Benha University
Shoubra Faculty of Engineering
Communication and Computer
Engineering Department
2024/2025 Academic Year

كلية الهندســــة بشبرا
FACULTY OF ENGINEERING- SHOUBRA

Report on MICROCONTROLLERS: AN IN-DEPTH LOOK AT PIC16F877A AND ATMEGA328P

Second Year

Presented by

NOURA EMAD HAMEDY Ahmed

Summer 2025

Contents

PIC16F877A Microcontroller: A Technical Overview

Good morning, team! Today, we'll be diving into the PIC16F877A microcontroller from Microchip. This presentation aims to provide you with a comprehensive understanding of its architecture, pinout, and key functionalities, which will be essential as we move into the system specification phase of our project.

a) PIC16F877A Pin Description

The PIC16F877A is a 40-pin microcontroller, offering a versatile set of I/O ports and peripheral functionalities. Understanding each pin's role is crucial for effective hardware interfacing.

Power supply Pin:

VSS (Pins 19, 31): Ground reference for the microcontroller.

VDD (Pins 20, 32): Positive supply voltage (typically +5V)

Oscillator Pins:

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

  Oscillator circuits are electronic circuits that generate a periodic signal at a fixed frequency, serving as the **clock** for the microcontroller to determine instruction execution speed and synchronize its operations.

Reset Pin:

- **MCLR/VPP (Pin 1):** Master Clear (Reset) input. Can also be used as a programming voltage input (VPP) during Flash programming.

I/O Ports:

The PIC16F877A features five 8-bit I/O ports (Port A, Port B, Port C, Port D, Port E), each with multiple functions.

PORTA (RA0-RA5)

- **RA0/AN0 (Pin 2): General Purpose I/O, Analog Input 0.**
- **RA1/AN1 (Pin 3): General Purpose I/O, Analog Input 1.**

- RA2/AN2/VREF-/CVREF (Pin 4): General Purpose I/O, Analog Input 2, Analog Reference Voltage Negative, Comparator Voltage Reference.
- RA3/AN3/VREF+ (Pin 5): General Purpose I/O, Analog Input 3, Analog Reference Voltage Positive.
- RA4/T0CKI/C1OUT (Pin 6): General Purpose I/O, Timer0 Clock Input, Comparator 1 Output. Note: This pin has an open-drain output, meaning it can only sink current, not source it. A pull-up resistor is typically required for proper operation when used as an output.
- RA5/AN4/SS/C2OUT (Pin 7): General Purpose I/O, Analog Input 4, Slave Select for SPI, Comparator 2 Output.

PORTB (RB0-RB7):

- RB0/INT (Pin 33): General Purpose I/O, External Interrupt input.
- RB1 (Pin 34): General Purpose I/O.
- RB2 (Pin 35): General Purpose I/O.
- RB3/PGM (Pin 36): General Purpose I/O, Low-Voltage ICSP Programming Enable.
- RB4 (Pin 37): General Purpose I/O.
- RB5 (Pin 38): General Purpose I/O.
- RB6/PGC (Pin 39): General Purpose I/O, In-Circuit Serial Programming Clock.
- RB7/PGD (Pin 40): General Purpose I/O, In-Circuit Serial Programming Data.

PORTC (RC0-RC7):

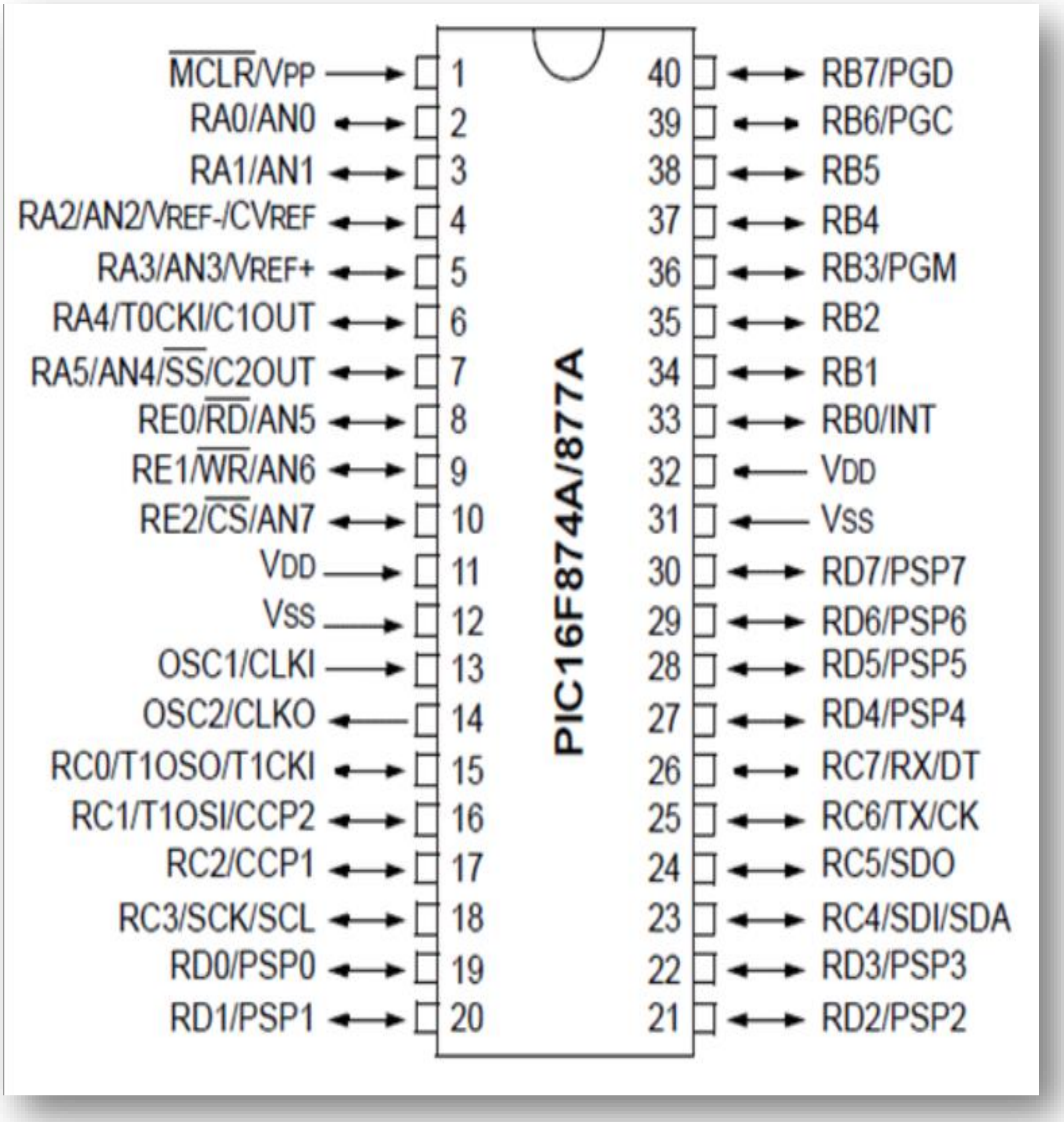
- RC0/T1OSO/T1CKI (Pin 15): General Purpose I/O, Timer1 Oscillator Output, Timer1 Clock Input.
- RC1/T1OSI/CCP2 (Pin 16): General Purpose I/O, Timer1 Oscillator Input, Capture/Compare/PWM 2.
- RC2/CCP1 (Pin 17): General Purpose I/O, Capture/Compare/PWM 1.
- RC3/SCK/SCL (Pin 18): General Purpose I/O, SPI Clock (Synchronous Serial Clock), I2C Serial Clock.
- RC4/SDI/SDA (Pin 23): General Purpose I/O, SPI Data In, I2C Serial Data.
- RC5/SDO (Pin 24): General Purpose I/O, SPI Data Out.
- RC6/TX/CK (Pin 25): General Purpose I/O, USART Transmit, Synchronous Clock.
- RC7/RX/DT (Pin 26): General Purpose I/O, USART Receive, Synchronous Data.

