Title: ATM Management System

J Component Project

Group members

Saima Shahnawaz 19BCE0384

Aniket Singh 19BCE2103

Tanmay Srivastava 19BCE0416

Sanskar Tiwari 19BCE0955

Ayush Thakur 19BCE0885

Slot:E1

B.Tech. Computer Science and Engineering

School of Computer Science and Engineering

Vellore Institute of Technology

Vellore, Tamil Nadu

November, 2021

Abstract

Information technology is a rapidly growing sector nowadays. Daily new inventions are taking place in the market. In the financial sector especially the banking sector there are so many new technologies taking place in financial operations.ATM is an important invention for the banking sector. The wonders of modern technology have made it possible for bank customers to interact with an electronic banking facility such as Automated Teller Machine (ATM) rather than with a human Being for cash transactions. Electronic banking is one of the newest services offered by almost all banks to their customers. Electronic banking involves amongst others, Automated Teller Machine (ATM), Point-Of-Sale (POS), and Telephone banking and so on. The high cost of setting up and operating full-service branch offices has led recently to a sharp expansion in limited-service facilities such as Automated Teller Machines (ATMs). The use of ATMs has grown rapidly in popularity because of its low bank transactions costs and customers convenience which has made it a basic element of today’s financial service offering.

Introduction:

Clients of a bank can access their accounts with an Automated Teller Machine (ATM) without having to visit the bank. This is only possible if the application is built utilising online ideas.

When the product is deployed, the user will be able to see all of the information and services supplied by the ATM after entering the required options and arguments. The programme also offers services such as cheque requests, cash deposits, and other advanced user needs. The information is saved in a database and accessed as needed. ATM machine hardware is required for the implementation to work, however identical simulated conditions can also be used to successfully use the generated product.

In this project, we are going to implement the ATM management system. Due to the basic necessity of atm services, we need to be updated, which is very necessary as the atm services are the basic proprity of every citizen. Due to the increase in the number of banks in both the regional and private sectors, we can see that all banks have their own ATM systems with different inbuilt mechanisms that highly differ from each other, as well as, according to where they are located in the region, they highly differ in language, facilities, and services. ATMs are widely used everywhere, so in this project, we will concentrate on how we will try to build an ATM management system using an 8086 emulator, which provides all the features with the basic modules and instructions available while providing the best result. The services provided by the 8086 emulator include functionality such as balance enquiry, withdrawal of money, and money transfer.

Literature Review:

Application for Emu8086 and Proteus in Microcomputer Principle Teaching Qing WANG* and Zhong-nian LI

Combined with the teaching of Microcomputer Principle and interface technology in electrical engineering and automation, the authors proposes a new teaching method. Firstly, the authors analyzed the teaching and learning situation of Microcomputer Principle and interface technology. Secondly, in the theory and practice teaching process, the authors put forward a new designing scheme that through the Emu8086 and Proteus software for assembly program writing, hardware circuit design and simulation design. The simulation example of frequency divider circuit is constructed by 8086 and 8253A.The experimental results show that the constructed frequency divider circuit can be realized by 8086 simulation. Finally, the authors summarizes the teaching methods of virtual simulation for Emu8086 and Proteus software, and obtains very good teaching results. In this paper, this innovation mode (that is, through the vivid simulation effect) arouse the student's study enthusiasm and provides a new method for embedded curriculum teaching reform.

Effects of Automated Teller Machine on the Performance of Nigerian Banks

American Journal of Applied Mathematics and Statistics 2(1):40-46

DOI:10.12691/ajams-2-1-7

Jegede C.A.

This study investigates the effects of ATM on the performance of Nigerian banks. Available studies have concentrated on the significant dimensions of ATM (automated teller machine) service quality and its effect on customer satisfaction with a bias against ATM producers. The study is motivated by the astronomical challenges confronting the proliferation of ATM infrastructure and attendant financial loss to banks which are often under-reported. Also, there are serious debate on the relevance of ATM technology as most countries in the world are moving away from the virus technology to the more secured chip cards free of credit and debit frauds. Questionnaire was used to collect the data from a convenience sample of 125 employees of five selected banks in Lagos State with interswitch network. Therefore, data collected through the questionnaire were analyzed statistically by using the Software Package for Social Science (SPSS Version 20.0 for Student Version) and chi-square technique. The results indicate that less than the benefits, the deployment of ATMs terminals have averagely improved the performance of Nigerian banks because of the alarming rate of ATM fraud. Similarly, ATM service quality is less correlated to security and privacy of users and providers.The conclusion therefore is that banks should strive to increase their security layers to subvert the tricks of web scammers, limit the amount which customers may be allowed to withdraw at a time and provide electronic alerts to customers' phone for all transactions carried out on their bank accounts through ATMs and the provisions of extra security layer that can prevent third party to make use someone else's ATM card for unauthorized withdrawals electronically.

https://www.atmmarketplace.com/articles/cash-matters-now-more-than-ever-in-the-time-of-coronavirus/

The ATMIA, along with several groups and organizations within the ATM industry, came together in Brussels to support the use of currency in the worldwide marketplace. The result of this effort was a joint statement to banks, industry, government and the media, outlining their position. ATM Marketplace is printing this statement in its entirety.

https://www.deccanherald.com/business/are-contactless-atm-transactions-a-solution-during-the-covid-19-pandemic-862728.html

ATM Machine System in C++

https://www.engineersgarage.com/atm-machine/

In this project, the C++ language was selected to carry out all the functionality of a stansard ATM machine. The user can check the amount present in the account and the functionalities are provided to him, such as withdrawing and depositing. For security purposes, the ATM card has a pin code and a password authorization. The logic is built on two functions: main() and printstar() which are embedded in the code. The main() function act as a core of the system where as printstar() is for displaying the detiails required.

Methodology:

Software Used-

- Emu8086: An emulator for the 8086 microprocessor.
  This is used for running assembly language codes.

Functions-

We'll execute the assembly language code on an 8086 emulator and create several features that the ATM system offers.

First, we need to make our ATM management system multi-user, thus we need to get the input of a lot of people.

The bank will generate a userid for the user.

Then we'll take the user's password as input to verify that it's a valid password and if you're an authorised user or not.

Main functions that we will include in our program are as follows:

Userid:

The user will be asked to provide their bank-generated user id, and if the user id matches the user's account, pin validation will be used for further verification.

PIN Validation:

Users will be requested to enter a PIN before being granted access to further functions if their PIN is validated.

Balance enquiry:

In this function, users can get all his account details and balance in his account.

Withdraw of money:

The user can withdraw money from his bank account, and the system will also inquire how much money the user wishes to remove from his account, as well as display the account's final balance after the money has been withdrawn.

<u>Transfering of the money:</u>

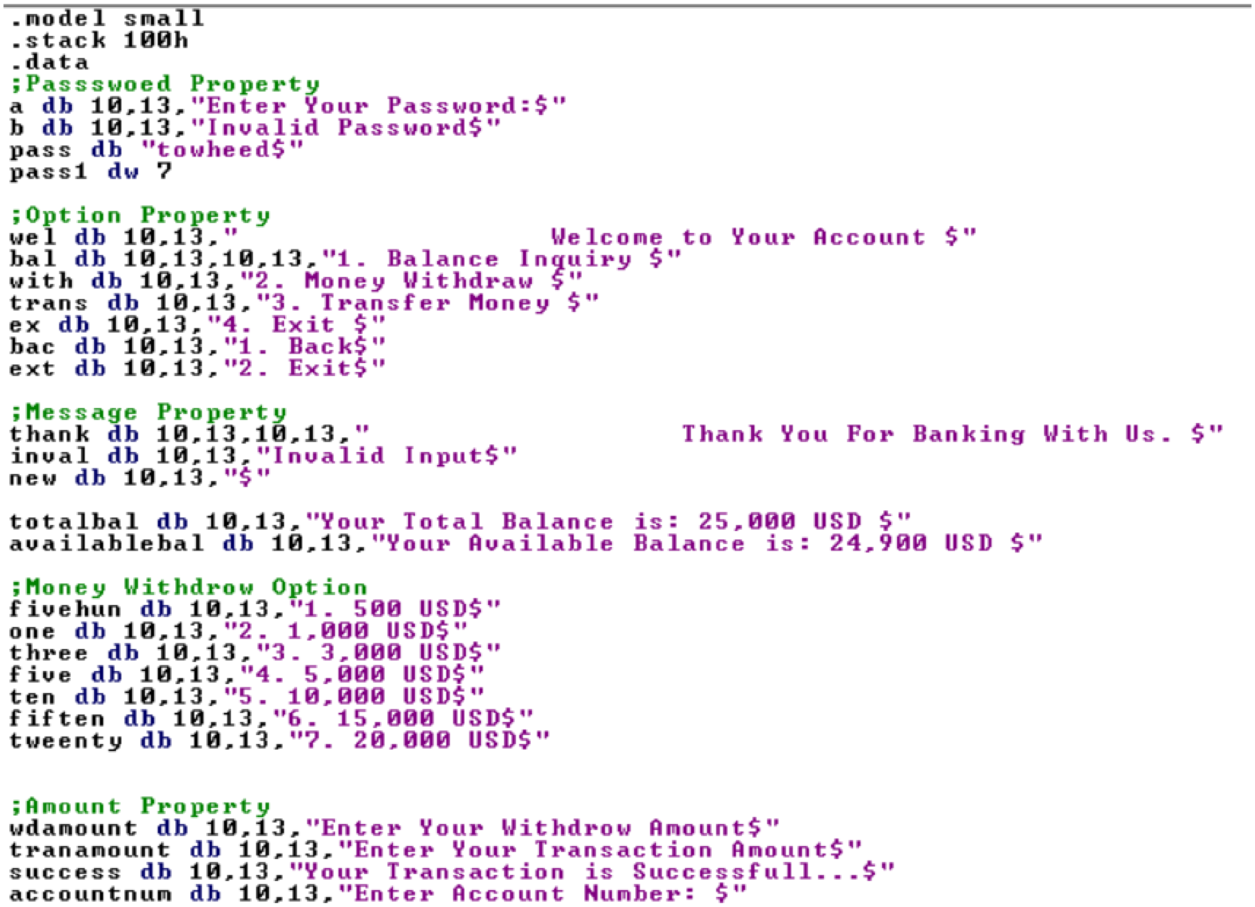
The user can send money to other people's accounts. It will request the user's account number as input, transfer money to the other account, and display the user's final balance and the amount transferred to the other account.

