Servo Angle Detector

8051’s Microcontroller

Team:

Mahmoud Morsy Abdallah Abdelrahman

Abdelrahman Mohamed AboAlfotoh

Mahmoud Atif Ahmed

Mahmoud Sherif Abdalazem

Mahmoud Hamed Alsaid

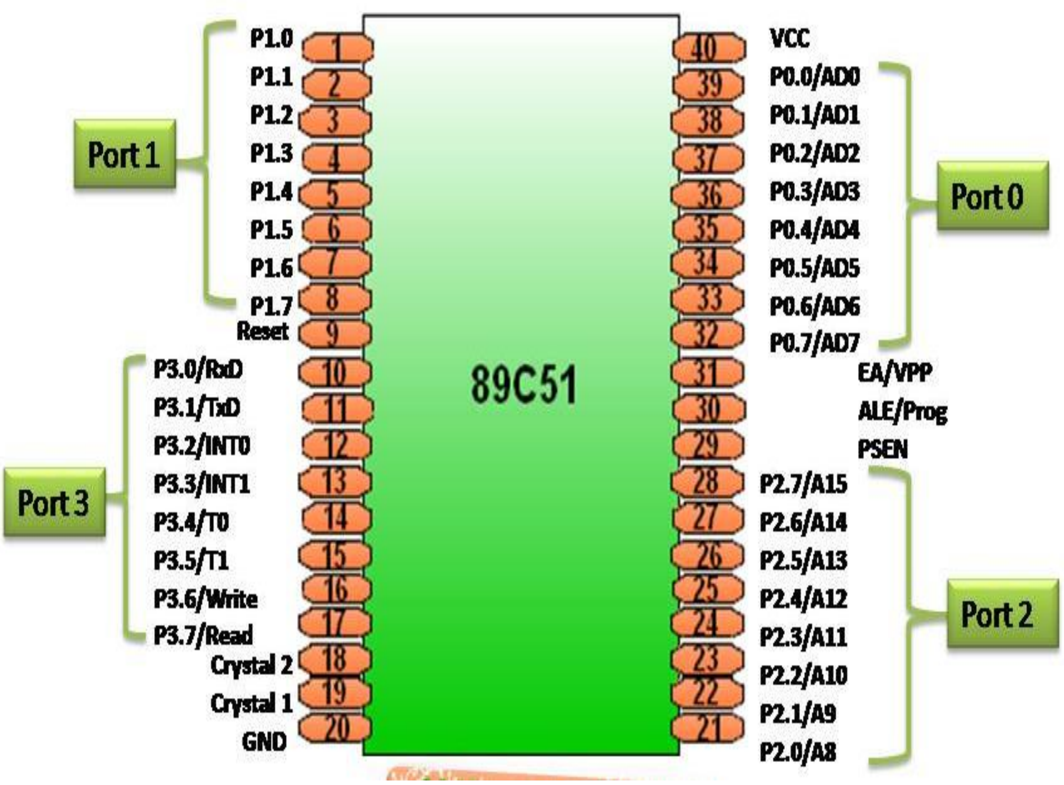
Supervisor: **ENG. Fatma Naser**

Project Definition:

The project aims to create an interface for a microcontroller using Assembly language to monitor and display the angle of rotation for a servo motor on a seven-segment display. The project utilizes a timer to generate a Pulse Width Modulation (PWM) signal, which controls the servo motor's position based on the desired angle of rotation.