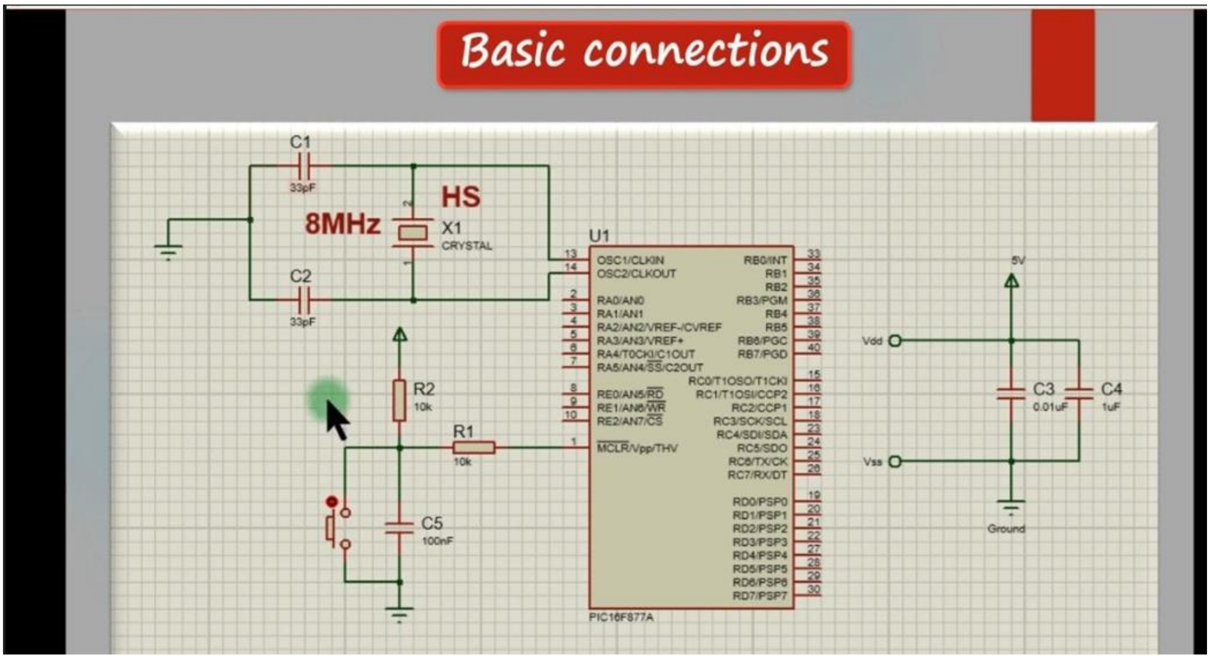
PORTD (RD0-RD7):

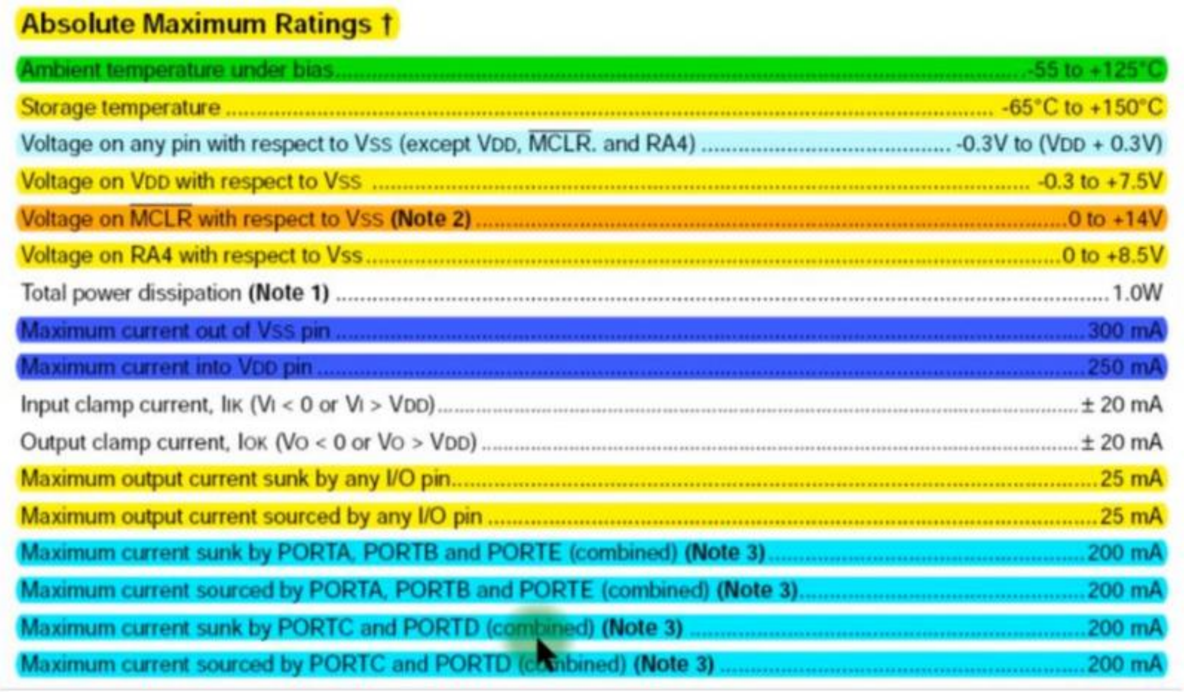
- RD0-RD7 (Pins 19-22, 27-30): General Purpose I/O. These pins are often used for parallel communication with peripherals like LCDs or external memory.

