sleep modes, slightly fewer pins (easier for smaller projects), more PWM channels.
- Typical Uses: Widely used in Arduino boards, suitable for battery-powered applications due to lower power consumption, good for projects requiring more memory.

PIC16F877A:

- Advantages: More digital I/O pins, more ADC channels.
- Typical Uses: Suitable for projects needing more I/O pins and ADC channels, used in industrial applications due to robustness and wide operating voltage range.

In conclusion, the choice between ATMega328P and PIC16F877A depends on the specific requirements of the project, such as memory needs, power consumption, pin count, and peripheral availability.

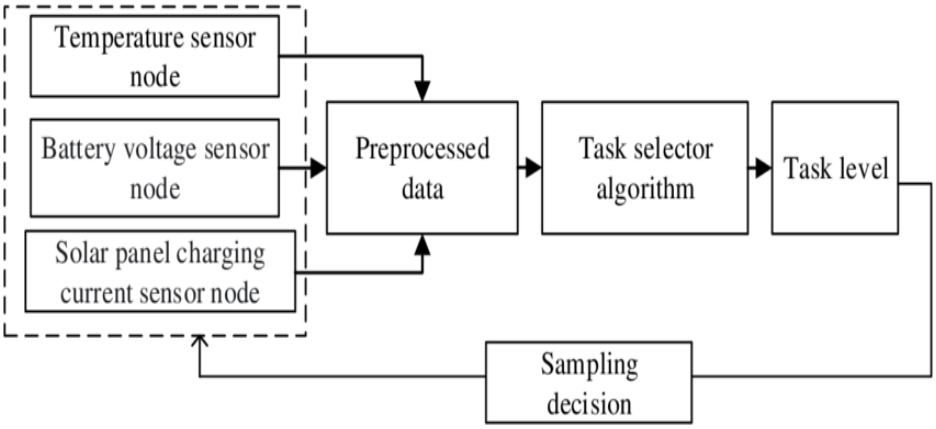
2 examples of embedded systems where ATMega328P is a better:

Example 1: Battery-Powered IoT Sensor Node.

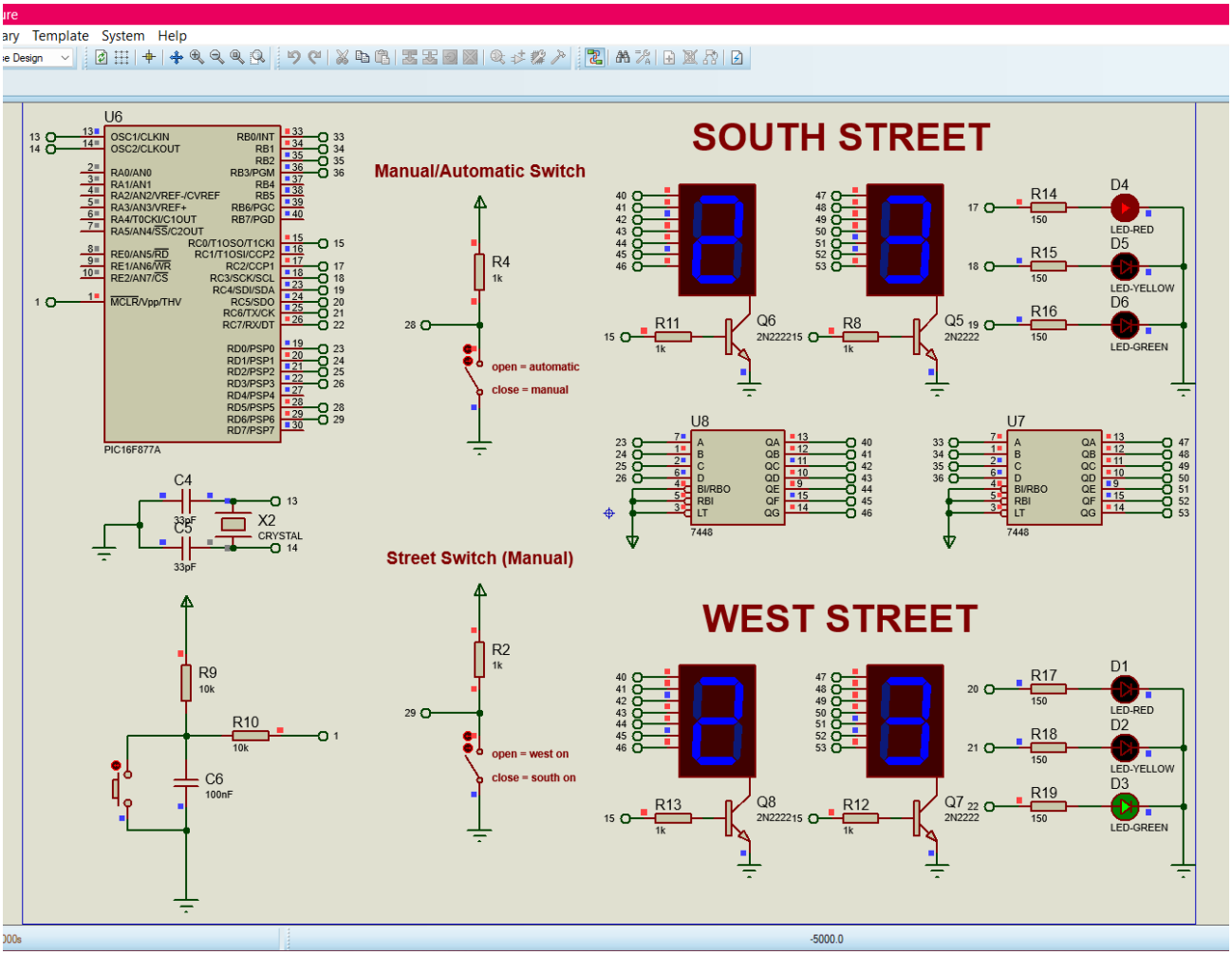
Example 2: Wearable Health Monitoring Device.

