|  |
| --- |
| **EL-303 - Microprocessor Interfacing & Programming** |
| **Fall 2020**  **Semester Project**  **Phase 1 & 2**  **Alishbah Gul 18I-0858**  **Ayesha Ishaq 18I-0772**  **SMART PARKING MANAGEMENT SYSTEM** |

**Overview:**

The aim of this project is to implement a system that can automatically maintain the number of customers using a parking space to prevent overcrowding. This system will also keep record of the time each customer uses a parking space and charge them accordingly. At the entrance, upon detection of a car, the availability of parking spaces will be checked (which will be displayed at the entrance), and the barrier will be lifted if there is any vacant space. The system will allot the customer a token and will record their time of entrance against the token. Each parking space will be fitted with a sensor to allow the program to display at the entrance where the available parking spaces are, to guide the customer through the parking lot. Each time a car enters, the number of parking spaces will be decremented. At the exit, the customer will provide their token, and their fee will be calculated for the time they used a parking space. Once paid, the barrier will be lifted to allow them to exit. Parking spaces will be incremented when a car leaves.

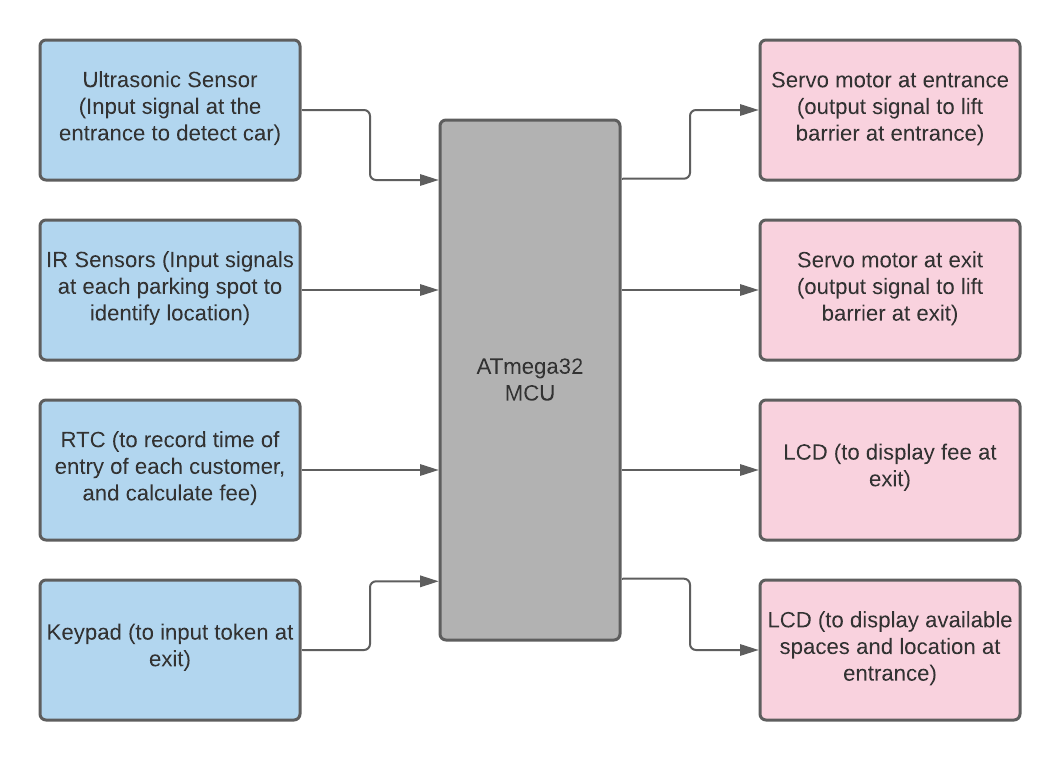
**Input Signals:**

* Keypad at the exit.
* Car detection sensor at the entrance.
* Overhead sensors at parking spots
* RTC