PORTE (RE0-RE2):

- **RE0/RD/AN5 (Pin 8): General Purpose I/O, Read control for Parallel Slave Port, Analog Input 5.**
- **RE1/WR/AN6 (Pin 9): General Purpose I/O, Write control for Parallel Slave Port, Analog Input 6.**
- **RE2/CS/AN7 (Pin 10): General Purpose I/O, Chip Select for Parallel Slave Port, Analog Input 7.**







Absolute Maximum Ratings †

Ambient temperature under bias	-55 to +125°C
Storage temperature	-65°C to +150°C
Voltage on any pin with respect to VSS (except VDD, $\overline{\text{MCLR}}$. and RA4)	-0.3V to (VDD + 0.3V)
Voltage on VDD with respect to VSS	-0.3 to +7.5V
Voltage on $\overline{\text{MCLR}}$ with respect to VSS **(Note 2)**	0 to +14V
Voltage on RA4 with respect to Vss	0 to +8.5V
Total power dissipation **(Note 1)**	1.0W
Maximum current out of VSS pin	300 mA
Maximum current into VDD pin	250 mA
Input clamp current, I_{IK} ($V_I < 0$ or $V_I > V_{DD}$)	± 20 mA
Output clamp current, I_{OK} ($V_O < 0$ or $V_O > V_{DD}$)	± 20 mA
Maximum output current sunk by any I/O pin	25 mA
Maximum output current sourced by any I/O pin	25 mA
Maximum current sunk by PORTA, PORTB and PORTE (combined) **(Note 3)**	200 mA
Maximum current sourced by PORTA, PORTB and PORTE (combined) **(Note 3)**	200 mA
Maximum current sunk by PORTC and PORTD (combined) **(Note 3)**	200 mA
Maximum current sourced by PORTC and PORTD (combined) **(Note 3)**	200 mA

b) Functions of Main Blocks in PIC16F877A

Let's explore the core functional blocks within the PIC16F877A architecture.

ALU (Arithmetic Logic Unit):

- The heart of the CPU, responsible for executing arithmetic operations (like addition, subtraction) and logical operations (like AND, OR, NOT). It performs calculations on data fetched from memory or registers.

Status and Control (Status Register):

- The Status Register holds the current status of the ALU and the CPU. It contains various flags (e.g., Carry, Zero, Digit Carry, Overflow) that indicate the result of the last arithmetic or logical operation. It also contains bits for controlling CPU operation, such as the Global Interrupt Enable bit.

Program Counter (PC):

- A special-purpose register that holds the memory address of the next instruction to be fetched and executed. After an instruction is fetched, the PC is incremented to point to the next instruction. In case of jumps, calls, or returns, the PC is loaded with a new address.

Flash Program Memory:

- This is the non-volatile memory where the microcontroller's program (firmware) is stored. "Flash" indicates that it can be electrically erased and reprogrammed multiple times. The PIC16F877A has 8K words of Flash Program Memory, meaning it can store 8192 instructions.

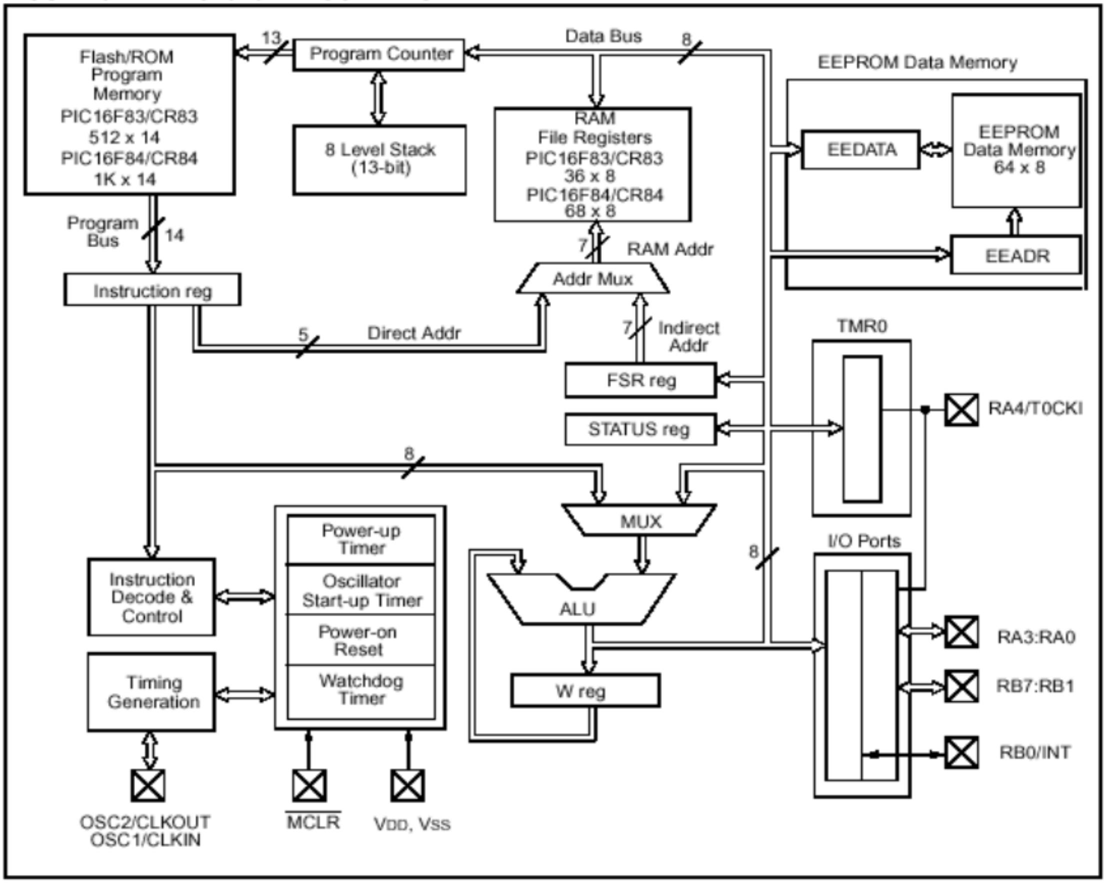
Instruction Register (IR):

- After an instruction is fetched from Program Memory, it is temporarily held in the Instruction Register. This register holds the binary code of the instruction currently being executed.