<u>Exit:</u>

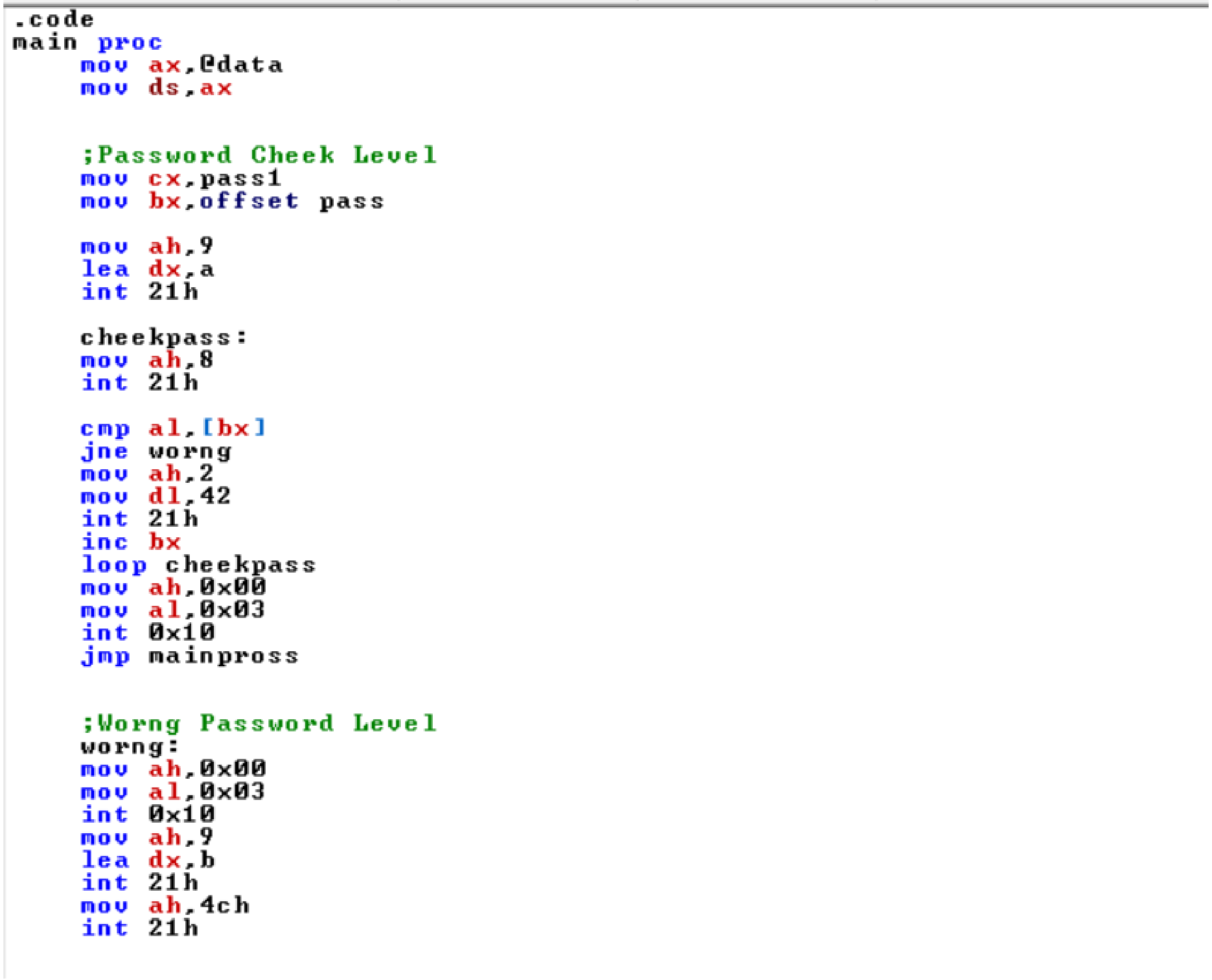
After using various services of the system user can exit from the system by using this function.

Code:

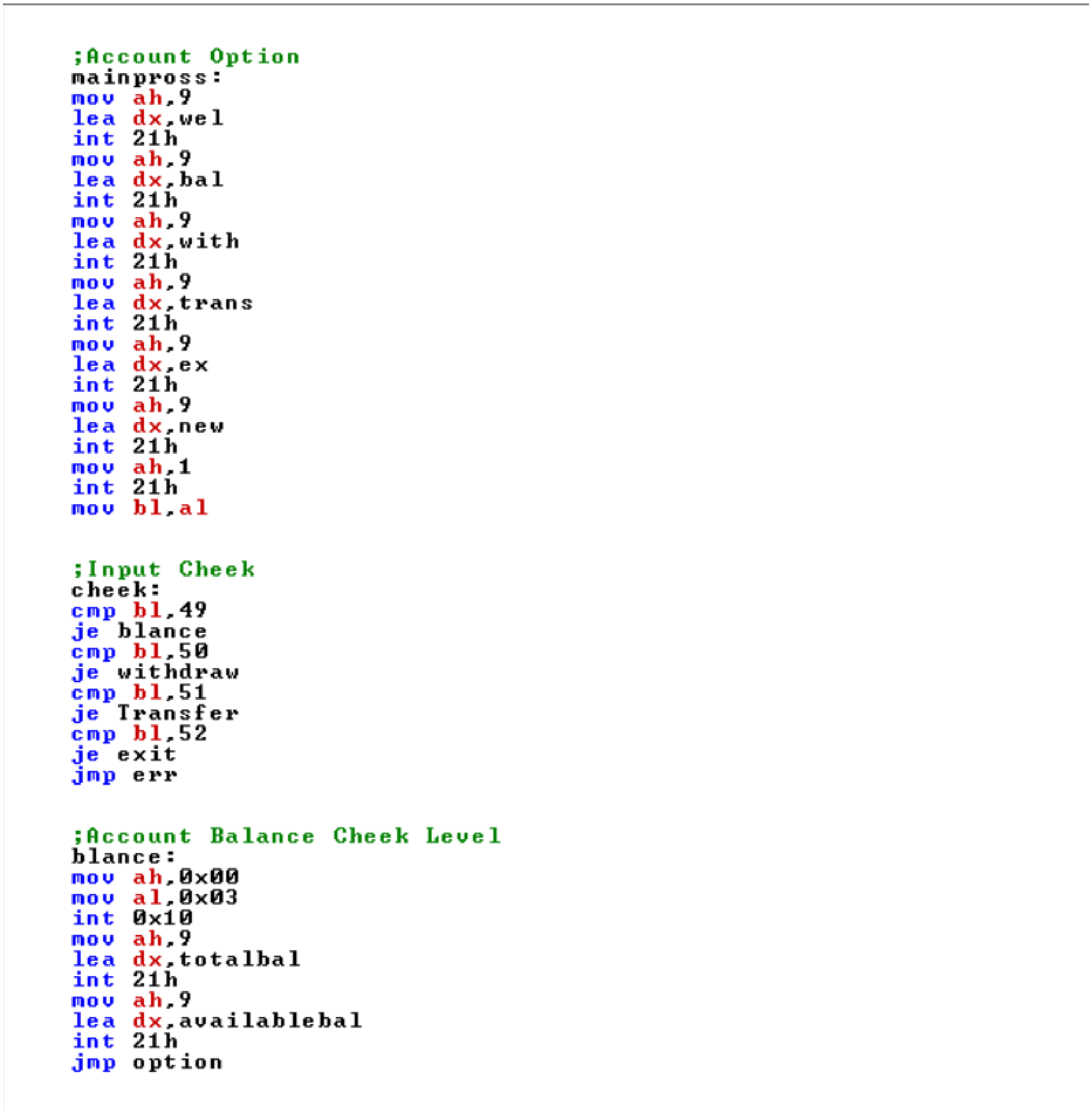
User Interface



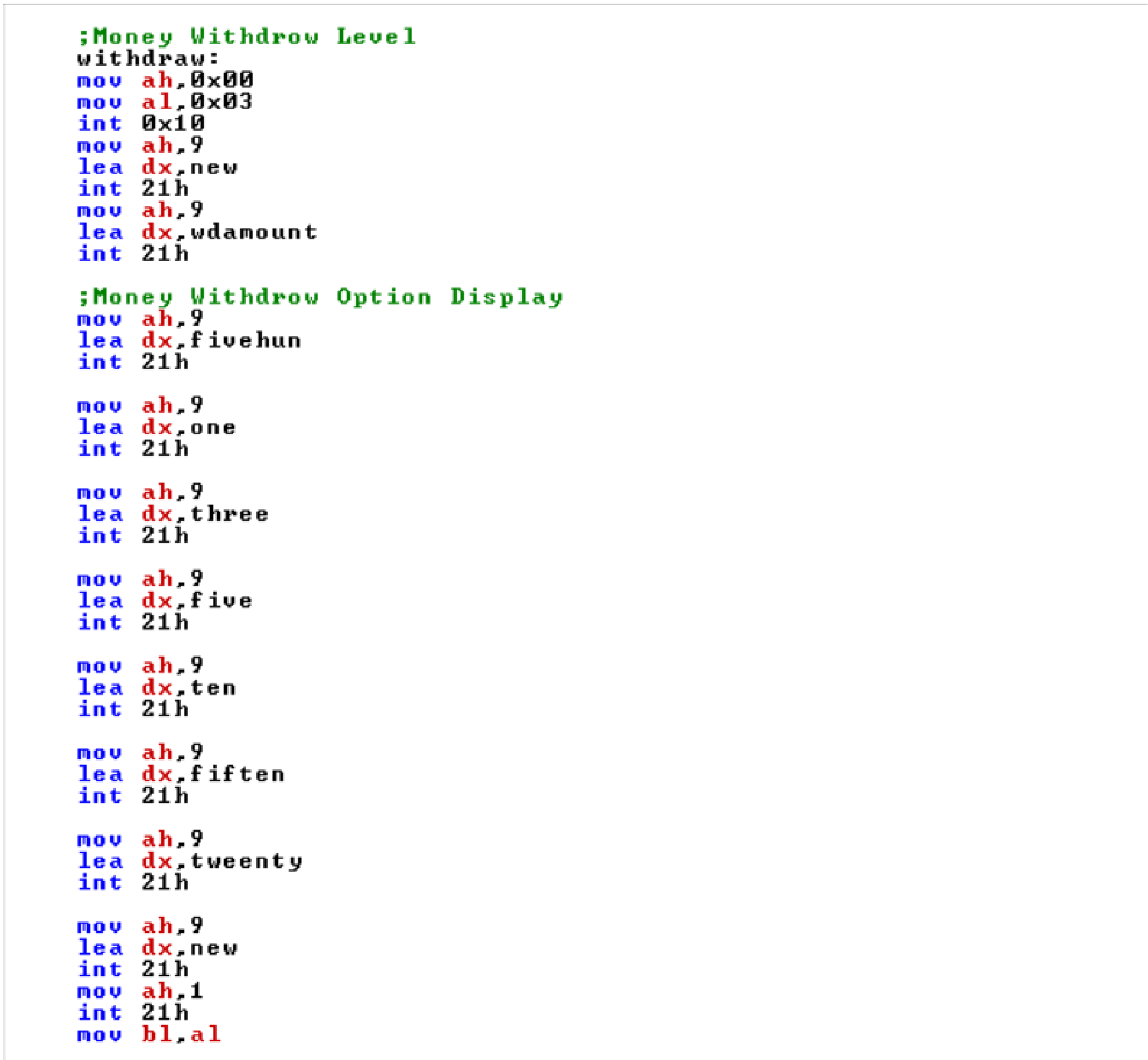
```
.model small
.stack 100h
.data
;Passswoed Property
a db 10,13,"Enter Your Password:$"
b db 10,13,"Invalid Password$"
pass db "towheed$"
pass1 dw 7

;Option Property
wel db 10,13,"                   Welcome to Your Account $"
bal db 10,13,10,13,"1. Balance Inquiry $"
with db 10,13,"2. Money Withdraw $"
trans db 10,13,"3. Transfer Money $"
ex db 10,13,"4. Exit $"
bac db 10,13,"1. Back$"
ext db 10,13,"2. Exit$"

;Message Property
thank db 10,13,10,13,"                   Thank You For Banking With Us. $"
inval db 10,13,"Invalid Input$"
new db 10,13,"$"

totalbal db 10,13,"Your Total Balance is: 25,000 USD $"
availablebal db 10,13,"Your Available Balance is: 24,900 USD $"

;Money Withdrow Option
fivehun db 10,13,"1. 500 USD$"
one db 10,13,"2. 1,000 USD$"
three db 10,13,"3. 3,000 USD$"
five db 10,13,"4. 5,000 USD$"
ten db 10,13,"5. 10,000 USD$"
fiften db 10,13,"6. 15,000 USD$"
tweenty db 10,13,"7. 20,000 USD$"


;Amount Property
wdamount db 10,13,"Enter Your Withdrow Amount$"
tranamount db 10,13,"Enter Your Transaction Amount$"
success db 10,13,"Your Transaction is Successfull...$"
accountnum db 10,13,"Enter Account Number: $"
```