Key Components:

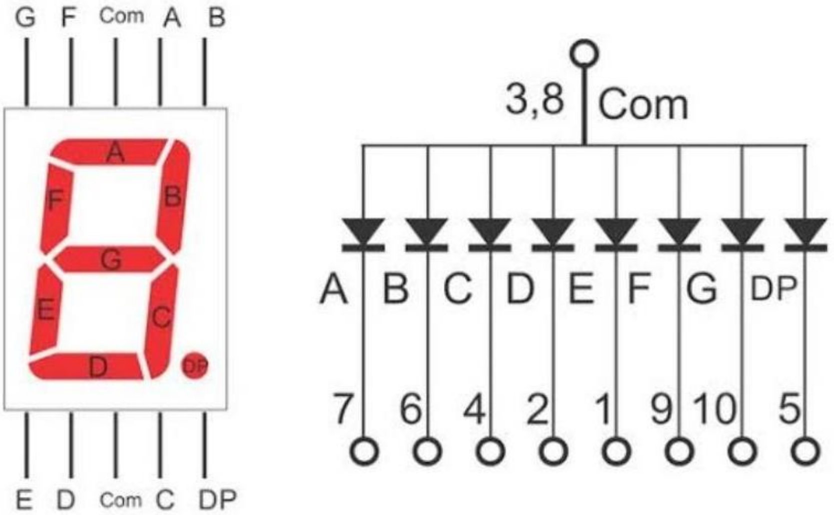
1. **AT89C51 Microcontroller:** This microcontroller, like the 8051 Microcontroller, serves as the main control unit for the project. It provides the necessary processing power and I/O capabilities to interface with the various components.



2. **Crystals (12MHz):** Two crystals are used to provide the necessary clock signal for the microcontroller, ensuring accurate timing and synchronization of operations.

3. **Push Buttons:** Seven push buttons are incorporated into the interface to allow users to input the desired angle of rotation for the servo motor. These buttons enable the user to increment or decrement the angle and provide control over the rotation.

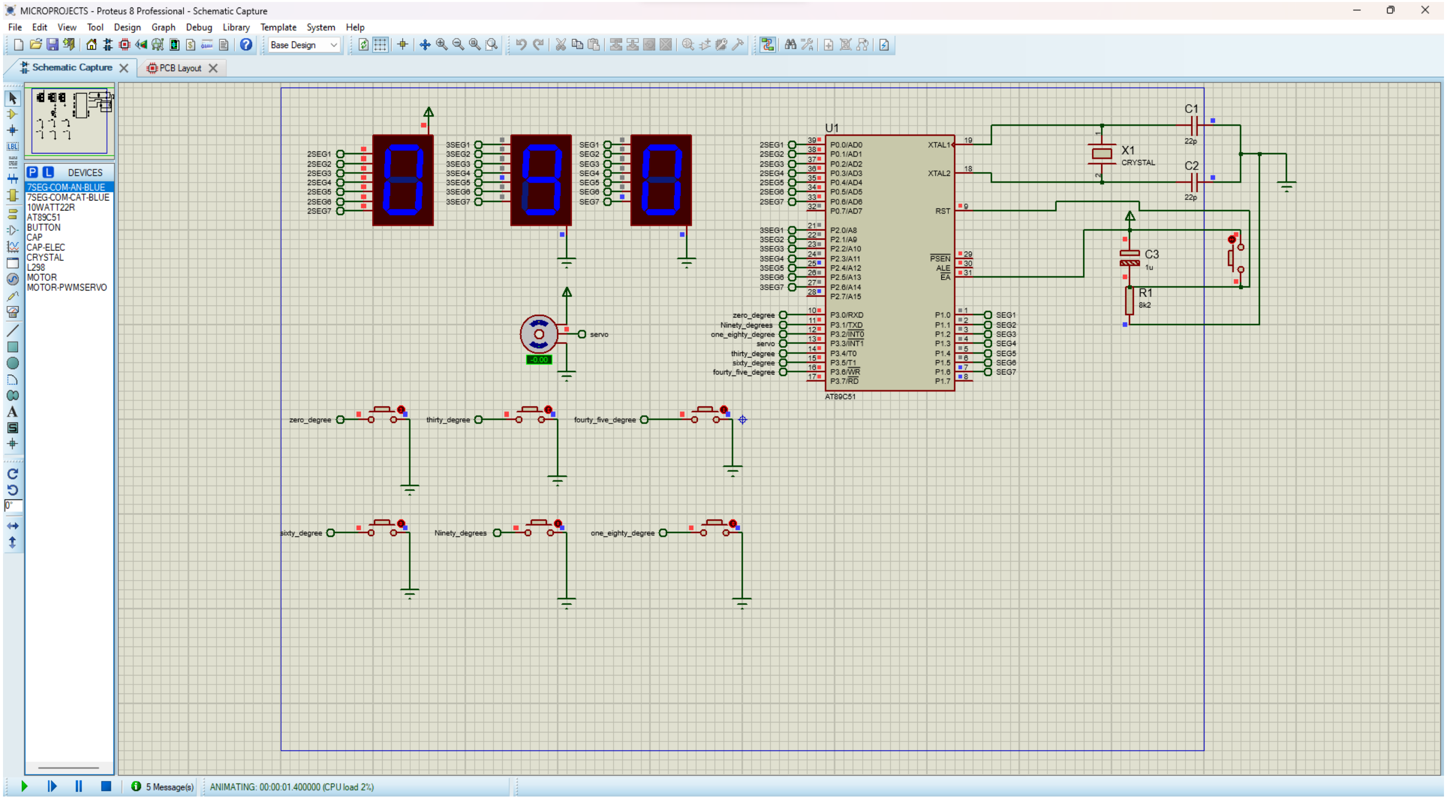
4. **Seven-Segment Displays:** Three seven-segment displays are utilized to visually represent the angle of rotation. These displays show the numerical value of the angle in a clear and readable format.



