Department : Computer Engineering
Course Code : CC421
Course Name : Microprocessor Systems
Lecturer : Prof. Ahmed Fahmy

PARKING SYSTEMS

Presented By:

ID	Name	G1	G2	G3	D1	D2	D3	D4
211004633	TARIQ ALI ELYOUSFY							
211008104	AHMAD MOHAMMAD ELBAATHY							
211004746	ZIAD ESMAT							

Agenda

Summary:

The parking lot management system leverages servo motor control and LCD integration to automate parking slot availability and gate access, enhancing user convenience and operational efficiency. This system provides real-time feedback to users regarding parking slots and seamlessly operates gates using servo mechanisms. Commands and actions are processed based on pre-defined inputs, ensuring modular and scalable functionality.
The system's modular design allows it to be installed in various parking spaces, simplifying the process of slot management and access control. The main control unit coordinates inputs, processes parking slot availability, and provides real-time status updates via an LCD. The gate's servo motor operates based on user input or slot availability, making the system both user-friendly and practical.

Introduction

The parking lot system integrates a microcontroller with peripherals such as a servo motor and LCD display to efficiently manage parking slots and gate control. Input signals from designated ports determine slot availability, while the servo motor simulates gate movements, and the LCD provides real-time user feedback.

LEDs represent the status of parking slots, while precise PWM signals operate the servo motor, allowing it to mimic gate opening and closing at specific angles. A modular design enables scalability for larger parking systems or additional gates. Timer 1, configured in Mode 1, ensures precise pulse-width modulation, and the system dynamically monitors user inputs for real-time responsiveness.

Methodology

System Initialization

The microcontroller initializes key modules:

- **Timer 1** for PWM generation.
- Port configurations for servo motor and LCD control.
- LCD setup to enable data display and user feedback.

Servo Motor Control

The servo motor's position is controlled using precise pulse durations:

- **Center position**: The gate remains closed.
- **Left position**: The gate opens fully. Pulse durations are managed through high and low signal transitions using Timer 1 subroutines.

Parking Slot Detection

Slot availability is monitored via input signals on `P0.1`, `P0.2`, and `P0.3`. Each input pattern corresponds to a unique parking status:

- Available slots are dynamically displayed on the LCD.
- The system adjusts gate movements and messages based on parking conditions.

User Feedback and Display

A 16x2 LCD displays:

- Welcome messages.
- Slot availability updates.
- Gate status (open/closed). Pre-stored messages are selected based on real-time conditions.

Gate Control

Gate movements are determined by:

- Parking slot availability.
- External user commands (button presses). Servo position adjustments use high-precision signals generated by the timer module.