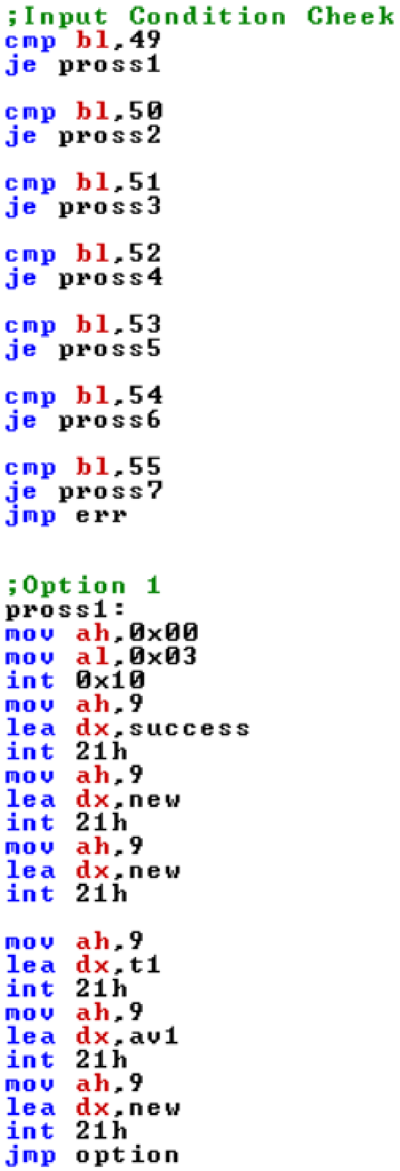
```
.code
main proc
     mov ax,@data
     mov ds,ax


     ;Password Cheek Level
     mov cx,pass1
     mov bx,offset pass

     mov ah,9
     lea dx,a
     int 21h

     cheekpass:
     mov ah,8
     int 21h

     cmp al,[bx]
     jne worng
     mov ah,2
     mov dl,42
     int 21h
     inc bx
     loop cheekpass
     mov ah,0x00
     mov al,0x03
     int 0x10
     jmp mainpross


     ;Worng Password Level
     worng:
     mov ah,0x00
     mov al,0x03
     int 0x10
     mov ah,9
     lea dx,b
     int 21h
     mov ah,4ch
     int 21h
```



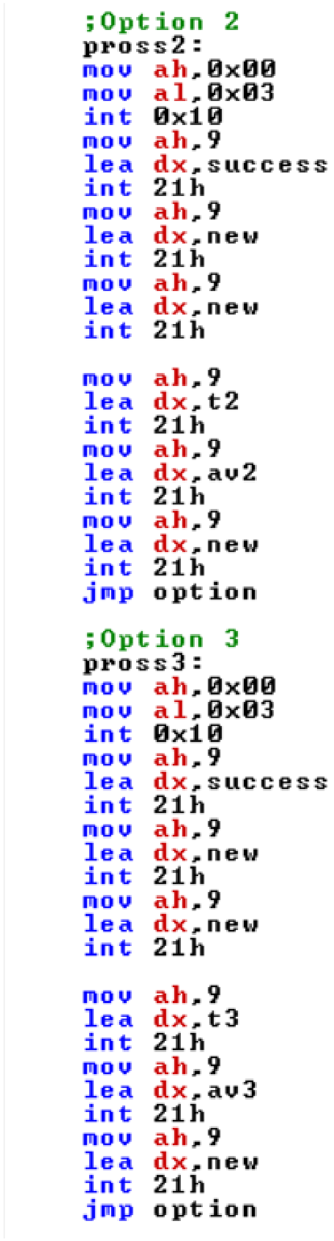
```
;Account Option
mainpross:
mov ah,9
lea dx,wel
int 21h
mov ah,9
lea dx,bal
int 21h
mov ah,9
lea dx,with
int 21h
mov ah,9
lea dx,trans
int 21h
mov ah,9
lea dx,ex
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
mov bl,al


;Input Cheek
cheek:
cmp bl,49
je blance
cmp bl,50
je withdraw
cmp bl,51
je Transfer
cmp bl,52
je exit
jmp err


;Account Balance Cheek Level
blance:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,totalbal
int 21h
mov ah,9
lea dx,availablebal
int 21h
jmp option
```



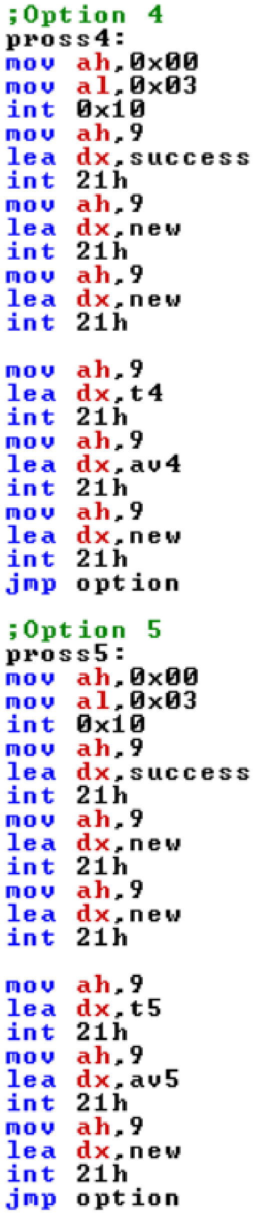
```
;Money Withdrow Level
withdraw:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,wdamount
int 21h

;Money Withdrow Option Display
mov ah,9
lea dx,fivehun
int 21h

mov ah,9
lea dx,one
int 21h

mov ah,9
lea dx,three
int 21h

mov ah,9
lea dx,five
int 21h

mov ah,9
lea dx,ten
int 21h

mov ah,9
lea dx,fiften
int 21h

mov ah,9
lea dx,tweenty
int 21h

mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
mov bl,al
```



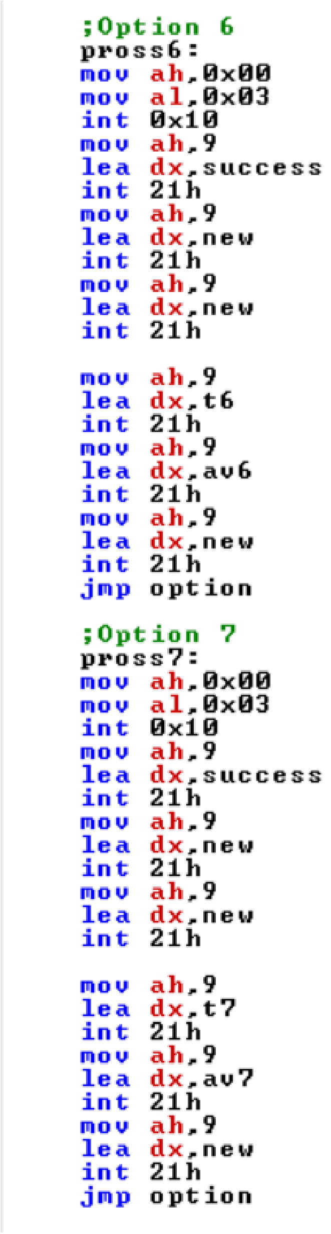
```
;Input Condition Cheek
cmp bl,49
je pross1

cmp bl,50
je pross2

cmp bl,51
je pross3

cmp bl,52
je pross4

cmp bl,53
je pross5

cmp bl,54
je pross6

cmp bl,55
je pross7
jmp err


;Option 1
pross1:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,success
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,new
int 21h

mov ah,9
lea dx,t1
int 21h
mov ah,9
lea dx,av1
int 21h
mov ah,9
lea dx,new
int 21h
jmp option
```



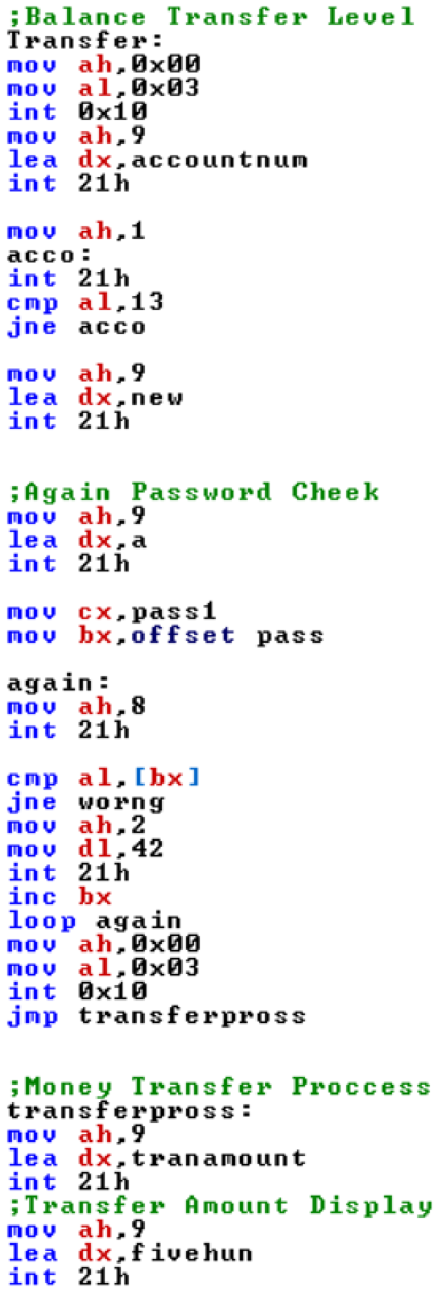
```
;Option 2
pross2:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,success
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,new
int 21h

mov ah,9
lea dx,t2
int 21h
mov ah,9
lea dx,av2
int 21h
mov ah,9
lea dx,new
int 21h
jmp option

;Option 3
pross3:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,success
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,new
int 21h

mov ah,9
lea dx,t3
int 21h
mov ah,9
lea dx,av3
int 21h
mov ah,9
lea dx,new
int 21h
jmp option
```



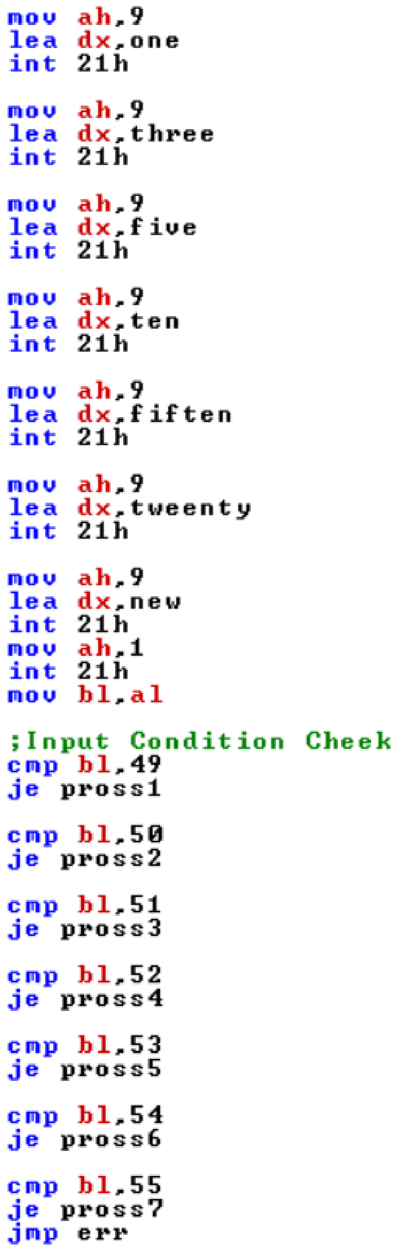
```
;Option 4
pross4:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,success
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,new
int 21h

mov ah,9
lea dx,t4
int 21h
mov ah,9
lea dx,av4
int 21h
mov ah,9
lea dx,new
int 21h
jmp option

;Option 5
pross5:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,success
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,new
int 21h

mov ah,9
lea dx,t5
int 21h
mov ah,9
lea dx,av5
int 21h
mov ah,9
lea dx,new
int 21h
jmp option
```