Summary

In both examples, the ATMega328P's lower power consumption, sufficient memory capacity, compact package options, and compatibility with the Arduino ecosystem make it a better choice for battery-powered IoT sensor nodes and wearable health monitoring devices. These advantages help in creating efficient, compact, and user-friendly embedded systems.



The circuit for the whole system using electronic circuits simulator program:



The code of the project.

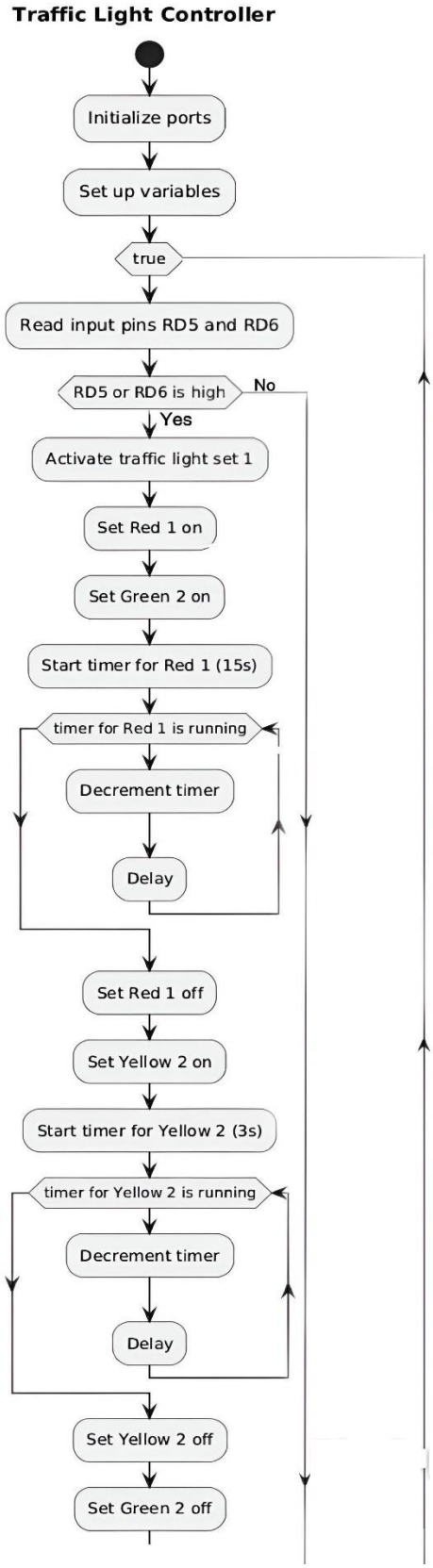
```
 int count=0;
 int count2=2;
void main() {
  trisb=0;
  trisd=0;
  trisc=0;
  trisc.RC0=0;
  trisd.RD5=1;
  trisd.RD6=1;
  portb=0;
  portd=0;
  portc=0;
  portc.RC0=1;
  while(1){
   if(portd.RD5==1 || portd.RD6==1){
         portc.RC2=1;  //RED 1 ON
         portc.RC7=1;  //green 2 on
         count2=2;
         portd=count2;
     for(count=3;count>=0;count--){
         portb=count;
```