System Operation Loop

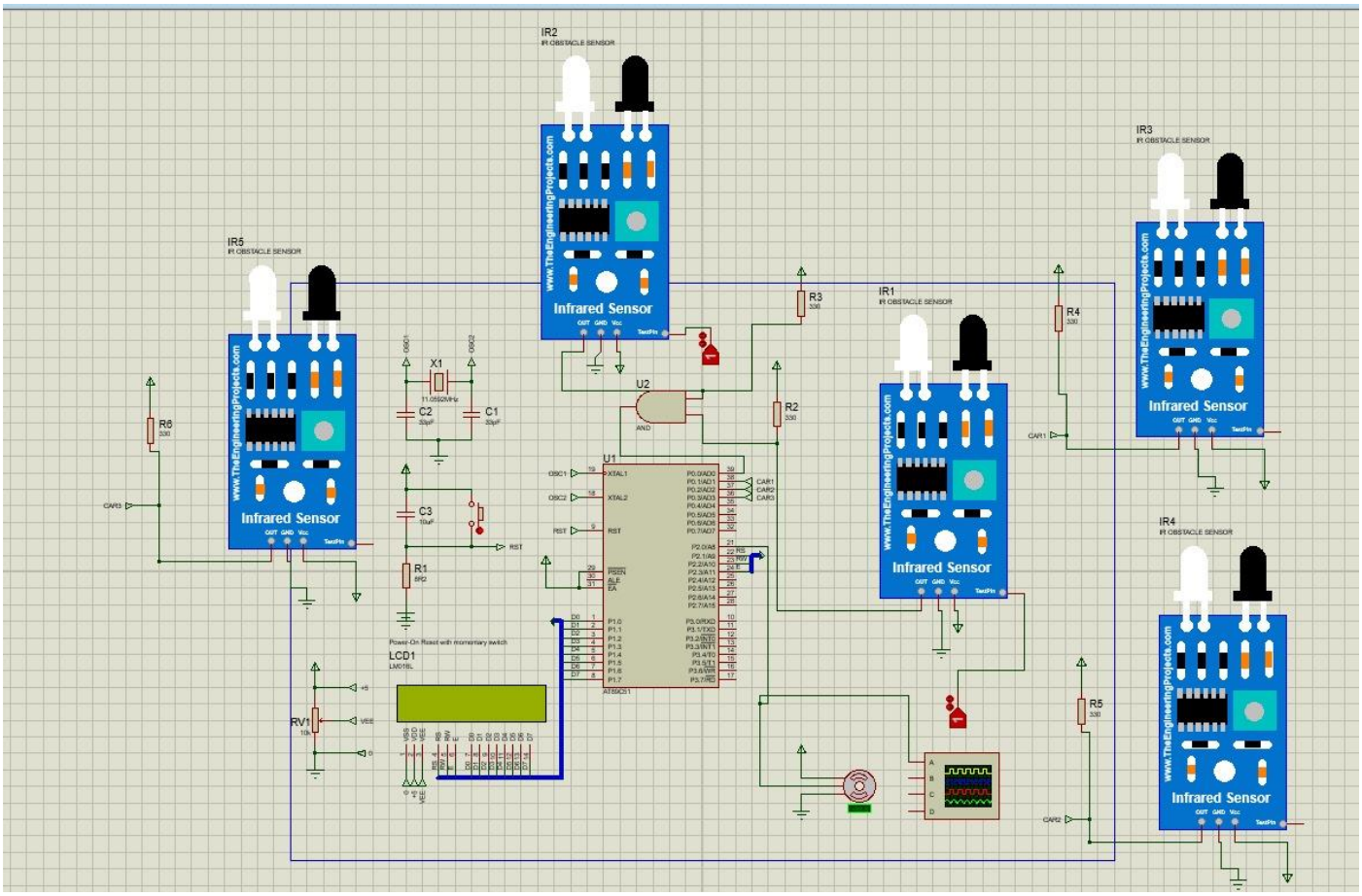
The program runs in an infinite loop to:

- Continuously monitor parking slot availability and user inputs.
- Dynamically update servo positions and LCD messages.

Power Management

External components such as the servo motor and LCD are powered via transistors or relays, as the microcontroller alone cannot directly supply sufficient current.

Schematic:


IR2
IR OBSTACLE SENSOR
Infrared Sensor
IR5
IR OBSTACLE SENSOR
Infrared Sensor
IR3
IR OBSTACLE SENSOR
Infrared Sensor
IR1
IR OBSTACLE SENSOR
Infrared Sensor
IR4
IR OBSTACLE SENSOR
Infrared Sensor
U1
U2
X1
C1
C2
C3
R1
R2
R3
R4
R5
R6
RV1
LCD1
Power-On Reset with momentary switch

Components and Tools:

- 5 IR sensors
- 330 Resistors x5
- Potentiometer
- LCD
- Push button
- 10 uF Capacitor
- Servo motor
- 33 pF Capacitors x2
- 7404 IC