5. **Servo Motor (SG90):** The servo motor is the output component in the project, responsible for physically rotating based on the control signal received from the microcontroller. The PWM signal generated by the timer is used to control the rotation angle of the servo motor.

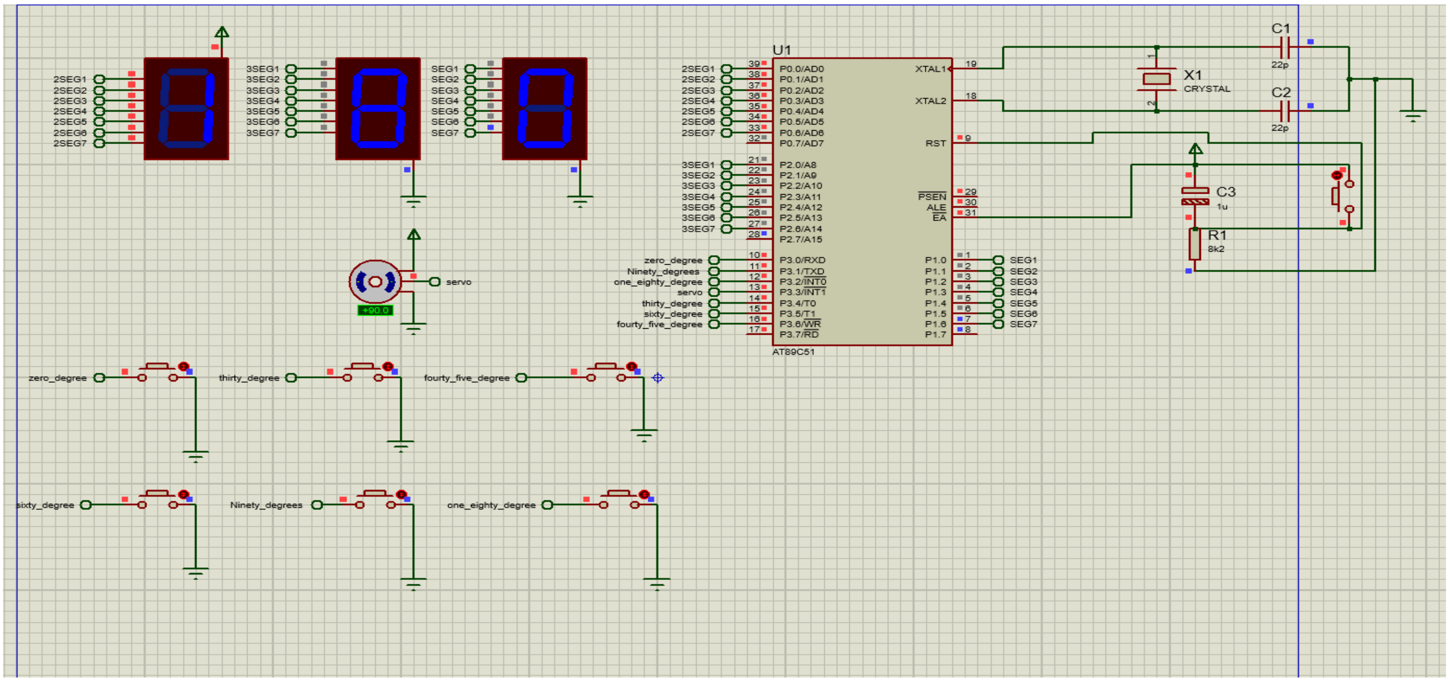
The project involves programming the microcontroller in Assembly language to handle user input from the push buttons, generate the PWM signal using the timer, and display the angle of rotation on the seven-segment displays. The servo motor responds to the PWM signal, adjusting its position to match the desired angle. This interface provides a user-friendly way to control and monitor the servo motor's rotation angle.