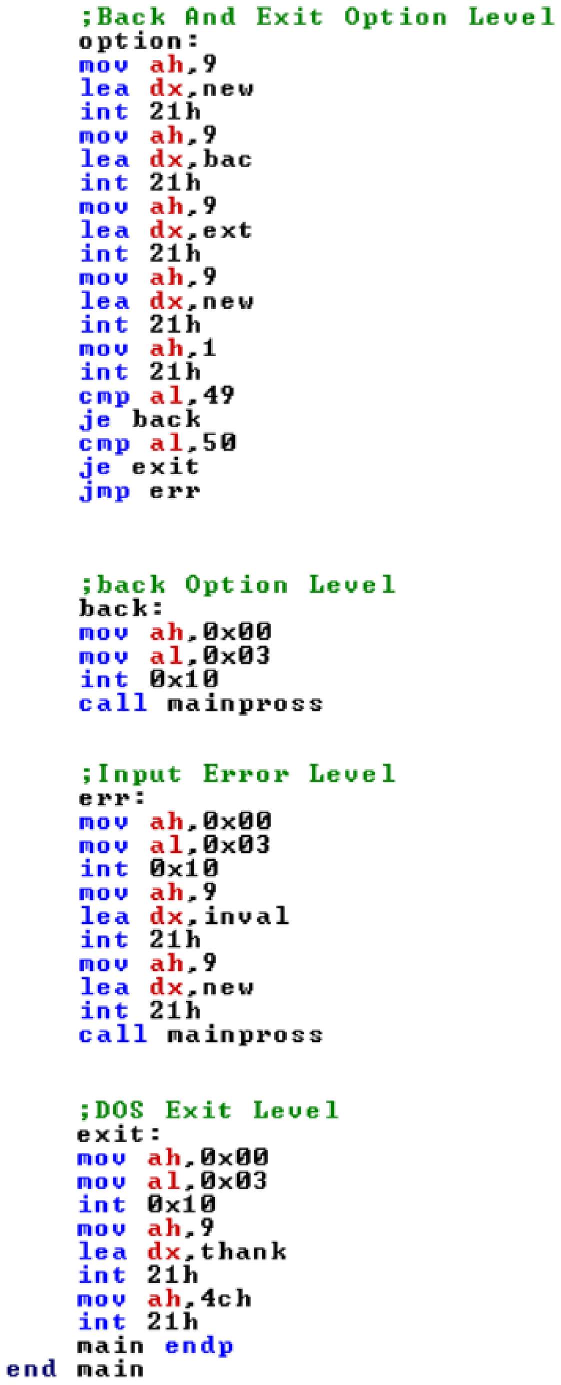
```
;Option 6
pross6:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,success
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,new
int 21h

mov ah,9
lea dx,t6
int 21h
mov ah,9
lea dx,av6
int 21h
mov ah,9
lea dx,new
int 21h
jmp option

;Option 7
pross7:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,success
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,new
int 21h

mov ah,9
lea dx,t7
int 21h
mov ah,9
lea dx,av7
int 21h
mov ah,9
lea dx,new
int 21h
jmp option
```



```
;Balance Transfer Level
Transfer:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,accountnum
int 21h

mov ah,1
acco:
int 21h
cmp al,13
jne acco

mov ah,9
lea dx,new
int 21h


;Again Password Cheek
mov ah,9
lea dx,a
int 21h

mov cx,pass1
mov bx,offset pass

again:
mov ah,8
int 21h

cmp al,[bx]
jne worng
mov ah,2
mov dl,42
int 21h
inc bx
loop again
mov ah,0x00
mov al,0x03
int 0x10
jmp transferpross


;Money Transfer Proccess
transferpross:
mov ah,9
lea dx,tranamount
int 21h
;Transfer Amount Display
mov ah,9
lea dx,fivehun
int 21h
```

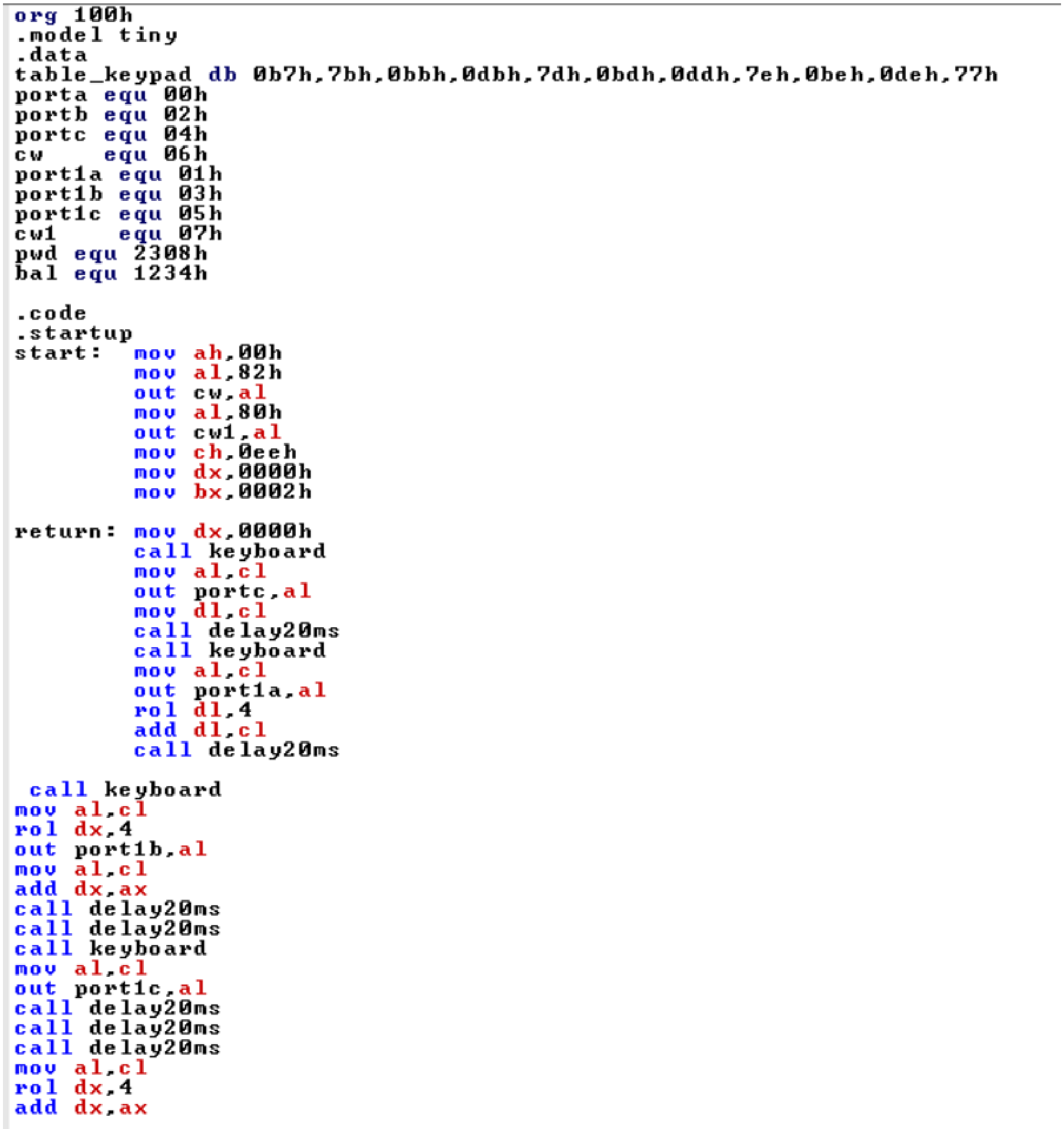


```
mov ah,9
lea dx,one
int 21h

mov ah,9
lea dx,three
int 21h

mov ah,9
lea dx,five
int 21h

mov ah,9
lea dx,ten
int 21h

mov ah,9
lea dx,fiften
int 21h

mov ah,9
lea dx,tweenty
int 21h

mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
mov bl,al

;Input Condition Cheek
cmp bl,49
je pross1

cmp bl,50
je pross2

cmp bl,51
je pross3

cmp bl,52
je pross4

cmp bl,53
je pross5

cmp bl,54
je pross6

cmp bl,55
je pross7
jmp err
```

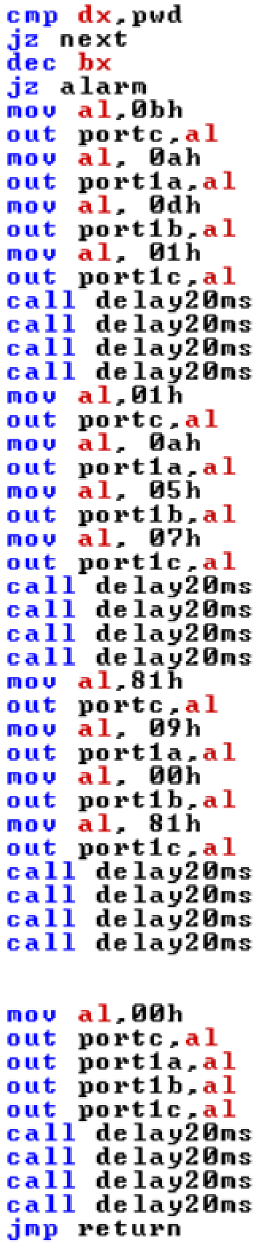


```
    ;Back And Exit Option Level
    option:
    mov ah,9
    lea dx,new
    int 21h
    mov ah,9
    lea dx,bac
    int 21h
    mov ah,9
    lea dx,ext
    int 21h
    mov ah,9
    lea dx,new
    int 21h
    mov ah,1
    int 21h
    cmp al,49
    je back
    cmp al,50
    je exit
    jmp err


    ;back Option Level
    back:
    mov ah,0x00
    mov al,0x03
    int 0x10
    call mainpross

    ;Input Error Level
    err:
    mov ah,0x00
    mov al,0x03
    int 0x10
    mov ah,9
    lea dx,inval
    int 21h
    mov ah,9
    lea dx,new
    int 21h
    call mainpross

    ;DOS Exit Level
    exit:
    mov ah,0x00
    mov al,0x03
    int 0x10
    mov ah,9
    lea dx,thank
    int 21h
    mov ah,4ch
    int 21h
    main endp
end main
```