Code Text:

```
ORG 0H
; SERVO REQUIRES A SIGNAL OF 50Hz WHICH IS 20ms.
; Position "0" ( 1.5 ms of 20 ms pulse) turns in the middle.
; Position "+90" ( 2ms of 20 ms pulse) turns all the way to the right.
; Position "-90" ( 1ms of 20 ms pulse) turns all the way to the left.
RS EQU P2.1
RW EQU P2.2
E EQU P2.3
CSG EQU P2.4
PARK1 EQU P0.1
PARK2 EQU P0.2
PARK3 EQU P0.3

;LCD INITIALIZATION

            MOV A, #00111000B       ; INITIATE LCD #38H
            ACALL COMMANDWRT
            ACALL DELAY

            MOV A, #00001110B       ; DISPLAY ON CURSOR OFF #0CH
            ACALL COMMANDWRT
            ACALL DELAY



            MOV A, #00000001B       ; CLEAR LCD #01H
            ACALL COMMANDWRT
            ACALL DELAY

;PRINTING A STRING

            MOV DPTR, #STRINGDATA1
STRING1:    CLR A
            MOVC A, @A+DPTR
            ACALL DATAWRT
            ACALL DELAYMSG
            INC DPTR

JNZ STRING1
```

```
ACALL PARKCHECK


JB P0.0, $                    ;not read in hardware



MOV TMOD, #00010000B; TIMER 1, MODE 1 - 16BIT MODE

MAIN:

JB P0.0, CON1

SJMP TURN_LEFT_90

CON1:
CALL CENTER_POS
SJMP MAIN

TURN_LEFT_90:
ACALL CLEARMSG
SETB CSG

CALL LEFT_POS
                SJMP MAIN


;;;;;;;;;;;;;;;;;SERVO POSITIONS;;;;;;;;;;;;;;;;;;;;

CENTER_POS:
; Position "0" (1.5 ms  pulse) is middle.
; High Part = 1.5ms
; 1.5ms / (1 / 11.0592 MHz) = 1.5 ms / 1.085 µs = 1382
; 65536 - 1382 = 64154 Dec = FA9A Hex
; Low Part = 18.5ms
; 18.5ms / (1 / 11.0592 MHz) = 18.5 ms / 1.085 µs = 17050
; 65536 - 17050 = 48486 Dec = BD66 Hex


;JNB CSG,CONTINUE
```

```
;JB P0.0,GC




        SETB P2.0
        MOV TL1, #0B2H    ;LET THIS VALUE = 0B2 FOR A PRECISE CENTER ANGLE AS MUCH
AS POSSIBLE
        MOV TH1, #0FAH   ;Puts it back to the center
        SETB TR1      ; Run Timer

        CALL HIGH_SIGNAL

        MOV TL1, #66H
        MOV TH1, #0BDH   ;        Allow it to turn
        SETB TR1      ; Run Timer

        CALL LOW_SIGNAL
JNB CSG,CONTINUE
        ACALL STR3WRT
CONTINUE:

        RET

LEFT_POS:
; Position "-90" (1 ms  pulse) is 90 degrees to the left.
; High Part = 1 ms
; 1 ms / (1 / 11.0592 MHz) = 1 ms / 1.085 µs = 921
; 65536 - 921 = 64615 Dec = FC67 Hex
; Low Part = 19 ms
; 19 ms / (1 / 11.0592 MHz) = 19 ms / 1.085 µs = 17511
; 65536 - 17511 = 48025 Dec = BB99 Hex
                ACALL CLEARMSG
                MOV DPTR, #STRINGDATA2
STRING2:        CLR A
                MOVC A, @A+DPTR
                ACALL DATAWRT
                INC DPTR




                JNZ STRING2
```

```
        ACALL CLEARMSG



    MOV TL1, #7CH      ;LET THIS VALUE = 7C FOR A PRECISE -90 ANGLE
    MOV TH1, #0FFH
    SETB TR1      ; Run Timer

    CALL HIGH_SIGNAL

    MOV TL1, #99H
    MOV TH1, #0BBH
    SETB TR1      ; Run Timer

    CALL LOW_SIGNAL





    RET



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;TIMER OF SIGNALS;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