Simulations:



MICROPROJECTS - Proteus 8 Professional - Schematic Capture

ANIMATING: 00:00:01.400000 (CPU load 2%)



Code:

```
ORG 00H //Start the program.

MAIN:

MOV TMOD, #01H ; using Timer 0 in Mode 1

Zero:JNB p3.0,Zero

    JNB p3.1, Ninty

            JNB p3.2, OneEighty

            JNB p3.4,Thirty

            JNB p3.5,Sixty

            JNB p3.6,FourtyFive

            ;JNB p3.7,OneTwenty

    LCALL zero_degrees ;Function to move to position = 0 deg

            SJMP Zero

Ninty:

     JNB p3.0,Zero

     JNB p3.1,Ninty

             JNB p3.2,OneEighty

             JNB p3.4,Thirty

             JNB p3.5,Sixty

             JNB p3.6,FourtyFive

             ;JNB p3.7,OneTwenty

     LCALL ninety_degrees ;Function to move to position = 90 deg

             SJMP Ninty

OneEighty:

     JNB p3.0,Zero

     JNB p3.1,Ninty

             JNB p3.2,OneEighty

             JNB p3.4,Thirty

             JNB p3.5,Sixty

             JNB p3.6,FourtyFive

             ;JNB p3.7,OneTwenty

     LCALL one_eighty_degrees ;Function to move to position = 180 deg

             SJMP OneEighty

Thirty:

     JNB p3.0,Zero

     JNB p3.1,Ninty

             JNB p3.2,OneEighty

             JNB p3.4,Thirty

             JNB p3.5,Sixty

             JNB p3.6,FourtyFive

             ;JNB p3.7,OneTwenty

     LCALL Thirty_degrees ;Function to move to position = 30 deg

             SJMP Thirty

Sixty:

     JNB p3.0,Zero

     JNB p3.1,Ninty

             JNB p3.2,OneEighty

             JNB p3.4,Thirty

             JNB p3.5,Sixty

             JNB p3.6,FourtyFive

             ;JNB p3.7,OneTwenty

     LCALL Sixty_degrees ;Function to move to position = 60 deg

             SJMP Sixty

FourtyFive:

     JNB p3.0,Zero

     JNB p3.1,Ninty

             JNB p3.2,OneEighty

             JNB p3.4,Thirty

             JNB p3.5,Sixty
            JNB p3.6,FourtyFive

           ; JNB p3.7,OneTwenty

     LCALL Fourty_Five_degrees ;Function to move to position = 60 deg

            SJMP FourtyFive

LJMP MAIN ;to repeat the loop until manually stopped

RET

zero_degrees: //To create a pulse of 1ms

MOV P1,#3fH

MOV P2,#3fH

Mov A ,#3fH

cpl A

MOV P0,A

MOV TH0,#0FCH //(FFFF - FC18 + 1)H = (03E8)H