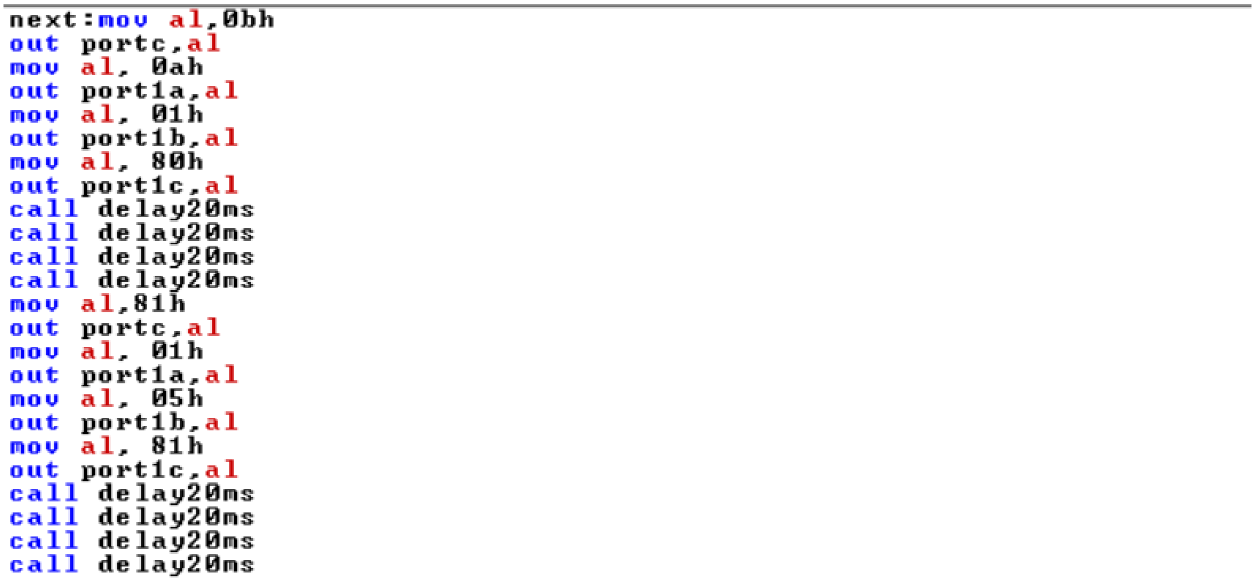
Keyboard Interfacing



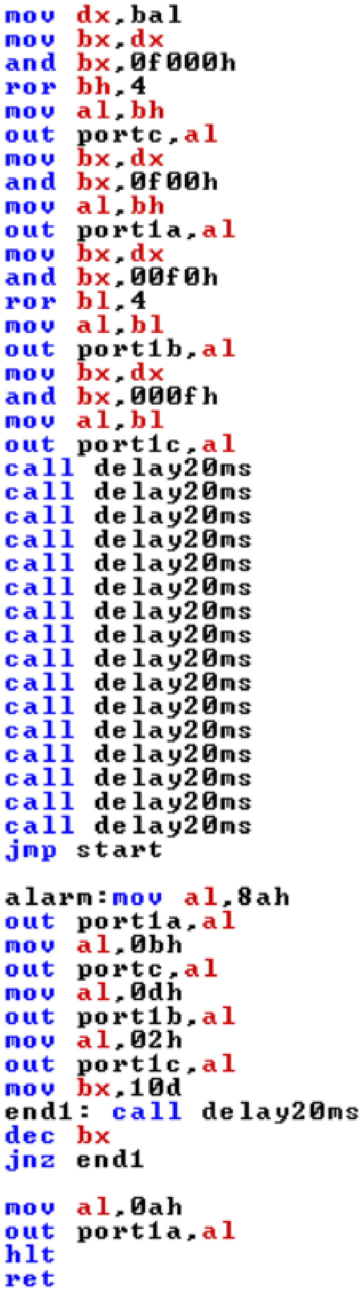
```
org 100h
.model tiny
.data
table_keypad db 0b7h,7bh,0bbh,0dbh,7dh,0bdh,0ddh,7eh,0beh,0deh,77h
porta equ 00h
portb equ 02h
portc equ 04h
cw    equ 06h
port1a equ 01h
port1b equ 03h
port1c equ 05h
cw1    equ 07h
pwd equ 2308h
bal equ 1234h

.code
.startup
start:  mov ah,00h
        mov al,82h
        out cw,al
        mov al,80h
        out cw1,al
        mov ch,0eeh
        mov dx,0000h
        mov bx,0002h

return: mov dx,0000h
        call keyboard
        mov al,cl
        out portc,al
        mov dl,cl
        call delay20ms
        call keyboard
        mov al,cl
        out port1a,al
        rol dl,4
        add dl,cl
        call delay20ms

 call keyboard
mov al,cl
rol dx,4
out port1b,al
mov al,cl
add dx,ax
call delay20ms
call delay20ms
call keyboard
mov al,cl
out port1c,al
call delay20ms
call delay20ms
call delay20ms
mov al,cl
rol dx,4
add dx,ax
```



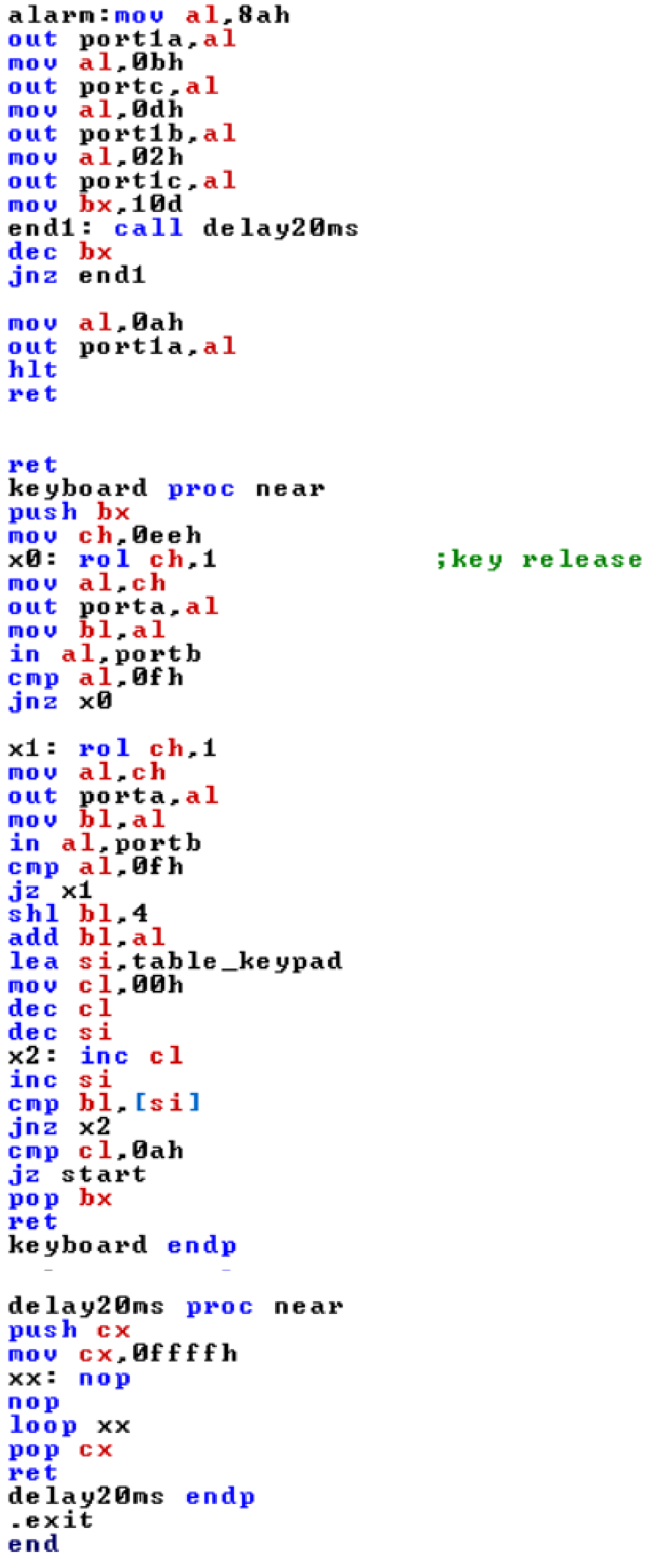
```
cmp dx,pwd
jz next
dec bx
jz alarm
mov al,0bh
out portc,al
mov al, 0ah
out port1a,al
mov al, 0dh
out port1b,al
mov al, 01h
out port1c,al
call delay20ms
call delay20ms
call delay20ms
call delay20ms
mov al,01h
out portc,al
mov al, 0ah
out port1a,al
mov al, 05h
out port1b,al
mov al, 07h
out port1c,al
call delay20ms
call delay20ms
call delay20ms
call delay20ms
mov al,81h
out portc,al
mov al, 09h
out port1a,al
mov al, 00h
out port1b,al
mov al, 81h
out port1c,al
call delay20ms
call delay20ms
call delay20ms
call delay20ms


mov al,00h
out portc,al
out port1a,al
out port1b,al
out port1c,al
call delay20ms
call delay20ms
call delay20ms
call delay20ms
jmp return
```



```
next:mov al,0bh
out portc,al
mov al, 0ah
out port1a,al
mov al, 01h
out port1b,al
mov al, 80h
out port1c,al
call delay20ms
call delay20ms
call delay20ms
call delay20ms
mov al,81h
out portc,al
mov al, 01h
out port1a,al
mov al, 05h
out port1b,al
mov al, 81h
out port1c,al
call delay20ms
call delay20ms
call delay20ms
call delay20ms
```



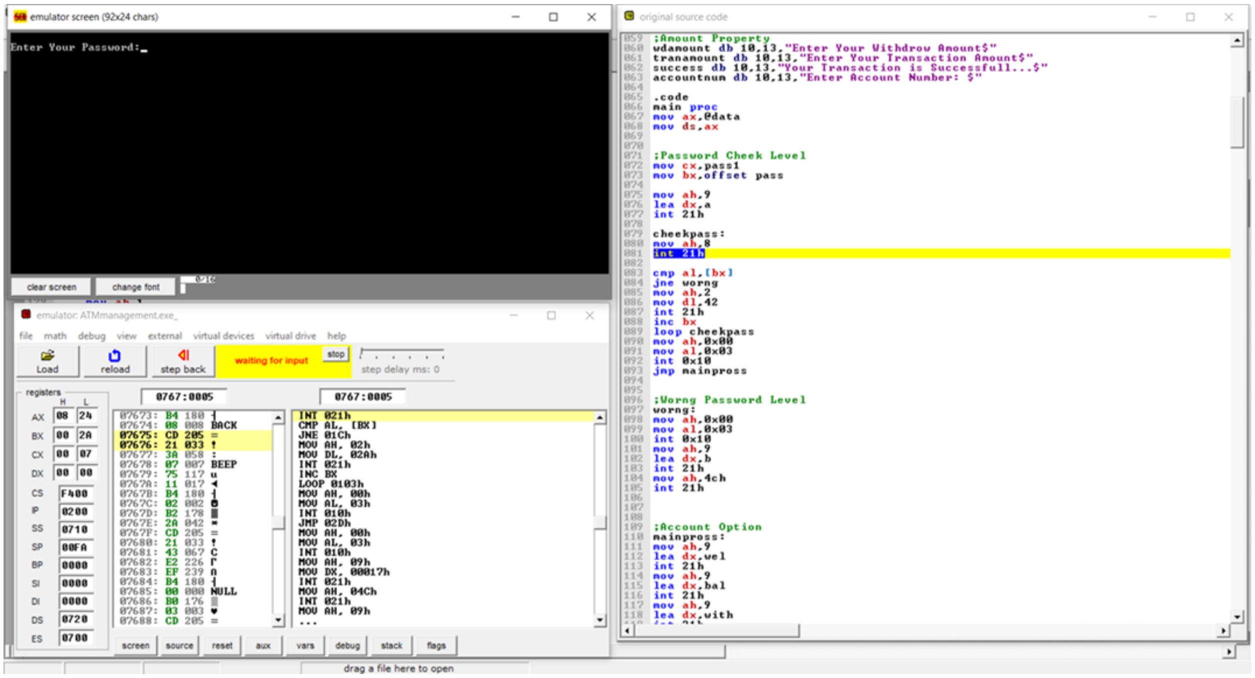
```
mov dx,bal
mov bx,dx
and bx,0f000h
ror bh,4
mov al,bh
out portc,al
mov bx,dx
and bx,0f00h
mov al,bh
out port1a,al
mov bx,dx
and bx,00f0h
ror bl,4
mov al,bl
out port1b,al
mov bx,dx
and bx,000fh
mov al,bl
out port1c,al
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
call delay20ms
jmp start

alarm:mov al,8ah
out port1a,al
mov al,0bh
out portc,al
mov al,0dh
out port1b,al
mov al,02h
out port1c,al
mov bx,10d
end1: call delay20ms
dec bx
jnz end1

mov al,0ah
out port1a,al
hlt
ret
```



```
alarm:mov al,8ah
out port1a,al
mov al,0bh
out portc,al
mov al,0dh
out port1b,al
mov al,02h
out port1c,al
mov bx,10d
end1: call delay20ms
dec bx
jnz end1

mov al,0ah
out port1a,al
hlt
ret


ret
keyboard proc near
push bx
mov ch,0eeh
x0: rol ch,1                ;key release
mov al,ch
out porta,al
mov bl,al
in al,portb
cmp al,0fh
jnz x0

x1: rol ch,1
mov al,ch
out porta,al
mov bl,al
in al,portb
cmp al,0fh
jz x1
shl bl,4
add bl,al
lea si,table_keypad
mov cl,00h
dec cl
dec si
x2: inc cl
inc si
cmp bl,[si]
jnz x2
cmp cl,0ah
jz start
pop bx
ret
keyboard endp

delay20ms proc near
push cx
mov cx,0ffffh
xx: nop
nop
loop xx
pop cx
ret
delay20ms endp
.exit
end
```

Output Screenshots:

First The User will be asked to enter password



```
Enter Your Password:_
```

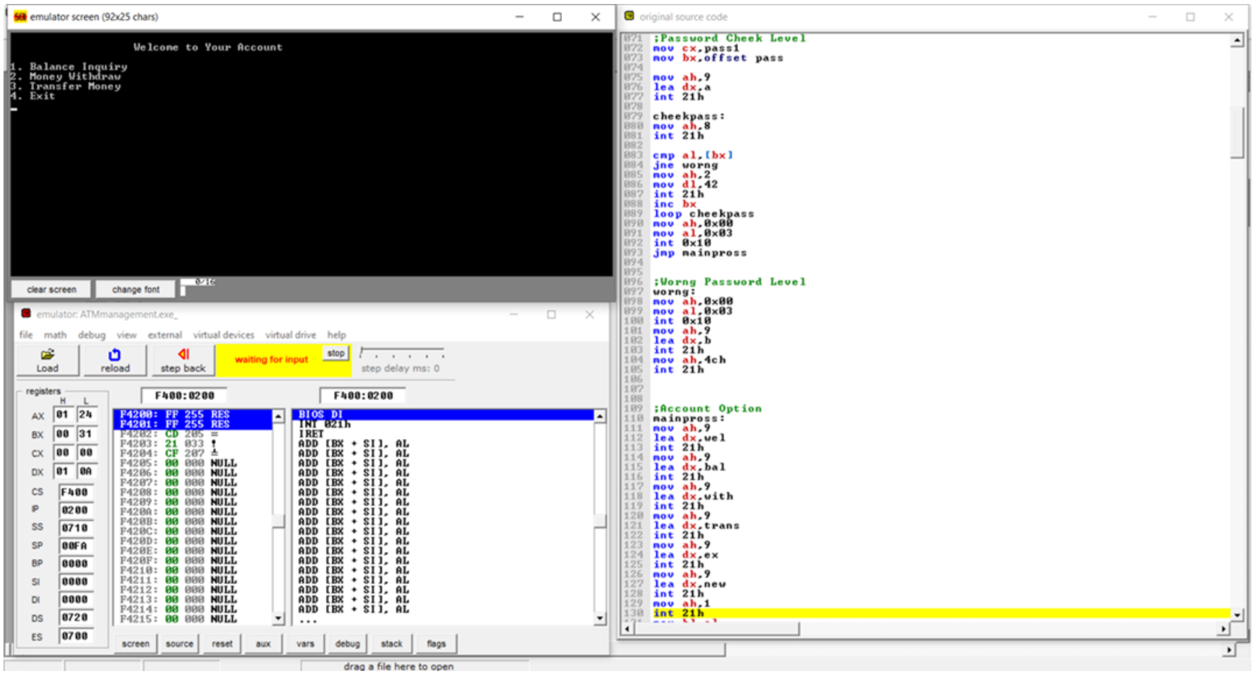
```
registers
     H   L
AX   08  24
BX   00  2A
CX   00  07
DX   00  00
CS   F400
IP   0200
SS   0710
SP   00FA
BP   0000
SI   0000
DI   0000
DS   0720
ES   0700
```

```
0767:0005

07673: B4 180 ┤
07674: 08 008 BACK
07675: CD 205 ═
07676: 21 033 !
07677: 3A 058 :
07678: 07 007 BEEP
07679: 75 117 u
0767A: 11 017 ◄
0767B: B4 180 ┤
0767C: 02 002 ☻
0767D: B2 178 ▓
0767E: 2A 042 *
0767F: CD 205 ═
07680: 21 033 !
07681: 43 067 C
07682: E2 226 Γ
07683: EF 239 ∩
07684: B4 180 ┤
07685: 00 000 NULL
07686: B0 176 ░
07687: 03 003 ♥
07688: CD 205 ═
```

```
0767:0005

INT 021h
CMP AL, [BX]
JNE 01Ch
MOV AH, 02h
MOV DL, 02Ah
INT 021h
INC BX
LOOP 0103h
MOV AH, 00h
MOV AL, 03h
INT 010h
JMP 02Dh
MOV AH, 00h
MOV AL, 03h
INT 010h
MOV AH, 09h
MOV DX, 00017h
INT 021h
MOV AH, 04Ch
INT 021h
MOV AH, 09h
...
```

```
;Amount Property
wdamount db 10,13,"Enter Your Withdrow Amount$"
tranamount db 10,13,"Enter Your Transaction Amount$"
success db 10,13,"Your Transaction is Successfull...$"
accountnum db 10,13,"Enter Account Number: $"

.code
main proc
mov ax,@data
mov ds,ax


;Password Cheek Level
mov cx,pass1
mov bx,offset pass

mov ah,9
lea dx,a
int 21h

cheekpass:
mov ah,8
int 21h

cmp al,[bx]
jne worng
mov ah,2
mov dl,42
int 21h
inc bx
loop cheekpass
mov ah,0x00
mov al,0x03
int 0x10
jmp mainpross


;Worng Password Level
worng:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,b
int 21h
mov ah,4ch
int 21h



;Account Option
mainpross:
mov ah,9
lea dx,wel
int 21h
mov ah,9
lea dx,bal
int 21h
mov ah,9
lea dx,with
```

After entering the correct password User will be directed to the menu



```
                         Welcome to Your Account

1. Balance Inquiry
2. Money Withdraw
3. Transfer Money
4. Exit
_
```

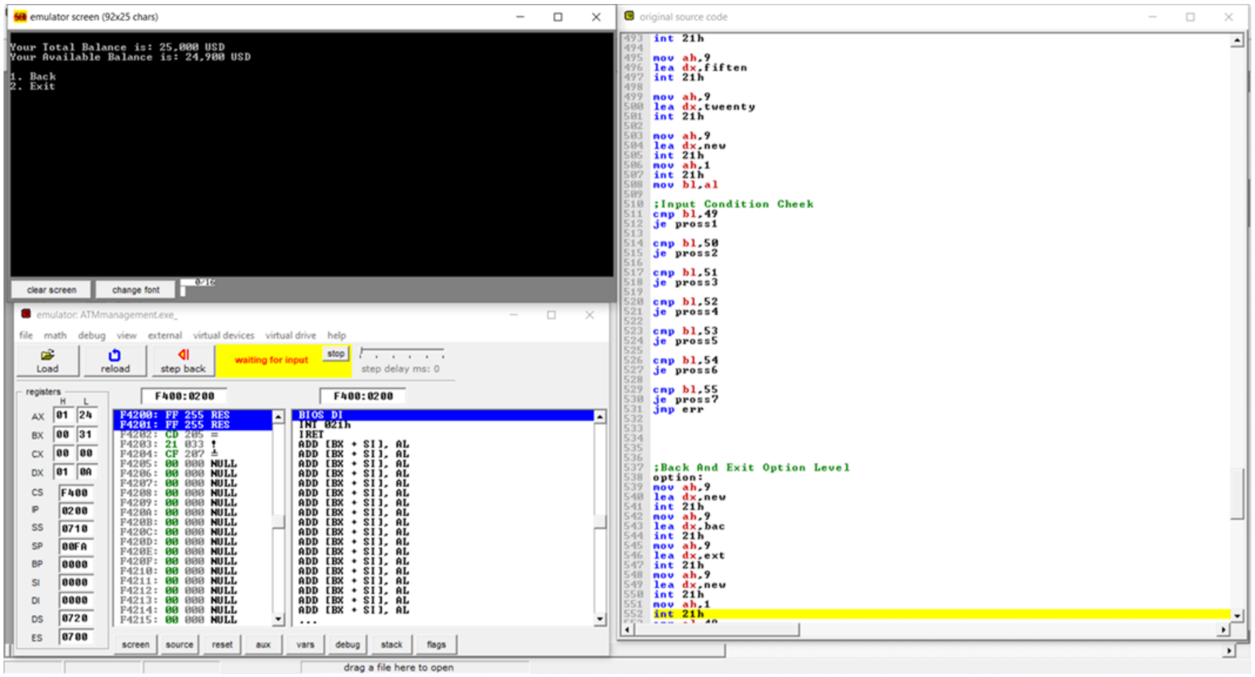
```
registers
     H   L
AX   01  24
BX   00  31
CX   00  00
DX   01  0A
CS   F400
IP   0200
SS   0710
SP   00FA
BP   0000
SI   0000
DI   0000
DS   0720
ES   0700
```

```
F400:0200

F4200: FF 255 RES
F4201: FF 255 RES
F4202: CD 205 ═
F4203: 21 033 !
F4204: CF 207 ╧
F4205: 00 000 NULL
F4206: 00 000 NULL
F4207: 00 000 NULL
F4208: 00 000 NULL
F4209: 00 000 NULL
F420A: 00 000 NULL
F420B: 00 000 NULL
F420C: 00 000 NULL
F420D: 00 000 NULL
F420E: 00 000 NULL
F420F: 00 000 NULL
F4210: 00 000 NULL
F4211: 00 000 NULL
F4212: 00 000 NULL
F4213: 00 000 NULL
F4214: 00 000 NULL
F4215: 00 000 NULL
```

```
F400:0200

BIOS DI
INT 021h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
;Password Cheek Level
mov cx,pass1
mov bx,offset pass

mov ah,9
lea dx,a
int 21h

cheekpass:
mov ah,8
int 21h

cmp al,[bx]
jne worng
mov ah,2
mov dl,42
int 21h
inc bx
loop cheekpass
mov ah,0x00
mov al,0x03
int 0x10
jmp mainpross


;Worng Password Level
worng:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,b
int 21h
mov ah,4ch
int 21h



;Account Option
mainpross:
mov ah,9
lea dx,wel
int 21h
mov ah,9
lea dx,bal
int 21h
mov ah,9
lea dx,with
int 21h
mov ah,9
lea dx,trans
int 21h
mov ah,9
lea dx,ex
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
```

If User selects option 1 he gets the balance enquiry



```
Your Total Balance is: 25,000 USD
Your Available Balance is: 24,900 USD

1. Back
2. Exit
```

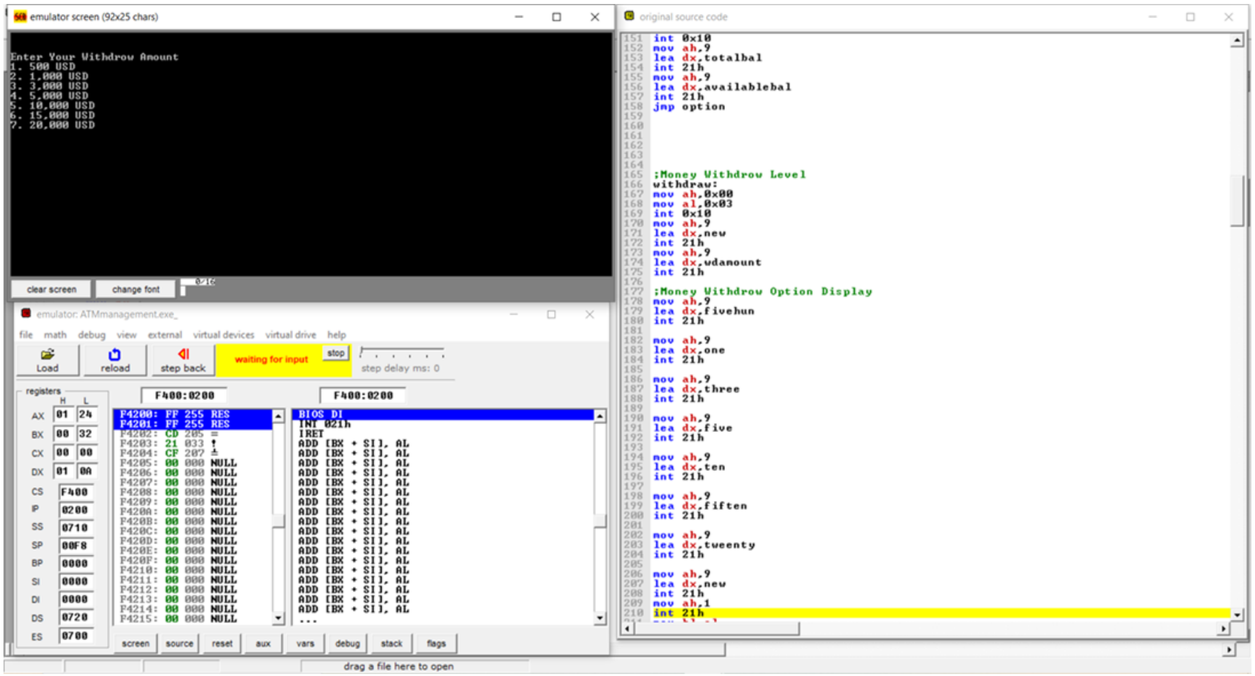
```
registers
     H   L
AX  01  24
BX  00  31
CX  00  00
DX  01  0A
CS  F400
IP  0200
SS  0710
SP  00FA
BP  0000
SI  0000
DI  0000
DS  0720
ES  0700
```

```
F400:0200
F4200: FF 255 RES
F4201: FF 255 RES
F4202: CD 205 ═
F4203: 21 033 !
F4204: CF 207 ╧
F4205: 00 000 NULL
F4206: 00 000 NULL
F4207: 00 000 NULL
F4208: 00 000 NULL
F4209: 00 000 NULL
F420A: 00 000 NULL
F420B: 00 000 NULL
F420C: 00 000 NULL
F420D: 00 000 NULL
F420E: 00 000 NULL
F420F: 00 000 NULL
F4210: 00 000 NULL
F4211: 00 000 NULL
F4212: 00 000 NULL
F4213: 00 000 NULL
F4214: 00 000 NULL
F4215: 00 000 NULL
```

```
F400:0200
BIOS DI
INT 021h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
int 21h

mov ah,9
lea dx,fiften
int 21h

mov ah,9
lea dx,tweenty
int 21h

mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
mov bl,al

;Input Condition Cheek
cmp bl,49
je pross1

cmp bl,50
je pross2

cmp bl,51
je pross3

cmp bl,52
je pross4

cmp bl,53
je pross5

cmp bl,54
je pross6

cmp bl,55
je pross7
jmp err





;Back And Exit Option Level
option:
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,bac
int 21h
mov ah,9
lea dx,ext
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
```