**Output Signals:**

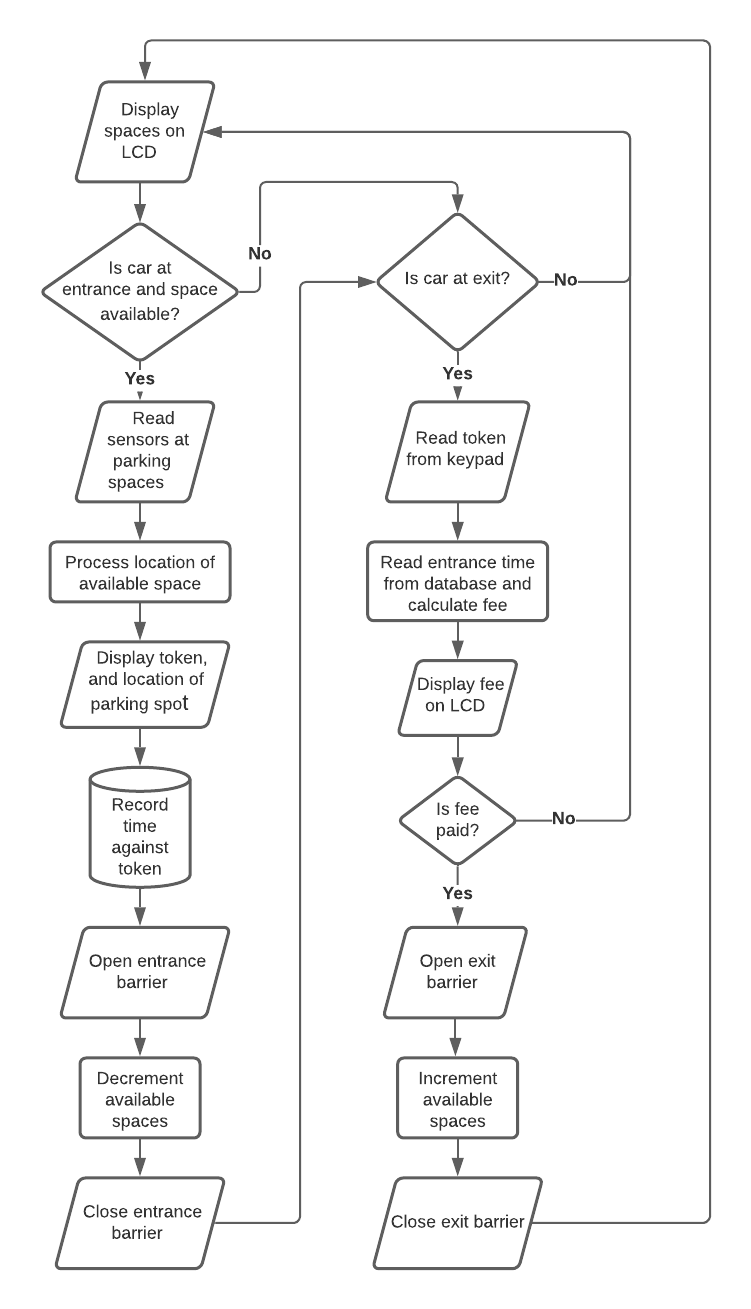
* Motor at entrance.
* LCD at entrance and exit.
* Motor at exit.

**BLOCK DIAGRAM**

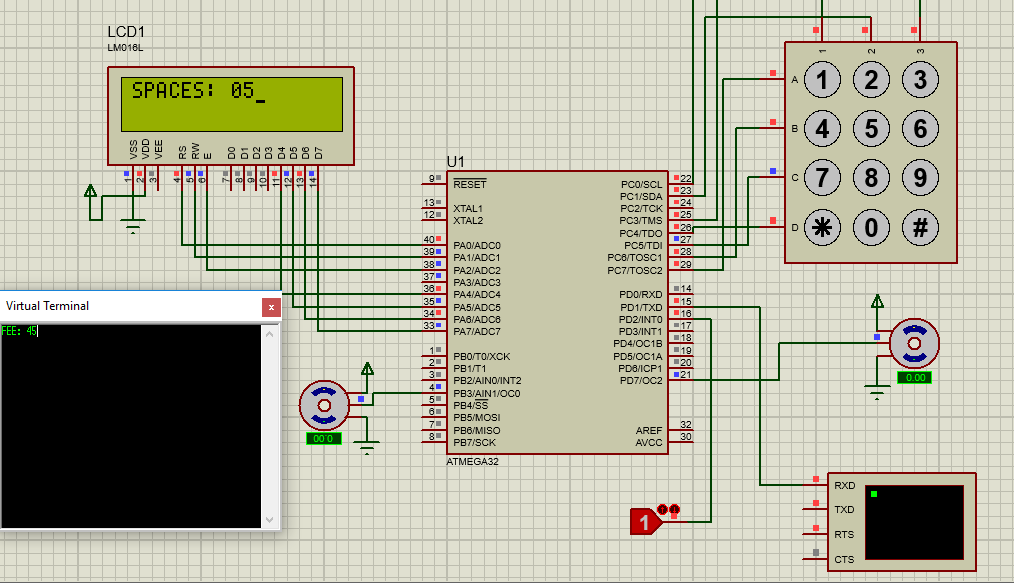
****

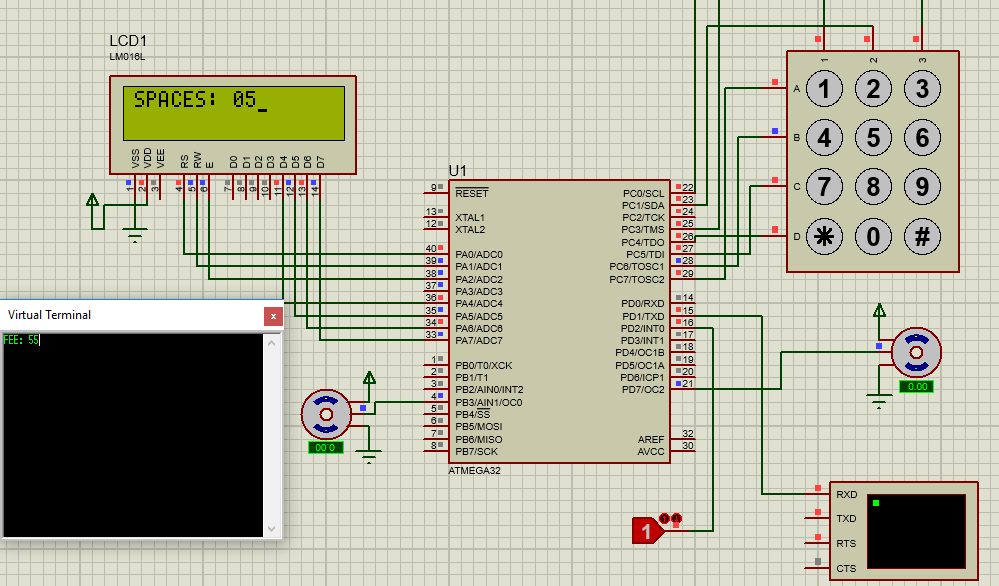
**Please note:** We have had to switch out ultrasonic sensors with IR sensors at parking spots due to the constraint on the number of ultrasonic sensors we can use. After interfacing ultrasonic sensor with AVR we found that in order to calculate distance we need to get pulse width from echo pin, this to our understanding is possible efficiently with input capture and external interrupts which are only a few. IR sensors can however be used at each port, allowing us a decent number of parking spaces. We also had to switch out beam break sensor at the entrance because we were unable to find it in Pakistan, we are using ultrasonic sensor as replacement.