HIGH_SIGNAL:
    JNB TF1, $    ;Wait till Timer overflow
    CLR TR1
    CLR P2.0      ;HIGH TO LOW TRANSITION
    CLR TF1
    RET

LOW_SIGNAL:
    JNB TF1, $    ;Wait till Timer overflow
    CLR TR1
    SETB P2.0     ;LOW TO HIGH TRANSITION
    CLR TF1
    RET
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
;COMMAND SUB-ROUTINE FOR LCD CONTROL
COMMANDWRT:

   MOV P1, A ;SEND DATA TO P1
        CLR RS      ;RS=0 FOR COMMAND
        CLR RW      ;R/W=0 FOR WRITE
        SETB E      ;E=1 FOR HIGH PULSE
        ACALL DELAY ;SOME DELAY
        CLR E  ;E=0 FOR H-L PULSE

        RET

;SUBROUTINE FOR DATA LACTCHING TO LCD
DATAWRT:

        MOV P1, A
        SETB RS     ;;RS=1 FOR DATA
        CLR RW
        SETB E
        ACALL DELAY
        CLR E

        RET



DELAY:
        MOV R0, #255
X:      MOV R1, #255
        DJNZ R1, $
        DJNZ R0, X
        RET


DELAYMSG:
        MOV R0, #1
Y:      MOV R1, #1
        DJNZ R1, $
        DJNZ R0, Y
        RET
```

```
CLEARMSG: MOV A, #00000001B       ; CLEAR LCD #01H
ACALL COMMANDWRT
ACALL DELAYMSG
RET

STR3WRT:
CLR CSG
MOV DPTR, #STRINGDATA3

STRING3:    CLR A
            MOVC A, @A+DPTR
            ACALL DATAWRT
            ACALL DELAYMSG
            INC DPTR


            JNZ STRING3
ACALL PARKCHECK

RET

STR4WRT:

MOV DPTR, #STRINGDATA4
STRING4:    CLR A
            MOVC A, @A+DPTR
            ACALL DATAWRT
            ACALL DELAYMSG
            INC DPTR


            JNZ STRING4

RET

STR5WRT:

MOV DPTR, #STRINGDATA5
STRING5:    CLR A
            MOVC A, @A+DPTR
            ACALL DATAWRT
            ACALL DELAYMSG
```

```
            INC DPTR


            JNZ STRING5

RET



STR6WRT:

MOV DPTR, #STRINGDATA6
STRING6:    CLR A
            MOVC A, @A+DPTR
            ACALL DATAWRT
            ACALL DELAYMSG
            INC DPTR


            JNZ STRING6

RET


STR7WRT:
        MOV DPTR, #STRINGDATA7
        STRING7:    CLR A
            MOVC A, @A+DPTR
            ACALL DATAWRT
            ACALL DELAYMSG
            INC DPTR


            JNZ STRING7

RET

PARKCHECK:

MOV A,#0C0H
ACALL COMMANDWRT
```

```
;0000 -> 0H
;0010 -> 2H
;0100 -> 4H
;0110 -> 6H
;1000 -> 8H
;1010 -> AH
;1100 -> CH
;1110 -> EH



MOV A,P0
ANL A,#00001110B


CJNE A,#0H,NEXT1
ACALL STR4WRT
NEXT1:
CJNE A,#2H,NEXT2 ;;;;;;;;;;;;;;;
ACALL STR5WRT
NEXT2:
CJNE A,#4H,NEXT3
ACALL STR5WRT
NEXT3:
CJNE A,#8H,NEXT4
ACALL STR5WRT
NEXT4:
CJNE A,#6H,NEXT5 ;;;;;;;;;;;;;;;;;;;
ACALL STR6WRT
NEXT5:
CJNE A,#0AH,NEXT6 ;;;;;;;;;;;;;;;;
ACALL STR6WRT
NEXT6:
CJNE A,#0CH,NEXT7
ACALL STR6WRT
NEXT7:
CJNE A,#0EH,NEXT8
ACALL STR7WRT
NEXT8:

RET
```

```
ORG 300H

STRINGDATA1:    DB    "Welcome !!!" ,0
STRINGDATA2:    DB    "Gate Opening " ,0
STRINGDATA3:    DB    "Gate Closed" ,0
STRINGDATA4:    DB    "Slots Left: 3" ,0
STRINGDATA5:    DB    "Slots Left: 2" ,0
STRINGDATA6:    DB    "Slots Left: 1" ,0
STRINGDATA7:    DB    "Slots Left: 0" ,0




END
```