Instruction Decoder:

- This block takes the instruction from the Instruction Register and decodes it. It interprets the binary instruction code and generates the necessary control signals to various parts of the CPU (like the ALU, registers, and memory) to execute that specific instruction. It essentially translates the machine code into actions.

FIGURE 3-1: PIC16F8X BLOCK DIAGRAM



c) Reasons for RA4 LED Flashing Problem

If a LED connected to RA4 is not working properly for flashing purposes, here are the common reasons to examine:

1. ### Open-Drain Output Characteristic: RA4 is a unique pin as it has an **open-drain output**. This means it can only pull the line low (sink current) but cannot actively drive it high (source current).

   - **Problem:** If you're trying to turn the LED ON by setting RA4 high, it won't work directly.
   - **Solution:** You *must* use an external pull-up resistor connected from RA4 to VDD. When RA4 is set to output low, the LED will turn ON (if connected to VDD through the LED and resistor). When RA4 is set to output high, the pin effectively floats, and the pull-up resistor pulls the line high, turning the LED OFF.

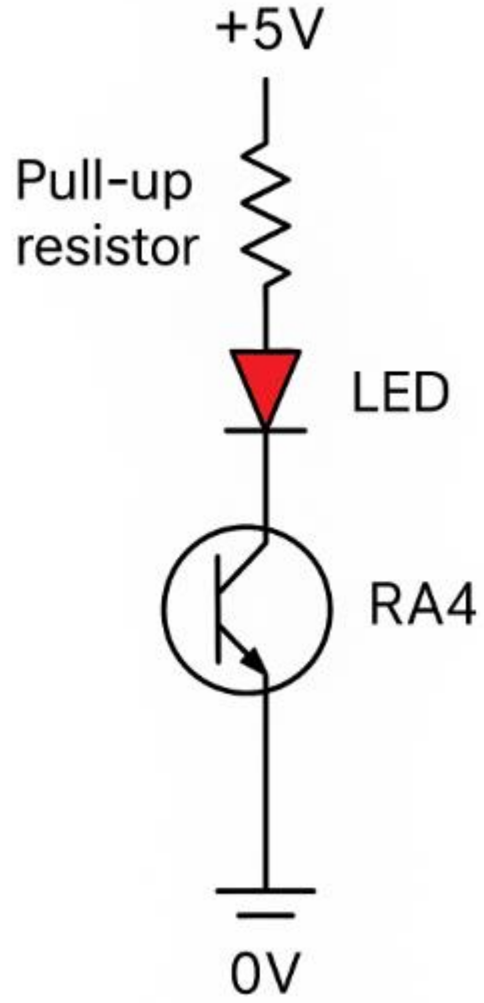
- **Correct Wiring:** LED anode to VDD, LED cathode to a current-limiting resistor, then to RA4. A pull-up resistor from RA4 to VDD is also needed.

2. TRISA Register Configuration: For RA4 to function as an output, the corresponding bit in the TRISA register (TRISA4) must be cleared to '0'.
   - **Problem:** If TRISA.F4 (or TRISA4) is set to '1' (input), the pin will not drive the LED.
   - **Solution:** Ensure TRISA.F4 = 0; in your initialization code.

3. Analog Function Configuration: RA4 also has an analog input function (AN4). If the Analog-to-Digital Converter (ADC) module is enabled and configured to use RA4, it will override its digital I/O function.
   - **Problem:** The ADC module might be inadvertently using RA4.
   - **Solution:** Ensure the ADCON1 register is configured correctly to disable analog functions on RA4 if you intend to use it as a digital I/O. Specifically, ensure ADCON1 bits are set to configure RA4 as a digital pin (e.g., ADCON1 = 0x06; for all digital I/O on Port A).

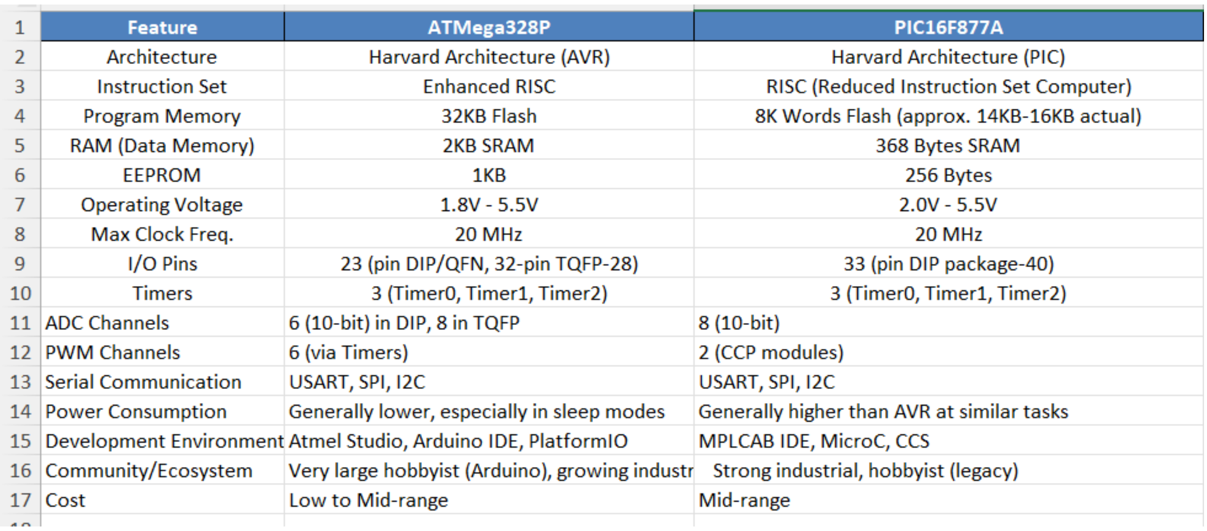
4. Incorrect LED Polarity/Resistor:
   - **Problem:** LED connected backward, or current-limiting resistor value is too high/low.
   - **Solution:** Double-check LED polarity (anode to positive, cathode to negative). Calculate the correct current-limiting resistor value based on LED forward voltage and desired current.

5. Code Logic Errors:
   - **Problem:** The flashing logic in the code might be incorrect (e.g., incorrect delays, not toggling the pin, or being stuck in an infinite loop elsewhere).
   - **Solution:** Review the code that controls RA4, ensuring it properly toggles the pin state and includes appropriate delays.


+5V
Pull-up resistor
LED
RA4
0V

d) ATMega328P vs. PIC16F877A Comparison and Use Cases

Both the ATMega328P (used in Arduino Uno) and PIC16F877A are popular 8-bit microcontrollers, but they have distinct characteristics that make them suitable for different applications.