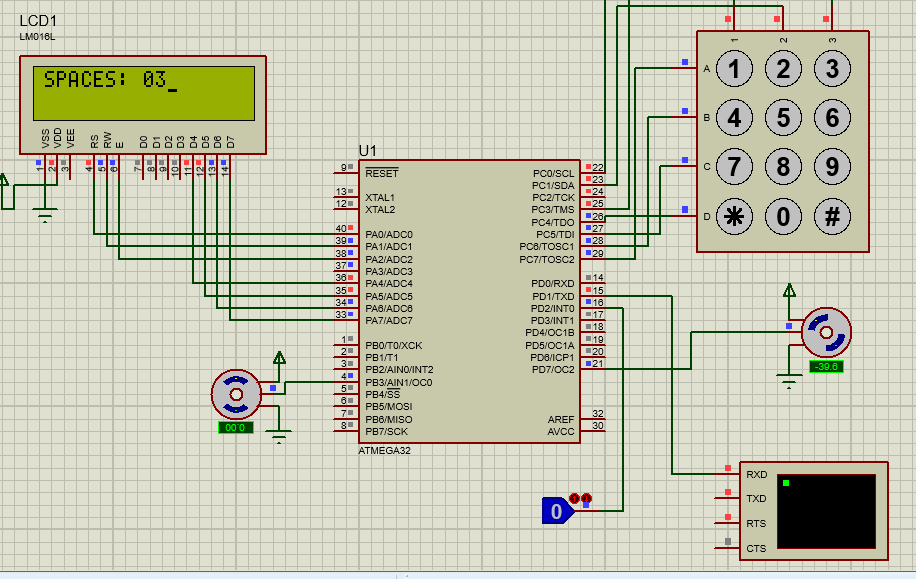
**Flow Chart**

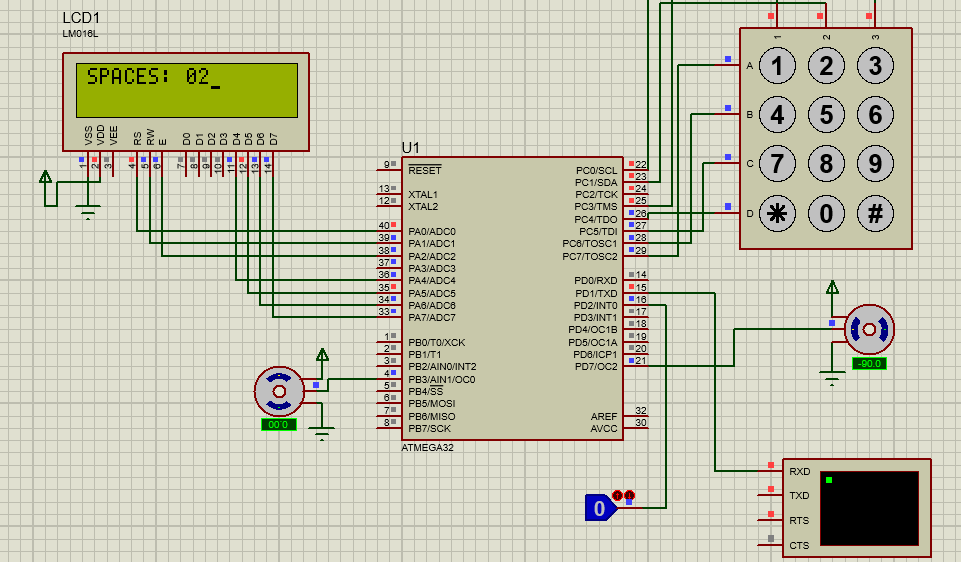
****

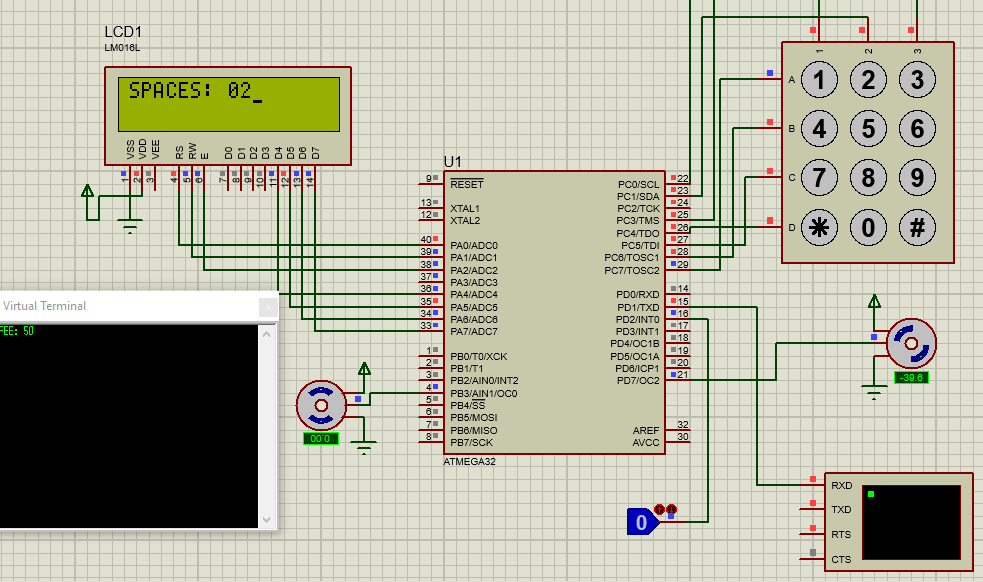
**Proteus simulation:**

****

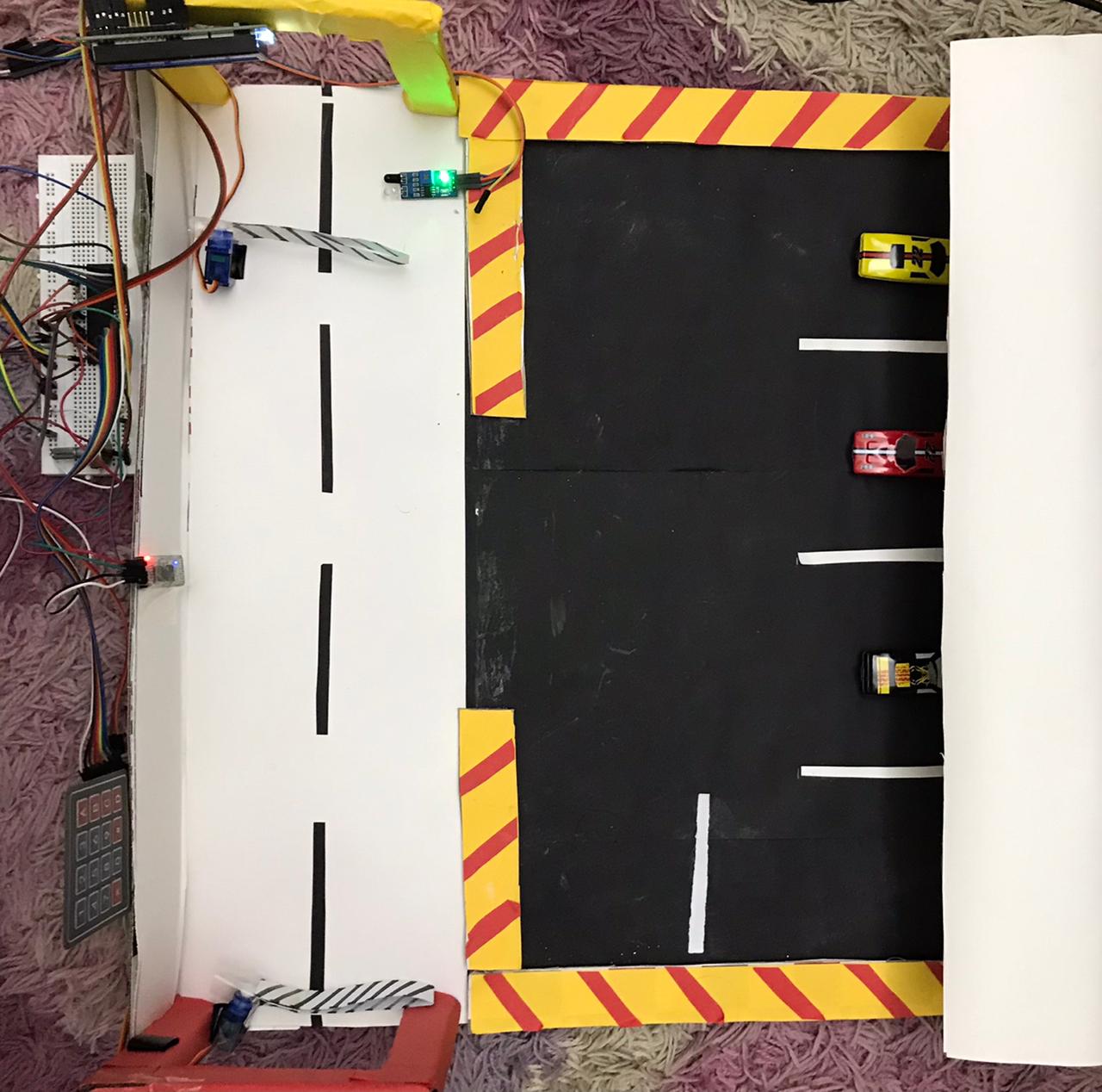
****

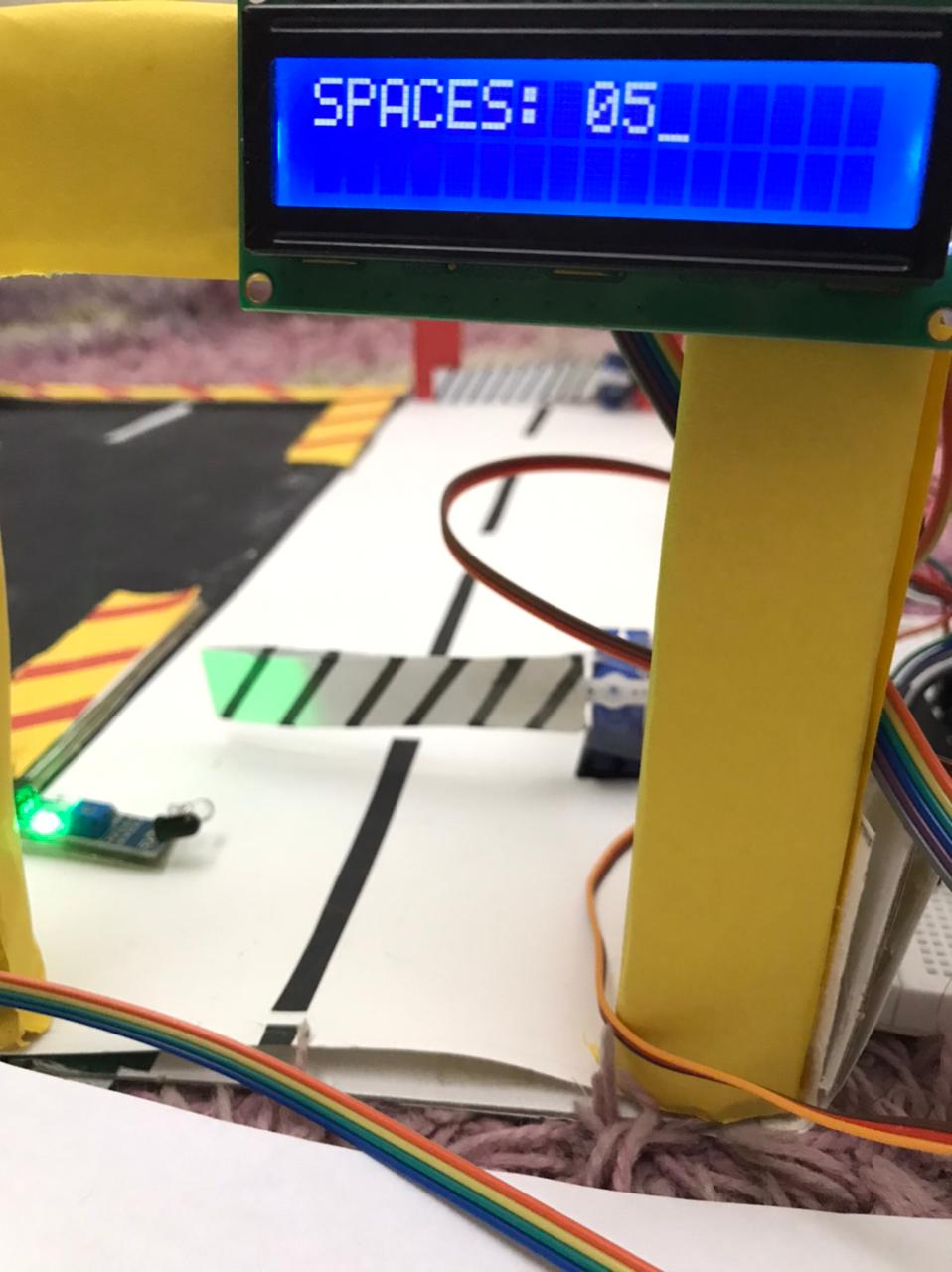
****

****

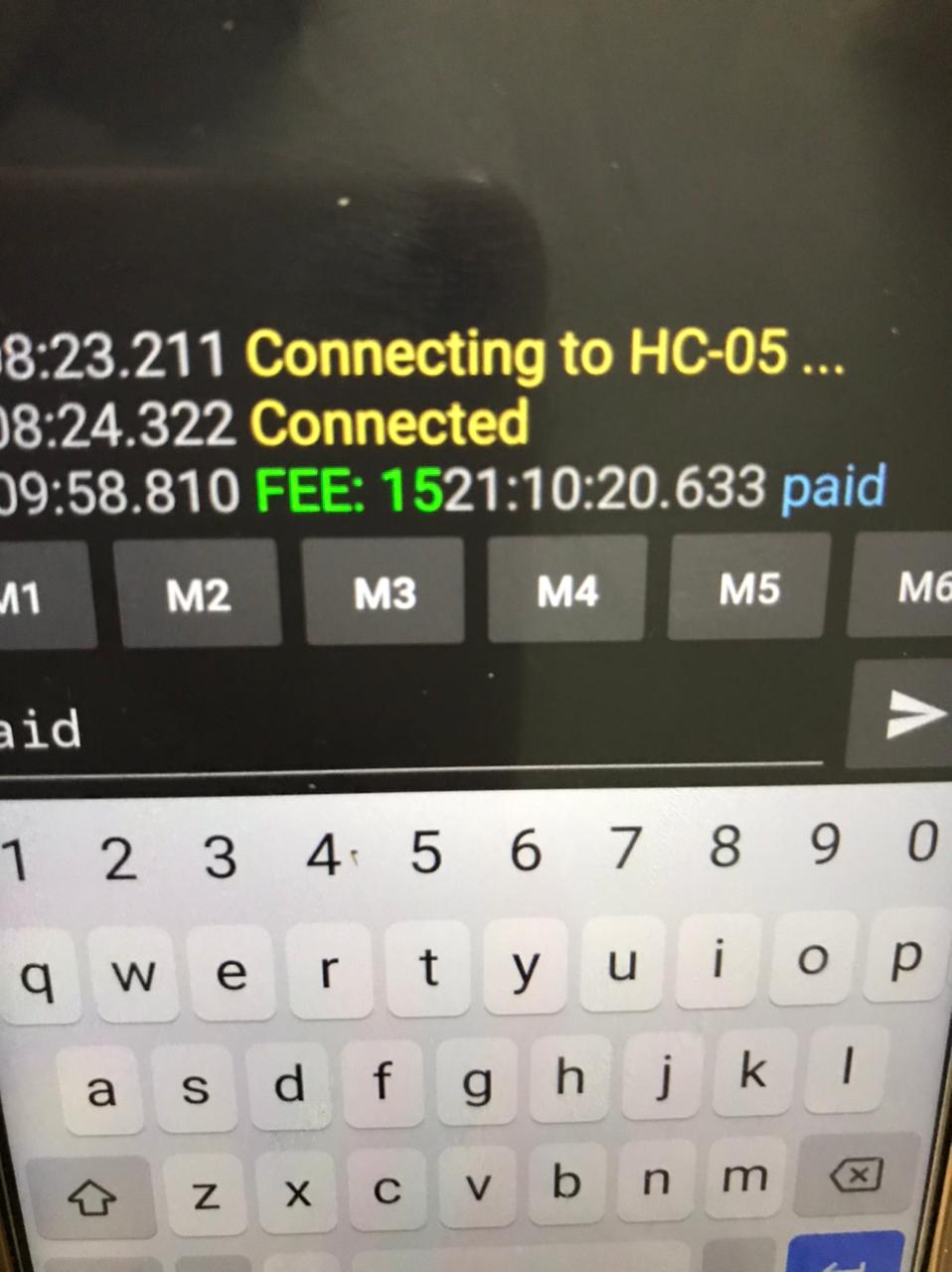
****

**Hardware implementation:**

****

****

****

****

**Results and findings:**

Over the preparation and execution of our project, we have gotten quite a few findings because of which the final product is slightly different to the proposed