If User selects option 2 then he will be asked to withdraw the amount.



```
Enter Your Withdrow Amount
1. 500 USD
2. 1,000 USD
3. 3,000 USD
4. 5,000 USD
5. 10,000 USD
6. 15,000 USD
7. 20,000 USD
```

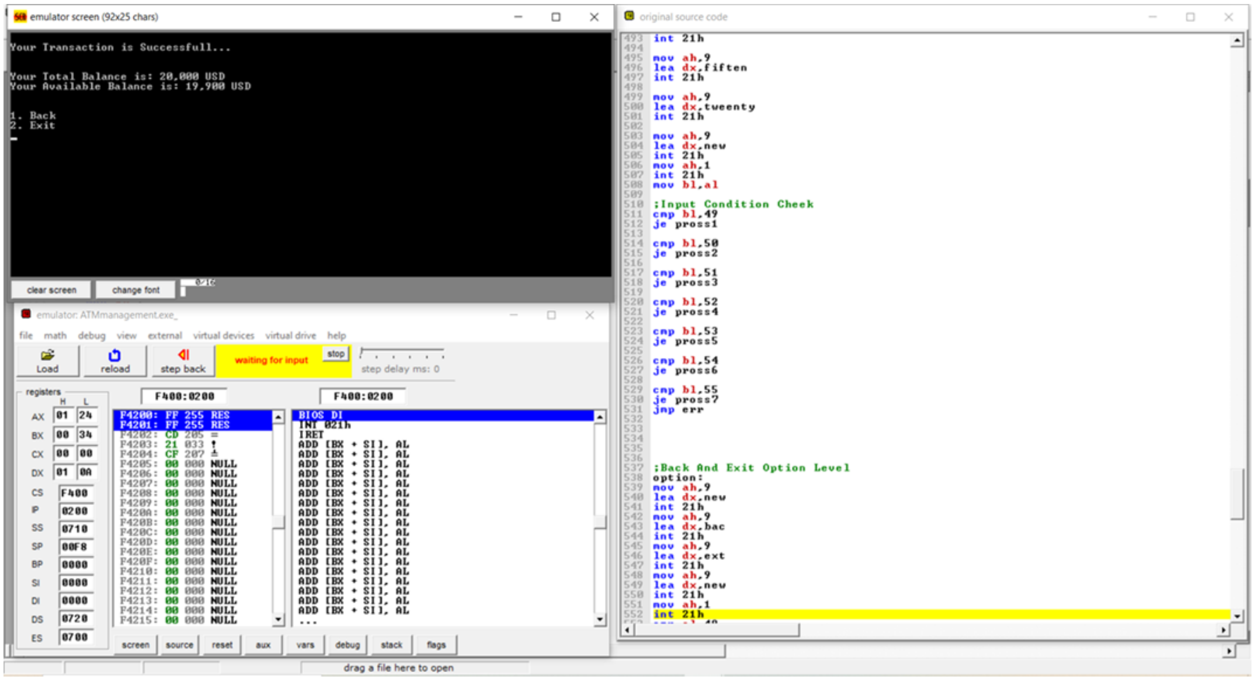
```
registers
     H   L
AX  01  24
BX  00  32
CX  00  00
DX  01  0A
CS  F400
IP  0200
SS  0710
SP  00F8
BP  0000
SI  0000
DI  0000
DS  0720
ES  0700
```

```
F400:0200
F4200: FF 255 RES
F4201: FF 255 RES
F4202: CD 205 ═
F4203: 21 033 !
F4204: CF 207 ╧
F4205: 00 000 NULL
F4206: 00 000 NULL
F4207: 00 000 NULL
F4208: 00 000 NULL
F4209: 00 000 NULL
F420A: 00 000 NULL
F420B: 00 000 NULL
F420C: 00 000 NULL
F420D: 00 000 NULL
F420E: 00 000 NULL
F420F: 00 000 NULL
F4210: 00 000 NULL
F4211: 00 000 NULL
F4212: 00 000 NULL
F4213: 00 000 NULL
F4214: 00 000 NULL
F4215: 00 000 NULL
```

```
F400:0200
BIOS DI
INT 021h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
int 0x10
mov ah,9
lea dx,totalbal
int 21h
mov ah,9
lea dx,availablebal
int 21h
jmp option






;Money Withdrow Level
withdraw:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,wdamount
int 21h

;Money Withdrow Option Display
mov ah,9
lea dx,fivehun
int 21h

mov ah,9
lea dx,one
int 21h

mov ah,9
lea dx,three
int 21h

mov ah,9
lea dx,five
int 21h

mov ah,9
lea dx,ten
int 21h

mov ah,9
lea dx,fiften
int 21h

mov ah,9
lea dx,tweenty
int 21h

mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
```

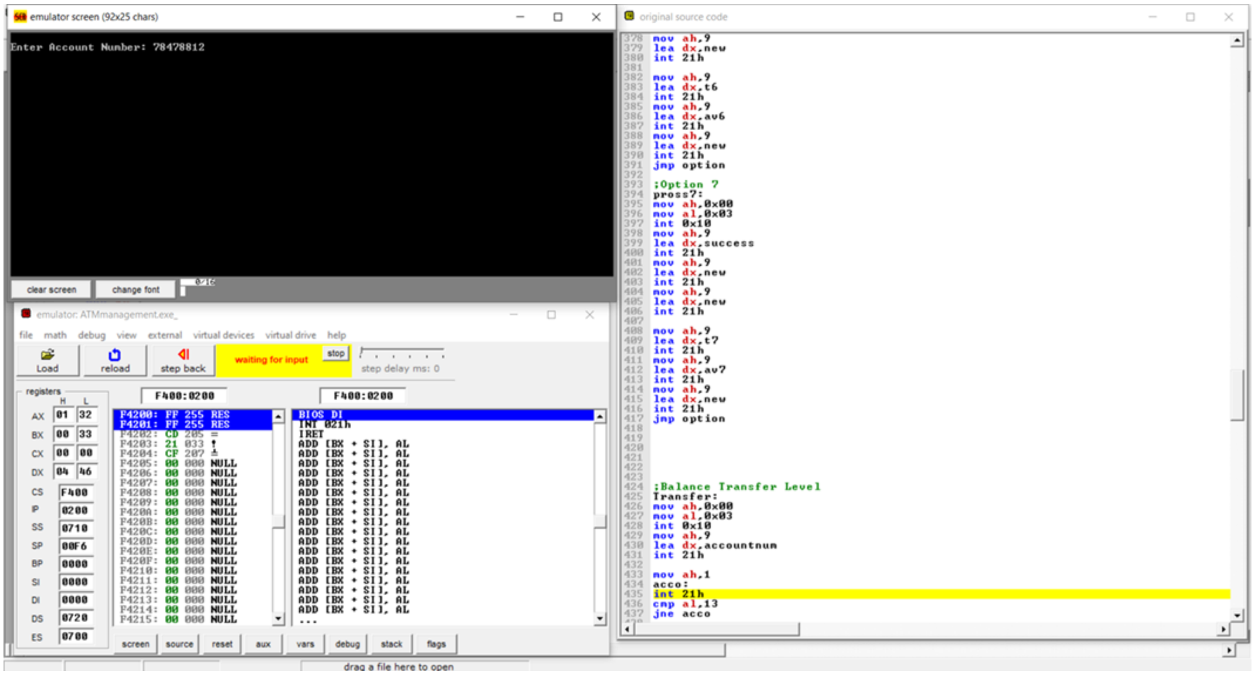
For example, User chose $5000 then it will be deducted from the account.



```
Your Transaction is Successfull...

Your Total Balance is: 20,000 USD
Your Available Balance is: 19,900 USD

1. Back
2. Exit
_
```

```
int 21h

mov ah,9
lea dx,fiften
int 21h

mov ah,9
lea dx,tweenty
int 21h

mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
mov bl,al

;Input Condition Cheek
cmp bl,49
je pross1

cmp bl,50
je pross2

cmp bl,51
je pross3

cmp bl,52
je pross4

cmp bl,53
je pross5

cmp bl,54
je pross6

cmp bl,55
je pross7
jmp err




;Back And Exit Option Level
option:
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,bac
int 21h
mov ah,9
lea dx,ext
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
```

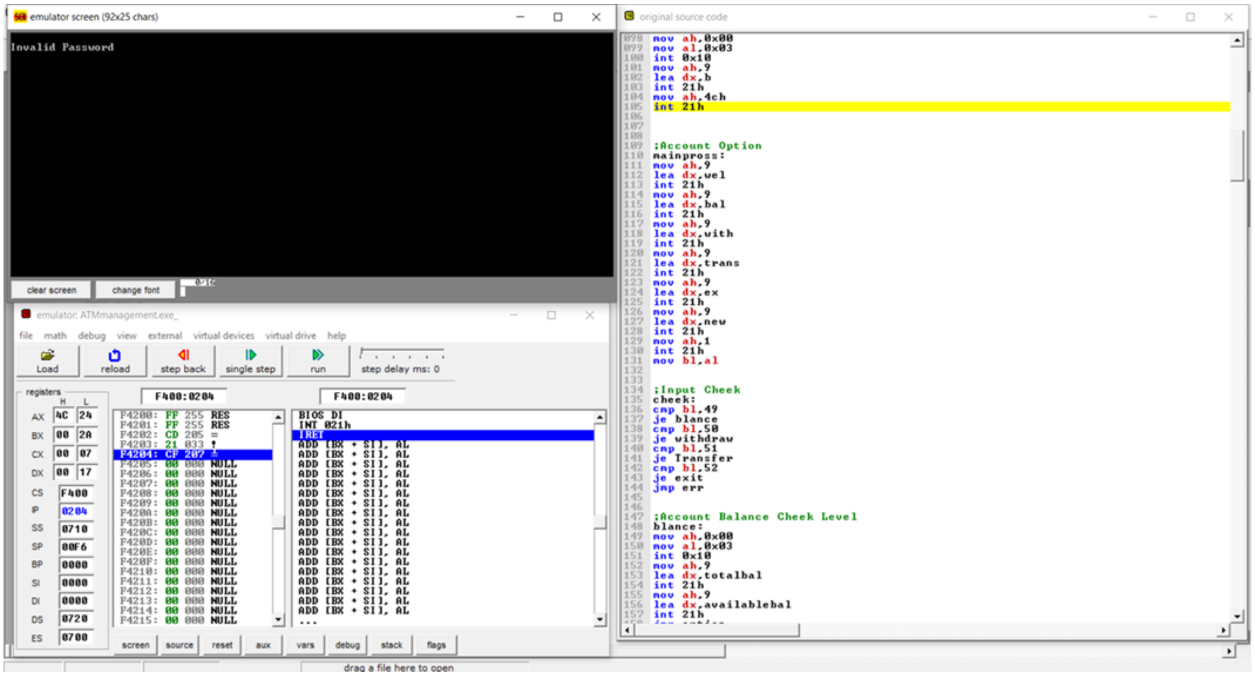
If he selects option 3 then User will be asked to enter account number



```
Enter Account Number: 78478812
```

```
mov ah,9
lea dx,new
int 21h

mov ah,9
lea dx,t6
int 21h
mov ah,9
lea dx,av6
int 21h
mov ah,9
lea dx,new
int 21h
jmp option

;Option 7
pross7:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,success
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,new
int 21h

mov ah,9
lea dx,t7
int 21h
mov ah,9
lea dx,av7
int 21h
mov ah,9
lea dx,new
int 21h
jmp option






;Balance Transfer Level
Transfer:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,accountnum
int 21h

mov ah,1
acco:
int 21h
cmp al,13
jne acco
```