1	Feature	ATMega328P	PIC16F877A
2	Architecture	Harvard Architecture (AVR)	Harvard Architecture (PIC)
3	Instruction Set	Enhanced RISC	RISC (Reduced Instruction Set Computer)
4	Program Memory	32KB Flash	8K Words Flash (approx. 14KB-16KB actual)
5	RAM (Data Memory)	2KB SRAM	368 Bytes SRAM
6	EEPROM	1KB	256 Bytes
7	Operating Voltage	1.8V - 5.5V	2.0V - 5.5V
8	Max Clock Freq.	20 MHz	20 MHz
9	I/O Pins	23 (pin DIP/QFN, 32-pin TQFP-28)	33 (pin DIP package-40)
10	Timers	3 (Timer0, Timer1, Timer2)	3 (Timer0, Timer1, Timer2)
11	ADC Channels	6 (10-bit) in DIP, 8 in TQFP	8 (10-bit)
12	PWM Channels	6 (via Timers)	2 (CCP modules)
13	Serial Communication	USART, SPI, I2C	USART, SPI, I2C
14	Power Consumption	Generally lower, especially in sleep modes	Generally higher than AVR at similar tasks
15	Development Environment	Atmel Studio, Arduino IDE, PlatformIO	MPLCAB IDE, MicroC, CCS
16	Community/Ecosystem	Very large hobbyist (Arduino), growing industr	Strong industrial, hobbyist (legacy)
17	Cost	Low to Mid-range	Mid-range

Evaluation:

- Memory Size: ATMega328P significantly outperforms PIC16F877A in both Program Memory (Flash) and Data Memory (SRAM). This allows for more complex programs and larger data buffers.

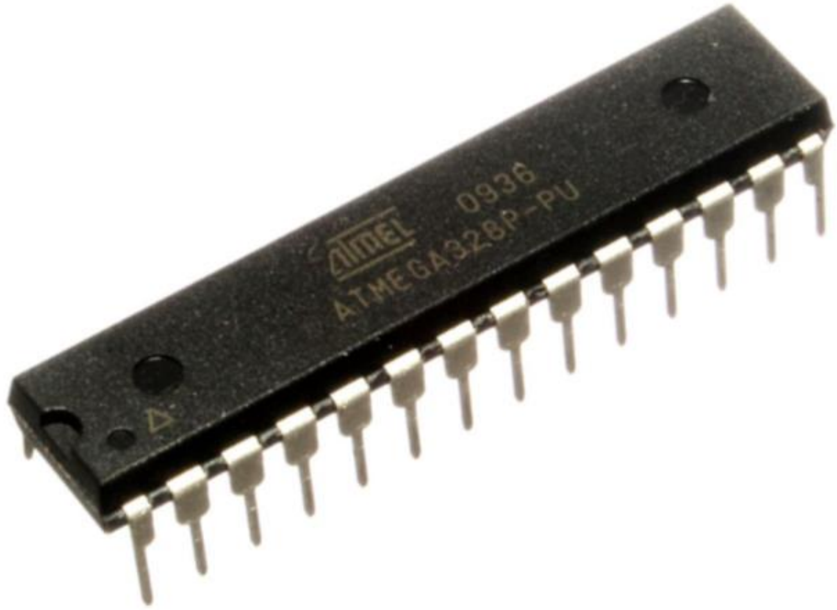
- Power Consumption: ATMega328P generally has better power efficiency, especially with its various sleep modes, making it more suitable for battery-powered applications.

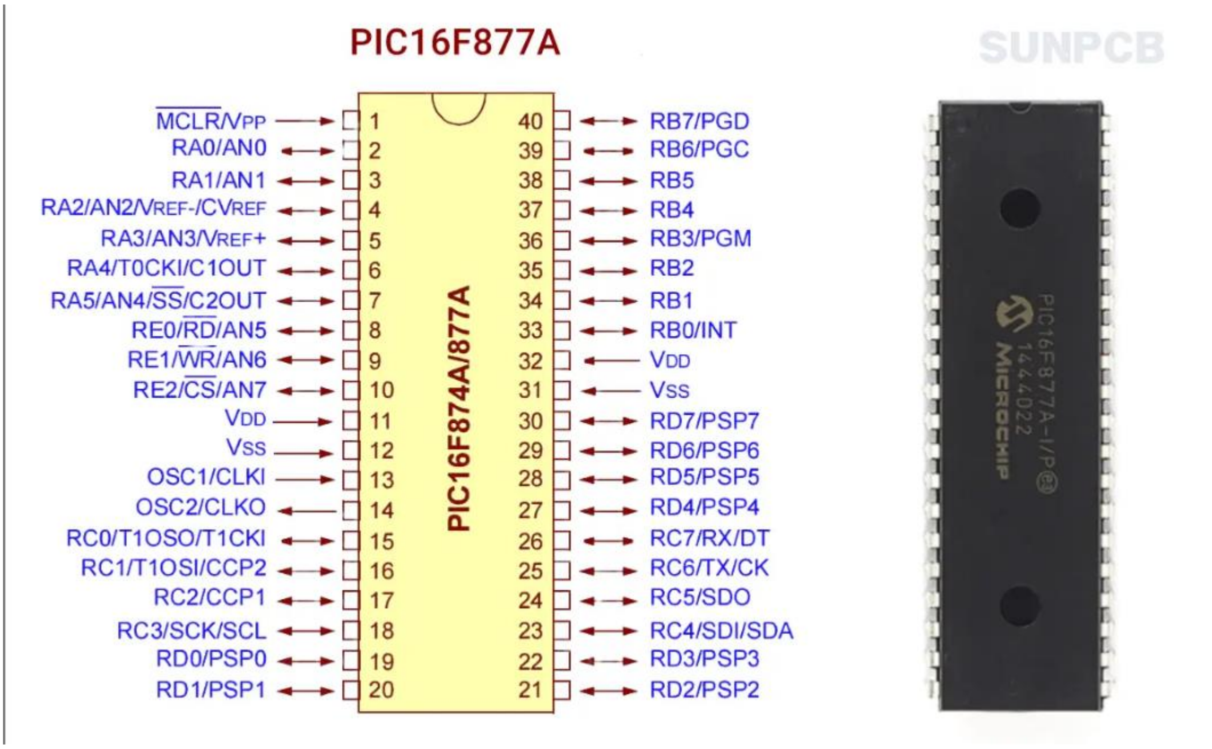
- Pin Count: PIC16F877A offers a higher pin count (33 I/O pins vs. 23 for ATMega328P in DIP package), providing more direct I/O options without external expanders.