Code Illustration

Pseudocode

```
BEGIN
  Configure Timer1 in 16-bit mode (Mode 1)
  Initialize LCD:
    Send Command: Set LCD to 8-bit mode
    Send Command: Turn on Display, Cursor off
    Send Command: Clear Display
  Display "Welcome!!!" on LCD


WHILE TRUE DO
  IF Parking sensor at entrance (P0.0) is triggered THEN
    Open Gate:
      Move Servo to CENTER position (1.5ms pulse)
      Display "Gate Opening" on LCD
    WAIT until car passes
    Close Gate:
      Move Servo to LEFT position (-90° or 1ms pulse)
      Display "Gate Closed" on LCD
  END IF

  Check parking slots:
    Read sensors on P0.1, P0.2, P0.3
    Determine available slots:
      CASE of combined sensor value:
        0H: Display "Slots Left: 3"
        2H: Display "Slots Left: 2"
        4H: Display "Slots Left: 1"
        6H: Display "Slots Left: 0"
    END CASE
  END CHECK
END WHILE
```

```
SUBROUTINE CENTER_POS
   Set P2.0 to HIGH (Servo signal start)
   Generate 1.5ms High Pulse:
      Load Timer1 registers for 1.5ms delay
      Start Timer1
      Wait for Timer1 overflow
   Set P2.0 to LOW
   Generate 18.5ms Low Pulse:
      Load Timer1 registers for 18.5ms delay
      Start Timer1
      Wait for Timer1 overflow
RETURN

SUBROUTINE LEFT_POS
   Set P2.0 to HIGH
   Generate 1ms High Pulse:
      Load Timer1 registers for 1ms delay
      Start Timer1
      Wait for Timer1 overflow
   Set P2.0 to LOW
   Generate 19ms Low Pulse:
      Load Timer1 registers for 19ms delay
      Start Timer1
      Wait for Timer1 overflow
RETURN
```

```
SUBROUTINE CENTER_POS
   Set P2.0 to HIGH (Servo signal start)
   Generate 1.5ms High Pulse:
      Load Timer1 registers for 1.5ms delay
      Start Timer1
      Wait for Timer1 overflow
   Set P2.0 to LOW
   Generate 18.5ms Low Pulse:
      Load Timer1 registers for 18.5ms delay
      Start Timer1
      Wait for Timer1 overflow
RETURN

SUBROUTINE LEFT_POS
   Set P2.0 to HIGH
   Generate 1ms High Pulse:
      Load Timer1 registers for 1ms delay
      Start Timer1
      Wait for Timer1 overflow
   Set P2.0 to LOW
   Generate 19ms Low Pulse:
      Load Timer1 registers for 19ms delay
      Start Timer1
      Wait for Timer1 overflow
RETURN

SUBROUTINE COMMANDWRT (COMMAND)
   Send COMMAND to P1 (LCD Data Port)
   Set RS = 0 (Command Mode)
   Set RW = 0 (Write Mode)
   Generate High-to-Low Enable Pulse on E
   Delay for LCD to process
RETURN

SUBROUTINE DATAWRT (DATA)
   Send DATA to P1 (LCD Data Port)
   Set RS = 1 (Data Mode)
   Set RW = 0 (Write Mode)
   Generate High-to-Low Enable Pulse on E
   Delay for LCD to process
RETURN
```

```
SUBROUTINE PARKCHECK
   Read Parking Sensor Bits (P0.1-P0.3)
   Mask and Decode Available Slots:
      CASE of masked bits:
         0H: Display "Slots Left: 3"
         2H: Display "Slots Left: 2"
         4H: Display "Slots Left: 1"
         6H: Display "Slots Left: 0"
   END CASE
RETURN
```

References:

- Eng. Mohamed Samy
- Chatgpt