Project #1: Car waiting meter

Name: Mohamed Ossama Ahmed Fathi Aboelnaga

Id: 7861

Group 2 sec 1

prof. Dr Hania Farag

Introduction:

The purpose of this project is to develop a car waiting meter system for a car parking facility. The system allows drivers to pay a certain amount of money, which grants them a specific amount of time to park their car. The project implements the meter using a microcontroller and includes a feature where a red LED is activated when the time expires. This report provides a detailed overview of the project, including the requirements, implementation, and testing.

Project Requirements:

The project requirements were as follows:

Develop a car waiting meter system for a car parking facility.

Accept an input for money paid by the driver.

Determine the corresponding parking time based on the payment.

Activate a red LED when the time expires.

Implement payment options: 1 pound for 5 seconds, 5 pounds for 30 seconds, and 10 pounds for 60 seconds.

Solution Overview:

The solution involves using a microcontroller to implement the car waiting meter system. The program prompts the user to enter the amount of money paid and then calculates the parking time based on the input. The system uses a loop with specific delay values to simulate the parking time. Once the time expires, a red LED is turned on to indicate that the parking period has ended.

Implementation Details:

The implementation consists of the following components:

Data segment: Contains the strings used for displaying the price list and messages.

Code segment: Includes the main procedure for the program execution.

Input and Validation: The program prompts the user to enter the amount of money and validates the input.

Time Delay Calculation: The program determines the appropriate time delay based on the input value.

Time Expiration: After the specified time delay, the system displays an expiration message and activates the red LED.

Error Handling: The program handles invalid input by displaying an error message and allowing the user to retry.

Testing:

To ensure the correctness of the implemented system, thorough testing was performed. The following test cases were executed:

Test Case 1: Input of 1 pound (expected output: 5 seconds delay).

Test Case 2: Input of 5 pounds (expected output: 30 seconds delay).

Test Case 3: Input of 10 pounds (expected output: 60 seconds delay).

Test Case 4: Input of 0 pounds (expected output: program exit).

Test Case 5: Input of an invalid value (expected output: error message displayed, retry option).

All test cases produced the expected outputs, validating the accuracy of the implemented solution.

Conclusion:

The project successfully implemented a car waiting meter system using a microcontroller. The solution effectively calculates the parking time based on the amount of money paid and activates a red LED upon time expiration. The implemented system fulfills all the specified requirements and has been thoroughly tested to ensure its reliability.

Future Enhancements:

Although the current implementation meets the project requirements, there are potential areas for future enhancements, such as:

Integration with a display module to show the remaining parking time.

Addition of multiple payment options for increased flexibility.

Implementation of a billing system to keep track of payments and generate reports.

Integration with a mobile application for remote payment and monitoring.

By incorporating these enhancements, the car waiting meter system can be further improved to provide an enhanced user experience and additional functionalities.

Overall, the project demonstrates the successful implementation of a car waiting meter system and serves as a foundation for further development and improvements in the future.

code:

```
.data
STRING1 DB   '******Price list****** $'
STRING2 DB  DB 0AH, 0DH, '-->1 pound for 5 seconds. $'
STRING3 DB  DB 0AH, 0DH, '-->5 pounds for 30 seconds. $'
STRING4 DB  DB 0AH, 0DH, '-->10 pounds for 60 seconds. $'
MoneyMessage db db 0AH, 0DH, 'Enter the amount of money you will pay (0 to exit): $'
ExpireMessage db db 0AH, 0DH, 0AH, 0DH, 'Time expired and LED turned red.$'
invalidMessage db db 0AH, 0DH,0AH, 0DH, "undefined value,please enter valid value $"

.code

 main proc

 mov ax, @data
 mov ds, ax
 ; load address of the strings
 LEA DX,STRING1
 MOV AH,09H
 INT 21H
 LEA DX,STRING2
 MOV AH,09H
 INT 21H
 LEA DX,STRING3
 MOV AH,09H
 INT 21H
 LEA DX,STRING4
 MOV AH,09H
 INT 21H
```

```
main2:
  mov ax, @data
  mov ds, ax

  lea dx, MoneyMessage
  mov ah, 09h       ; Display message
  int 21h

  mov dl,10 ;intialize dl by 10
  mov bl,0  ;intialize bl by 0

; Read input
scanNumber:

     mov ah, 01h
     int 21h

     cmp al, 13   ; if "ENTER"
     je  Return

     mov ah, 0
     sub al, 48   ; ASCII

     mov cl, al
     mov al, bl   ; bl contains the previous value

     mul dl       ; multiply the previous value with 10

     add al, cl   ; previous * 10 + new value
     mov bl, al
```

```
    jmp scanNumber

Return:

  ; Check value and choose delay
  cmp bl, 0
  je exit

  cmp bl, 1
  je delay_5sec

  cmp bl, 5
  je delay_30sec

  cmp bl, 0Ah
  je delay_60sec

  ; Invalid value
  jmp invalid

delay_5sec:
  mov cx, 2600
  loop $
  jmp expired

delay_30sec:
  mov cx, 16500
  loop $
  jmp expired

delay_60sec:
```

```
  mov cx, 33500
  loop $
  jmp expired

expired:
  ; Display ExpireMessage
  mov ah, 09h
  lea dx, ExpireMessage
  int 21h

  ; Turn on LED (Assume port 199 control LED)
  mov ax, 1
  out 199, ax

  jmp main2      ; Repeat the process

invalid:
  ; Display error message for invalid value
  mov ah, 09h
  lea dx, invalidMessage
  int 21h

  jmp main2      ; Repeat process

exit:
  mov ah, 4Ch   ; Exit program
  int 21h

main endp
end main
```

extra Enhancements for bonus

check the entered value from user that is integer and not letters

Increasing flexibility in payments by accepting any integer value as 2,6,11,17
in case value 2$ the time will be 5s(1$)+5s(1$)=10 sec
in case value 6$ the time will be 30s(5$)+5s(1$)=35 sec
in case value 11$ the time will be 60s(10$)+5s(1$)=65sec
in case value 17$ the time will be 60s(10$)+30s(5$)+5s(1$)+5s(1$)=100sec

in the code I edited this parts:

```
scanNumber:
    mov ah, 01h
    int 21h

    cmp al, 13   ; if "ENTER"
    je  Return

    cmp al, '0'   ; Check if the input is a digit
    jl invalid
    cmp al, '9'
    jg invalid

    sub al, '0'   ; ASCII to decimal conversion

    mov cl, al
    mov al, bl   ; bl contains the previous value

    mul dl       ; multiply the previous value with 10

    add al, cl   ; previous * 10 + new value
    mov bl, al
```

```
    jmp scanNumber


Return:

  ; Check value and choose delay
  cmp bl, 0
  je exit


  cmp bl, 5
  jl delay_5secN

  cmp bl, 10
  jl delay_30secN

  cmp bl, 0Ah
  jge delay_60secN

  ; Invalid value
  jmp invalid


delay_5secN:
  mov cx, 2700
  loop $
  dec bl
  cmp bl,0
  jg  delay_5secN
  jmp expired
```

```
delay_30secN:
  mov cx, 16000
  loop $
  sub bl,5
  cmp bl,0
  jg  delay_5secN
  jmp expired

delay_60secN:
  mov cx, 32500
  loop $
  sub bl,10
  cmp bl,5
  jge delay_30secN
  cmp bl,0
  jg delay_5secN
  jmp expired
```