- Development Ecosystem: The ATMega328P benefits immensely from the Arduino ecosystem, offering a vast library of ready-to-use code, shields, and a massive community, which significantly speeds up development for many applications. PIC has a strong industrial base but a less accessible hobbyist entry point compared to Arduino.

2_ Examples of Embedded Systems where ATMega328P is a better choice than PIC16F877A:

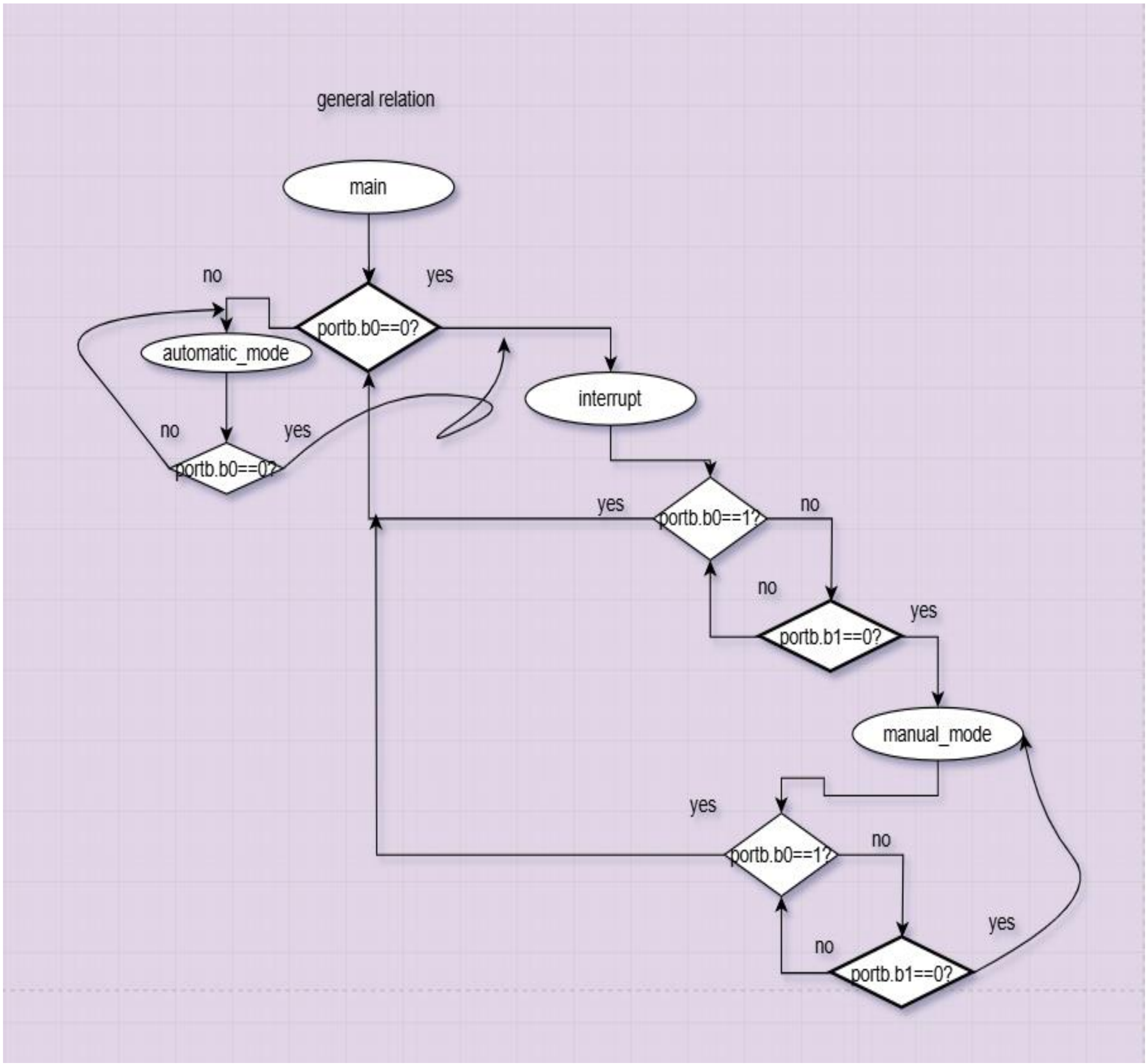
1. **Portable weather monitoring station – low power consumption extends battery life.**
2. **Arduino-based home automation system – larger program memory allows for more features and libraries.**

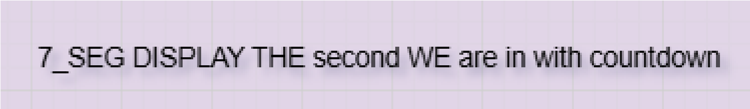

ATMEL
0936
ATMEGA328P-PU


PIC16F877A
SUNPCB
PIC16F874A/877A
MCLR/VPP 1
RA0/AN0 2
RA1/AN1 3
RA2/AN2/VREF-/CVREF 4
RA3/AN3/VREF+ 5
RA4/T0CKI/C1OUT 6
RA5/AN4/SS/C2OUT 7
RE0/RD/AN5 8
RE1/WR/AN6 9
RE2/CS/AN7 10
VDD 11
VSS 12
OSC1/CLKI 13
OSC2/CLKO 14
RC0/T1OSO/T1CKI 15
RC1/T1OSI/CCP2 16
RC2/CCP1 17
RC3/SCK/SCL 18
RD0/PSP0 19
RD1/PSP1 20
40 RB7/PGD
39 RB6/PGC
38 RB5
37 RB4
36 RB3/PGM
35 RB2
34 RB1
33 RB0/INT
32 VDD
31 VSS
30 RD7/PSP7
29 RD6/PSP6
28 RD5/PSP5
27 RD4/PSP4
26 RC7/RX/DT
25 RC6/TX/CK
24 RC5/SDO
23 RC4/SDI/SDA
22 RD3/PSP3
21 RD2/PSP2
PIC16F877A-I/P
1444022
MICROCHIP