MOV TL0,#1CH //equal TO (1000)D = 1ms

SETB P3.3 ;Make P3.3 HIGH

SETB TR0 ;Start the timer 0

WAIT1:JNB TF0, WAIT1 ;Wait till the TF0 flag is set

CLR P3.3 ;Make P3.3 LOW

CLR TF0  ;Clear the flag manually

CLR TR0  ;Stop the timer 0

RET

Thirty_degrees: //To create a pulse of 1.1665ms

MOV P1,#3fH

MOV P2,#4fH

Mov A ,#3fH

cpl A

MOV P0,A

MOV TH0,#0FBH //(FFFF - FB73 + 1)H = (048D)H

MOV TL0,#74H //equal TO (1165)D = 1.165ms

SETB P3.3 ;Make P3.3 HIGH

SETB TR0 ;Start the timer 0

WAIT4:JNB TF0, WAIT4 ;Wait till the TF0 flag is set

CLR P3.3 ;Make P3.3 LOW

CLR TF0  ;Clear the flag manually

CLR TR0  ;Stop the timer 0

RET

Fourty_Five_degrees: //To create a pulse of 1.1665ms

MOV P1,#6DH

MOV P2,#66H

Mov A ,#3fH

cpl A

MOV P0,A

MOV TH0,#0FBH //(FFFF - FB73 + 1)H = (048D)H

MOV TL0,#20H //equal TO (1165)D = 1.165ms

SETB P3.3 ;Make P3.3 HIGH

SETB TR0 ;Start the timer 0

WAIT6:JNB TF0, WAIT6 ;Wait till the TF0 flag is set

CLR P3.3 ;Make P3.3 LOW

CLR TF0  ;Clear the flag manually

CLR TR0  ;Stop the timer 0

RET

Sixty_degrees: //To create a pulse of 1.333ms

MOV P1,#3fH

MOV P2,#7DH

Mov A ,#3fH

cpl A

MOV P0,A
MOV TH0,#0FAH //(FFFF - FACD + 1)H = (0533)H

MOV TL0,#0CDH //equal TO (1331)D = 1.331ms

SETB P3.3 ;Make P3.3 HIGH

SETB TR0 ;Start the timer 0

WAIT5:JNB TF0, WAIT5 ;Wait till the TF0 flag is set

CLR P3.3 ;Make P3.3 LOW

CLR TF0  ;Clear the flag manually

CLR TR0  ;Stop the timer 0

RET


ninety degrees: //To create a pulse of 1.5ms

MOV P1,#3fH

MOV P2,#6fH

Mov A ,#3fH

cpl A

MOV P0,A

MOV TH0,#0FAH //(FFFF - FA24 + 1)H = (05DC)H

MOV TL0,#26H //equal to (1500)D = 1.5ms

SETB P3.3 ;Make P3.3 HIGH

SETB TR0 ;Start the timer 0

WAIT2:JNB TF0, WAIT2 ;Wait till the TF0 flag is set

CLR P3.3 ;Make P3.3 LOW

CLR TF0 ;Clear the flag manually

CLR TR0 ;Stop the timer 0

RET

one_eighty_degrees: //To create a pulse of 2ms

MOV P1,#3fH

MOV P2,#7fH

Mov A ,#06H

cpl A

MOV P0,A

MOV TH0,#0F8H //(FFFF - F830 + 1)H = (07D0)H

MOV TL0,#30H //equal to (2000)D = 2ms

SETB P3.3 ;Make P3.3 HIGH

SETB TR0 ;Start the timer 0

WAIT3:JNB TF0, WAIT3 ;Wait till the TF0 flag is set

CLR P3.3 ;Make P3.3 LOW

CLR TF0 ;Clear the flag manually

CLR TR0 ;Stop the timer 0

RET

END
```