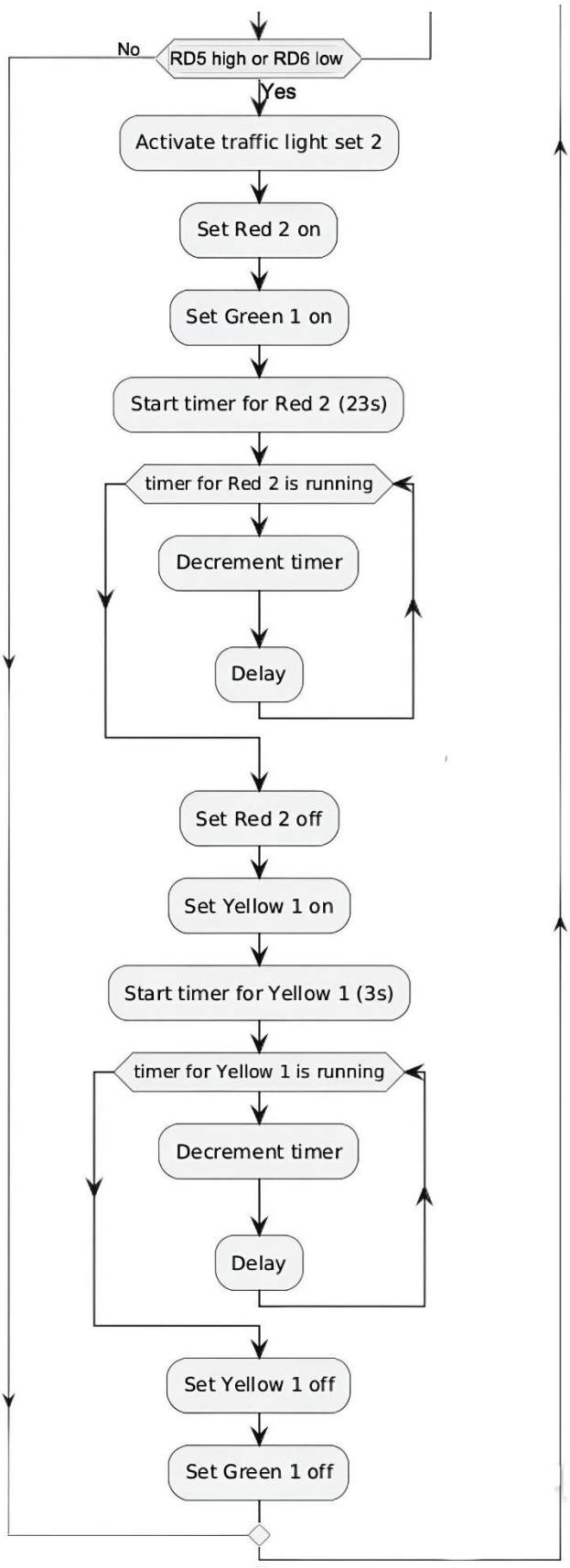
```
        delaY_ms(500);
        }
        count2--;
  while(count2>=0){
        portd=count2;
   for(count=9;count>=0;count--){
        portb=count;
        delaY_ms(500);
        if(count==4&&count2==0){
        portc.RC7=0;  //green 2 off after 20s
        portc.RC6=1;  //yellow 2 on

        }
   }
        count2--;
  }
        portc.RC6=0;  //yellow 2 off
        portc.RC2=0;  //RED 1 Off
 }
 if(portd.RD5==1 || portd.RD6==0){
        portc.RC5=1;  //RED 2 ON
        portc.RC4=1;  //green 1 on
        count2=1;
        portd=count2;
```

```
      for(count=5;count>=0;count--){
          portb=count;
          delaY_ms(500);
          }
          count2--;
     while(count2>=0){
          portd=count2;
      for(count=9;count>=0;count--){
          portb=count;
          delaY_ms(500);
          if(count==4&&count2==0){
          portc.RC4=0;  //green 1 off
          portc.RC3=1;  //yellow 1 on
          }
       }
          count2--;
      }
          portc.RC3=0;  //yellow 1 off
          portc.RC5=0;  //red 2 off
          count2=2;
     }
    }
}
```

The flowchart for programming.




No
RD5 high or RD6 low
Yes
Activate traffic light set 2
Set Red 2 on
Set Green 1 on
Start timer for Red 2 (23s)
timer for Red 2 is running
Decrement timer
Delay
Set Red 2 off
Set Yellow 1 on
Start timer for Yellow 1 (3s)
timer for Yellow 1 is running
Decrement timer
Delay
Set Yellow 1 off
Set Green 1 off

The component list :

Total Parts In Design 31

31 Miscellaneous

Quantity	References	Value
2	C4,C5	33pF
1	C6	100nF
2	D1,D4	LED-RED
2	D2,D5	LED-YELLOW
2	D3,D6	LED-GREEN
4	Q5,Q6,Q7,Q8	2N2222
6	R2,R4,R8,R11,R12,R13	1k
2	R9,R10	10k
6	R14,R15,R16,R17,R18,R19	150
1	U6	PIC16F877A
2	U7,U8	7448
1	X2	CRYSTAL

Thank you ☺