the flowchart for programming

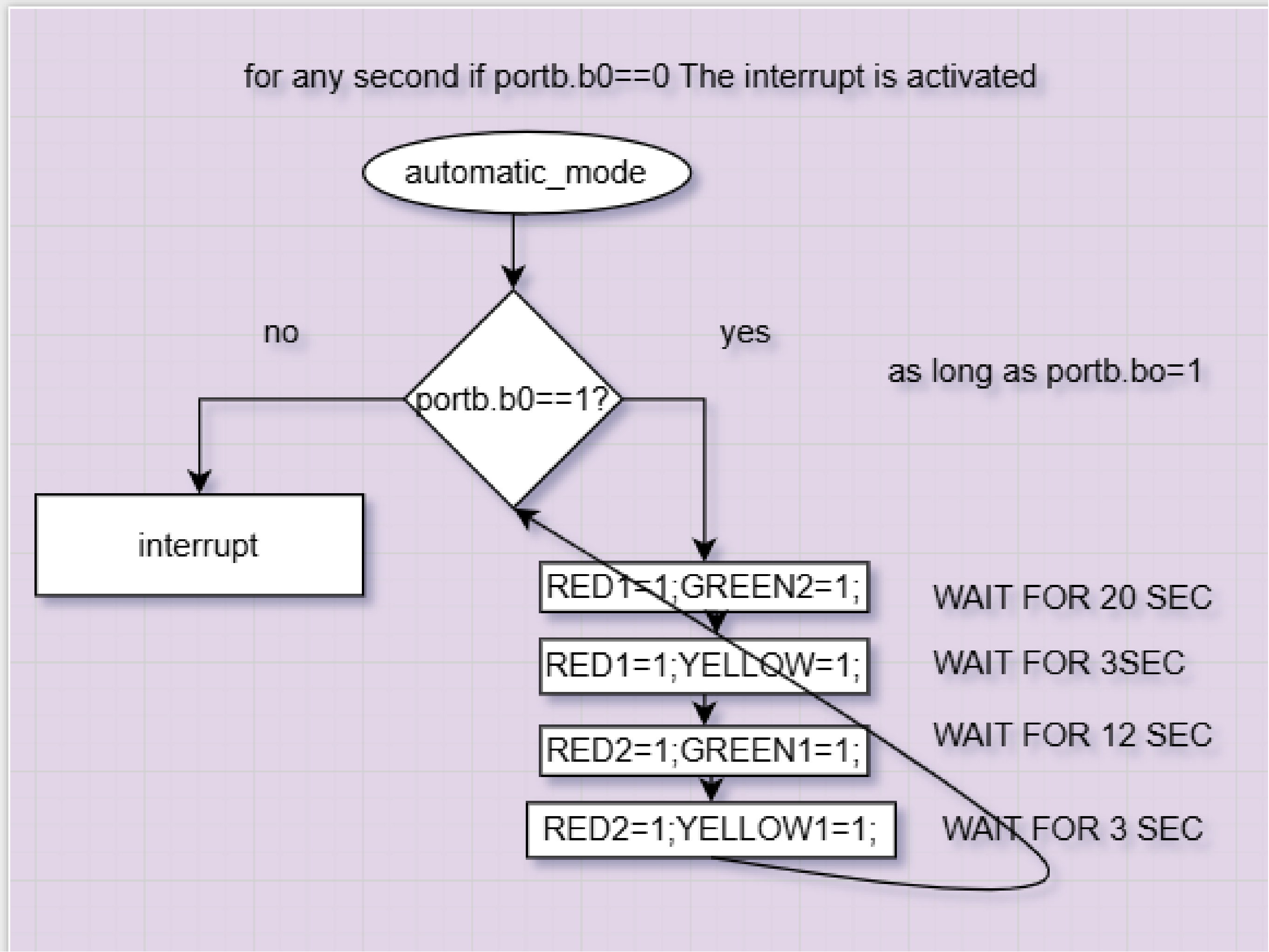
Here, I will begin by drawing a general relationship between all the functions used in the code. I will arrange them using a flowchart to illustrate the overall flow of the TRAFFIC LIGHT CONTROLLER program. I will start with the main function, where the program execution begins. In this code, I will set up an interrupt condition on pin B0 of PORTB, specifically on the falling edge. During continuous operation, if the interrupt is not triggered, the program will remain in a loop calling the automatic_mode function, while continuously monitoring for the interrupt condition. At any moment, if the interrupt condition is met (i.e., when PORTB.B0 == 0), its command will be executed. When the interrupt is triggered: if PORTB.B1 == 1, the reading on the 7-segment display will remain at zero. If PORTB.B1 == 0, the program will transition to the manual_mode function to display the yellow light along with the two primary colors, which are toggled when PORTB.B1 is pressed. This state is one of the most suitable for indicating an imminent signal change, with a countdown from 3 to 1, after which the opposite operation will occur, taking into account any changes that happen when the buttons are pressed.





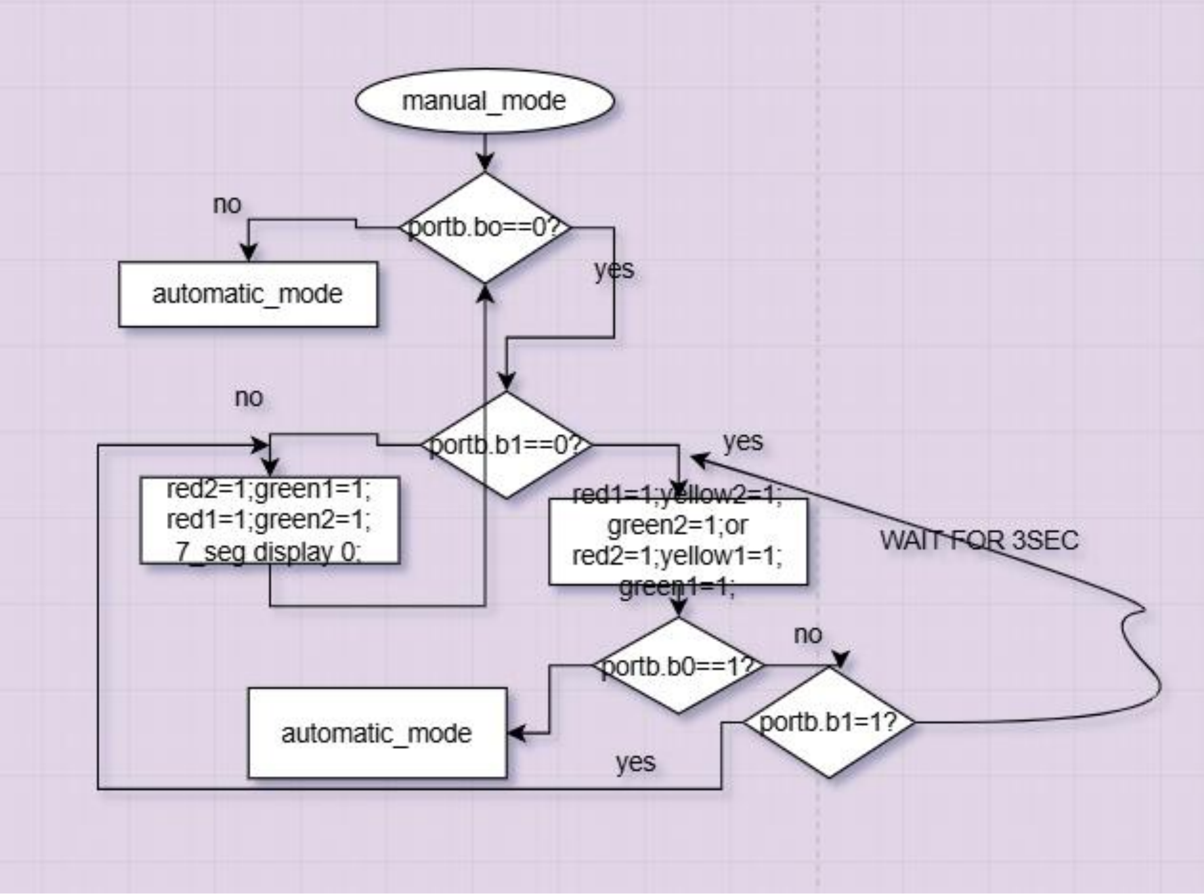
Here's how the automatic_mode function operates: When PORTB.B0 == 1, it functions regardless of the value of PORTB.B1. The switch represented by PORTB.B0 acts as a toggle between manual and automatic modes. In the normal state, without pressing the PORTB.B0 switch, the expected lighting for the South Street, displayed on the screen, first involves illuminating the red LED for 23 seconds. Concurrently, the green LED on the East Street illuminates for 20 seconds, followed by the yellow light for the East Street for 3 seconds. The code then transitions to the second loop, which will involve illuminating the red LED for the East Street for 15 seconds. Correspondingly, the green LED for the South Street will illuminate for 12 seconds, followed immediately by the yellow LED for the South Street for 3 seconds.