idea. We have had to leave out ultrasonic sensors, as the interrupt routine wasn’t functioning properly for us. We couldn’t use I2C interface for RTC as that port was occupied by keypad and alternate functions cannot be used with the port pins simultaneously. We didn’t have enough pins to interface two lcds, so alternatively we used Bluetooth to display the fee. Bluetooth served a second purpose of sending control signal for exit motor in accordance to fee payment status. Proteus simulation gave us opcode errors multiple times. These were resolved by fixing the overall loops and the addresses set by .ORG. In the process of burning code, we also bricked our MCU by setting the fuse bits incorrectly. We had to de-brick by supplying external clock to XTAL1 and resetting fuse bits. Interfacing all other components proved straightforward and simple and were achieved successfully.

**Code:**

MAIN:

.EQU P\_S=0X200

.EQU KEY\_PORT = PORTC

.EQU KEY\_PIN = PINC

.EQU KEY\_DDR = DDRC

.INCLUDE "M32DEF.INC"

LDI R16, HIGH(RAMEND)

OUT SPH,R16

LDI R16, LOW(RAMEND)

OUT SPL,R17 ;6B

LDI R16, (1<<U2X)

OUT UCSRA, R16

LDI R16, (1<<TXEN) |( 1<<RXEN) ; enable transmitter

OUT UCSRB, R16 ;and receiver

LDI R16, (1<<UCSZ1) |(1<<UCSZ0) | (1<<URSEL) ;8-bit data

OUT UCSRC, R16 ;no parity, 1 stop bit

LDI R16, 0x0C ; 9600 baud rate

OUT UBRRL, R16

LDI R16, 0XFF

OUT DDRA, R16

CBI DDRD, 2

SBI PORTD, 2

LDI R16, 5 ;TOTAL SPACES

STS P\_S, R16

LDI R16, 'p'

STS 0X201, R16

LDI R16, 'a'

STS 0X202, R16

LDI R16, 'i'

STS 0X203, R16

LDI R16, 'd'

STS 0X204, R16

SBI DDRD,7

SBI DDRB,3

CBI PORTA,2

LDI R20, 0xF0 //portc 0-3 input pins

OUT KEY\_DDR, R20

LDI R17, 8

WHILE:

CALL DELAY\_2

DEC R17

BRNE WHILE

LDI R16, 0X33

CALL CMDWRT

CALL DELAY\_2

LDI R16, 0X32

CALL CMDWRT

CALL DELAY\_2

LDI R16, 0X28

CALL CMDWRT

CALL DELAY\_2

LDI R16, 0X0E

CALL CMDWRT

LDI R16, 0X01

CALL CMDWRT

CALL DELAY\_2

LDI R16, 0X06

CALL CMDWRT

LDI ZH, HIGH(MSG<<1)

LDI ZL, LOW(MSG<<1)

LOOP: LPM R16,Z+

CPI R16, 0

BREQ OUT\_

CALL DATAWRT

RJMP LOOP

OUT\_:

HERE:

LDI R16, 0X88

CALL CMDWRT

LDS R16, P\_S

MOV R21, R16

CALL Convert

MOV R21, R25

SWAP R21

ANDI R21, 0X0F

MOV R16, R21

LDI R17, 48

ADD R16, R17

CALL DATAWRT

MOV R21, R25

ANDI R21, 0X0F

MOV R16, R21

ADD R16, R17

CALL DATAWRT

GroundAllRows:

LDI R20, 0x0F

OUT KEY\_PORT, R20

SBIS PIND,2

CALL OPEN

WAIT\_FOR\_KEY:

NOP

CALL WAIT15MS

IN R21,KEY\_PIN

ANDI R21,0x0F

CPI R21,0x0F

BREQ HERE

CALL KEYPAD

RJMP HERE

MSG: .DB "SPACES:",0

MSG2: .DB "LOT FULL!",0

KCODE0: .DB 10, 3, 2, 1 //col1

KCODE1: .DB 11, 6, 5, 4

KCODE2: .DB 12, 9, 8, 7

KCODE3: .DB 13, '#', 0, '\*'

BLU: .DB "FEE: ",0

OPEN:

LDS R16, P\_S

CPI R16, 0

BREQ LEAVE

CALL DELAYMOTOR

CALL SERVO1

RJMP OUT\_1

LEAVE:

LDI R16, 0XC0

CALL CMDWRT

LDI ZH, HIGH(MSG2<<1)

LDI ZL, LOW(MSG2<<1)

LOOP6: LPM R16,Z+

CPI R16, 0

BREQ OUT\_1

CALL DATAWRT

RJMP LOOP6

OUT\_1:

RET

CMDWRT:

MOV R20, R16

ANDI R20, 0XF0

IN R21, PORTA

ANDI R21, 0X0F

OR R21, R20

OUT PORTA, R21

CBI PORTA, 0

CBI PORTA, 1

SBI PORTA, 2

CALL SDELAY

CBI PORTA, 2

CALL DELAY\_100

MOV R20, R16

SWAP R20

ANDI R20, 0XF0

IN R21, PORTA

ANDI R21, 0X0F

OR R21, R20

OUT PORTA, R21

SBI PORTA, 2

CALL SDELAY

CBI PORTA, 2

CALL DELAY\_100

RET

DATAWRT:

MOV R20, R16

ANDI R20, 0XF0

IN R21, PORTA

ANDI R21, 0X0F

OR R21, R20

OUT PORTA, R21

SBI PORTA, 0

CBI PORTA, 1

SBI PORTA, 2

CALL SDELAY

CBI PORTA, 2

CALL DELAY\_100

MOV R20, R16

SWAP R20

ANDI R20, 0XF0

IN R21, PORTA

ANDI R21, 0X0F

OR R21, R20

OUT PORTA, R21

SBI PORTA, 2

CALL SDELAY

CBI PORTA, 2

CALL DELAY\_100

RET

DELAY\_100:

LDI R22, 50

LOOP2:

CALL SDELAY

DEC R22

BRNE LOOP2

RET

SDELAY:

NOP

NOP

RET

DELAY\_2:

LDI R23, 20

LOOP3: CALL DELAY\_100

DEC R23

BRNE LOOP3

RET

WAIT15MS:

LDI R24, 7

HOOP:

RCALL DELAY\_2

DEC R24

BRNE HOOP

RET

SERVO1:

rcall init90\_1

RCALL DELAYMOTOR

rcall init0\_1

RCALL DELAYMOTOR

;LDI R20, 0X00

;OUT TCCR2, R20

LDS R20, P\_S

CPI R20, 0

BREQ GO1

DEC R20

STS P\_S, R20

GO1:

RET

SERVO2:

RCALL DELAYMOTOR

rcall init90\_2

RCALL DELAYMOTOR

rcall init0\_2

RCALL DELAYMOTOR

;LDI R20, 0X00

;OUT TCCR0, R20

LDS R20, P\_S

CPI R20, 5

BREQ GO2

INC R20

STS P\_S, R20

GO2:

RET

init90\_1:

LDI R18, 13

OUT OCR2, R18

LDI R18,0

OUT TCNT2, R18

LDI R18, 0X6C

OUT TCCR2, R18

ret

init0\_1:

LDI R18, 19

OUT OCR2, R18

LDI R18,0

OUT TCNT2, R18

LDI R18, 0X6C

OUT TCCR2, R18

ret

init90\_2:

LDI R18, 13

OUT OCR0, R18

LDI R18,0

OUT TCNT0, R18

LDI R18, 0X6B

OUT TCCR0, R18

ret

init0\_2:

LDI R18, 19

OUT OCR0, R18

LDI R18,0

OUT TCNT0, R18

LDI R18, 0X6B

OUT TCCR0, R18

ret

init90:

LDI R18,13

OUT OCR2, R18

LDI R18,0

OUT TCNT2, R18

LDI R18, 0X6C

OUT TCCR2, R18

ret

init0:

LDI R18, 19

OUT OCR2, R18

LDI R18,0

OUT TCNT2, R18

LDI R18, 0X6C

OUT TCCR2, R18

ret

delaymotor:

ldi r18, high(-15625)

out tcnt1h, r18

ldi r18, high(-15625)

out tcnt1h, r18

ldi r16, 0x00

out tccr1A, r16

ldi r16, 0x03

out tccr1B, r16

AGAIN:IN R20,TIFR

SBRS R20,TOV1

RJMP AGAIN

LDI R20, 0x0

OUT TCCR1B,R20 ;stop TimerO

Ldi R20, (1<<TOV1)

OUT TIFR, R20

ret

DELAY:

ldi r18, 0

out tcnt0, r18

ldi r16, 0x05

out tccr0, r16

AGAIN2:IN R20,TIFR

SBRS R20,TOV0

RJMP AGAIN2

LDI R20, 0x0

OUT TCCR0,R20 ;stop TimerO

Ldi R20, (1<<TOV0)

OUT TIFR, R20

ret

Convert:CLR R25

NEXT:SUBI R21, 10 ; subtract decimal 10 from number

BRCS REXIT ; if number < 10 then exit

SUBI R25,-0X10 ; else 10 to BCD number

RJMP NEXT ; repeat subtraction

REXIT:SUBI R21, -10 ; undo last subtraction

ADD R25, R21 ; add unit to BCD

RET

KEYPAD:

LDI R21, 0b01111111

OUT KEY\_PORT,R21

NOP

IN R21,KEY\_PIN

ANDI R21,0x0F

CPI R21,0x0F

BRNE COL1

LDI R21, 0b10111111

OUT KEY\_PORT, R21

NOP

IN R21, KEY\_PIN

ANDI R21,0x0F

CPI R21,0x0F

BRNE COL2

LDI R21, 0b11011111

OUT KEY\_PORT, R21

NOP

IN R21, KEY\_PIN

ANDI R21,0x0F

CPI R21,0x0F

BRNE COL3

LDI R21, 0b11101111

OUT KEY\_PORT, R21

NOP

IN R21, KEY\_PIN

ANDI R21,0x0F

CPI R21,0x0F

BRNE COL4

RJMP EXITKEY

COL1:

LDI R30, LOW(KCODE0<<1)

LDI R31, HIGH(KCODE0<<1)

RJMP Find

COL2:

LDI R30, LOW(KCODE1<<1)

LDI R31, HIGH(KCODE1<<1)

RJMP Find

COL3:

LDI R30, LOW(KCODE2<<1)

LDI R31, HIGH(KCODE2<<1)

RJMP Find

COL4:

LDI R30, LOW(KCODE3<<1)

LDI R31, HIGH(KCODE3<<1)

Find:

LSR R21

BRCC Match

LPM R20, Z+

RJMP Find

MATCH:

LPM R20, Z

MOV R16,R20 ;HOURS FROM KEYPAD

LDI R21, 0

A:LDI R17, 5

ADD R21, R17 ;CALCULATE FEE

DEC R20 ;HOURS FROM KEYPAD

BRNE A

CALL Convert

MOV R21, R25

SWAP R21

ANDI R21, 0X0F

MOV R16, R21

LDI R17, 48

ADD R16, R17

MOV R5, R16

MOV R21, R25

ANDI R21, 0X0F

MOV R16, R21

ADD R16, R17

MOV R6, R16

LDI ZH, HIGH(BLU<<1)

LDI ZL, LOW(BLU<<1)

LOOP5: LPM R16,Z+

CPI R16, 0

BREQ T

CALL TRNSMT

RJMP LOOP5

T:

MOV R16, R5

CALL TRNSMT

MOV R16, R6

CALL TRNSMT

LDI YL, LOW(0X205)

LDI YH, HIGH(0X205)

LDI R18, 4

GO:

SBIS UCSRA,RXC

RJMP GO

IN R16,UDR

ST Y+, R16

LDI R17, '\n'

CP R16, R17

BRNE GO

LDI XL, LOW(0X201)

LDI XH, HIGH(0X201)

LDI YL, LOW(0X205)

LDI YH, HIGH(0X205)

REDO: LD R16, X+

LD R17, Y+

CP R16, R17

BRNE EXITKEY

DEC R18

BRNE REDO

LDS R16, P\_S

CPI R16, 0

BRNE SERV2

LDI R16, 0X01

CALL CMDWRT

CALL DELAY\_2

LDI R16, 0X06

CALL CMDWRT

LDI ZH, HIGH(MSG<<1)

LDI ZL, LOW(MSG<<1)

LOO: LPM R16,Z+

CPI R16, 0

BREQ SERV2

CALL DATAWRT

RJMP LOO

SERV2: CALL SERVO2

EXITKEY:

RET

TRNSMT:

SBIS UCSRA,UDRE

RJMP TRNSMT

OUT UDR,R16

RET

RJMP MAIN