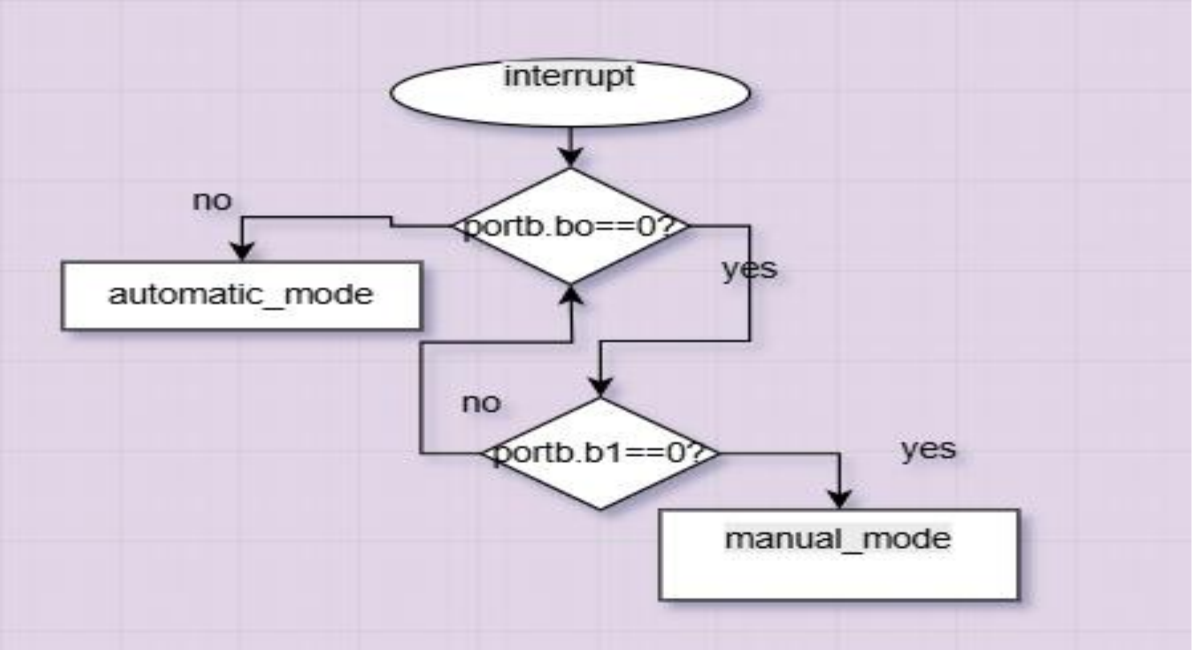
However, at any second the switch for toggling between automatic and manual mode is pressed, it will transition to the interrupt function.

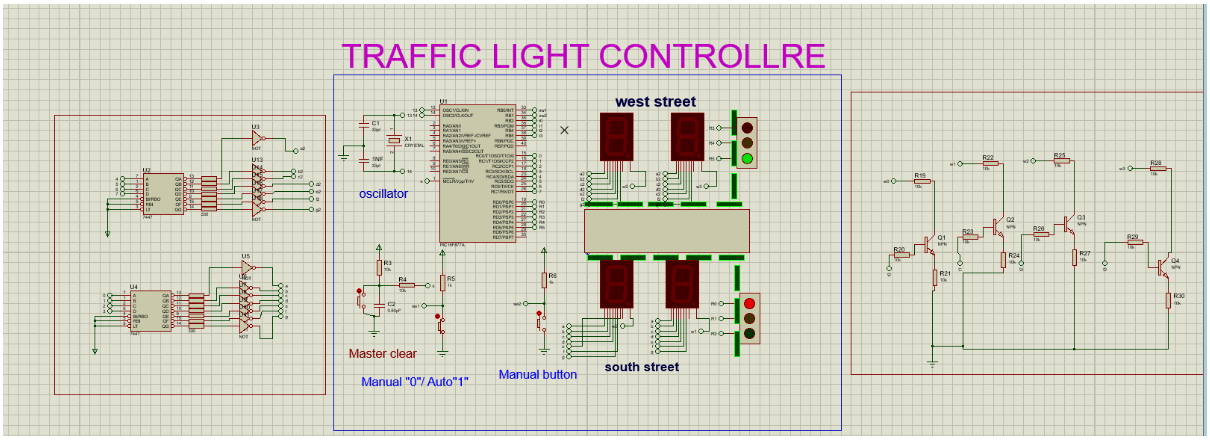


Upon transitioning to the interrupt state, if the button responsible for manual mode on PORTB.B1 is pressed (i.e., PORTB.B1 == 0), it will switch to the manual_mode function. This function is responsible for a three-second countdown, then reversing the signal and holding it in the reversed state until the manual mode button is released (i.e., when PORTB.B1 == 1).

Exiting the interrupt mode occurs at any moment the switch connected to PORTB.B0 is released, and the counting resumes from where it was paused.






TRAFFIC LIGHT CONTROLLRE
west street
oscillator
Master clear
Manual "0"/ Auto"1"
Manual button
south street