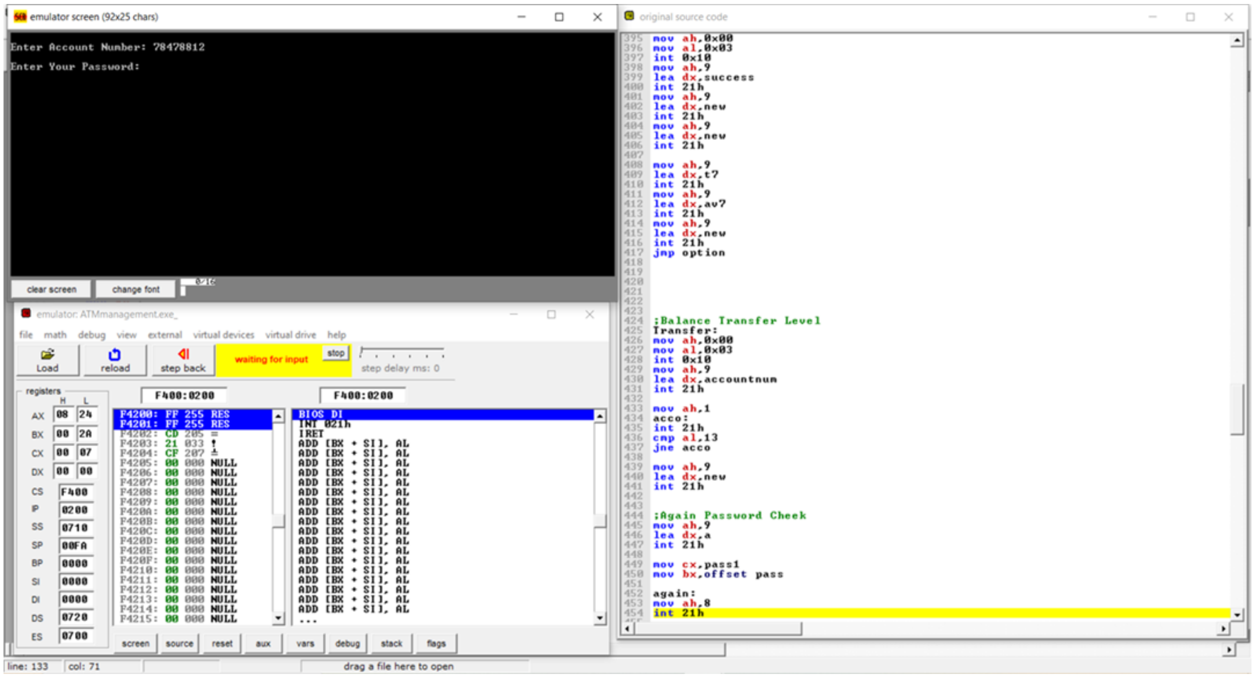
If he enters wrong password then User will be displayed invalid password



```
Invalid Password
```

```
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,b
int 21h
mov ah,4ch
int 21h



;Account Option
mainpross:
mov ah,9
lea dx,wel
int 21h
mov ah,9
lea dx,bal
int 21h
mov ah,9
lea dx,with
int 21h
mov ah,9
lea dx,trans
int 21h
mov ah,9
lea dx,ex
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
mov bl,al


;Input Cheek
cheek:
cmp bl,49
je blance
cmp bl,50
je withdraw
cmp bl,51
je Transfer
cmp bl,52
je exit
jmp err


;Account Balance Cheek Level
blance:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,totalbal
int 21h
mov ah,9
lea dx,availablebal
int 21h
```

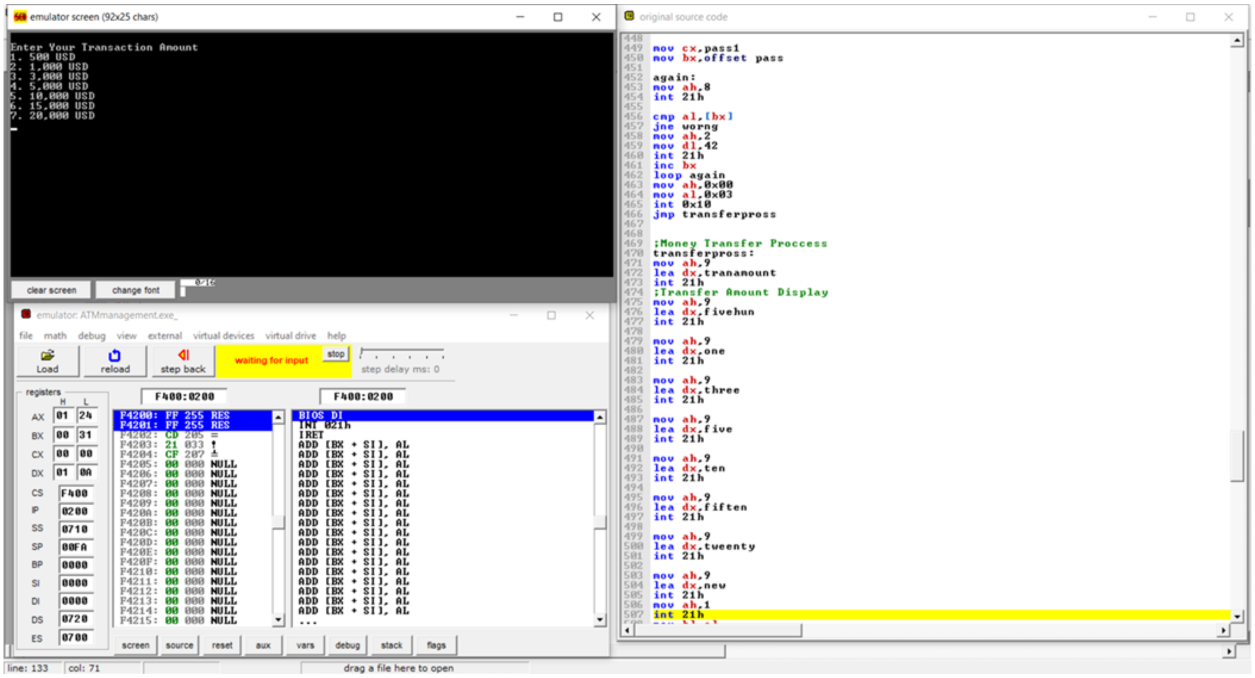
After entering account number User will be asked to enter password



```
Enter Account Number: 78478812

Enter Your Password:
```

```
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,success
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,new
int 21h

mov ah,9
lea dx,t7
int 21h
mov ah,9
lea dx,av7
int 21h
mov ah,9
lea dx,new
int 21h
jmp option






;Balance Transfer Level
Transfer:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,accountnum
int 21h

mov ah,1
acco:
int 21h
cmp al,13
jne acco

mov ah,9
lea dx,new
int 21h


;Again Password Cheek
mov ah,9
lea dx,a
int 21h

mov cx,pass1
mov bx,offset pass

again:
mov ah,8
int 21h
```

User will be asked to select then amount he wants to transfer



```
Enter Your Transaction Amount
1. 500 USD
2. 1,000 USD
3. 3,000 USD
4. 5,000 USD
5. 10,000 USD
6. 15,000 USD
7. 20,000 USD
_
```

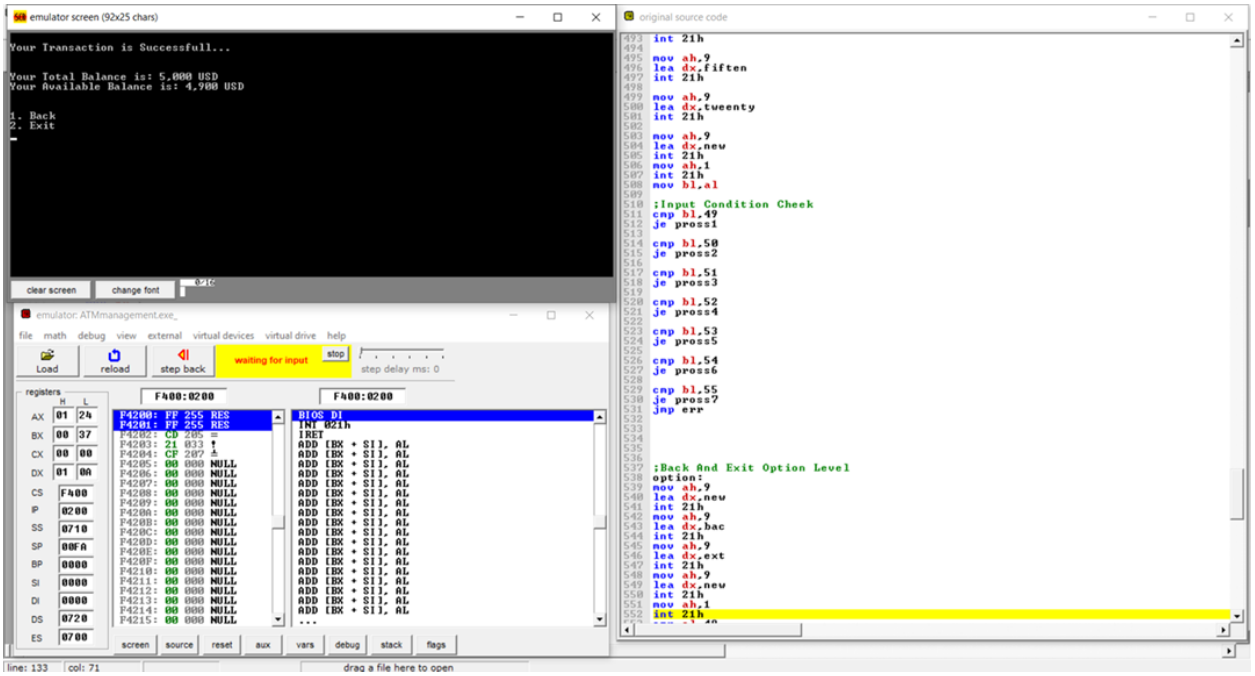
```
    H  L
AX  01 24
BX  00 31
CX  00 00
DX  01 0A
CS  F400
IP  0200
SS  0710
SP  00FA
BP  0000
SI  0000
DI  0000
DS  0720
ES  0700
```

```
F400:0200
F4200: FF 255 RES
F4201: FF 255 RES
F4202: CD 205 =
F4203: 21 033 !
F4204: CF 207 ±
F4205: 00 000 NULL
F4206: 00 000 NULL
F4207: 00 000 NULL
F4208: 00 000 NULL
F4209: 00 000 NULL
F420A: 00 000 NULL
F420B: 00 000 NULL
F420C: 00 000 NULL
F420D: 00 000 NULL
F420E: 00 000 NULL
F420F: 00 000 NULL
F4210: 00 000 NULL
F4211: 00 000 NULL
F4212: 00 000 NULL
F4213: 00 000 NULL
F4214: 00 000 NULL
F4215: 00 000 NULL
```

```
F400:0200
BIOS DI
INT 021h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```

mov cx,pass1
mov bx,offset pass

again:
mov ah,8
int 21h

cmp al,[bx]
jne worng
mov ah,2
mov dl,42
int 21h
inc bx
loop again
mov ah,0x00
mov al,0x03
int 0x10
jmp transferpross


;Money Transfer Proccess
transferpross:
mov ah,9
lea dx,tranamount
int 21h
;Transfer Amount Display
mov ah,9
lea dx,fivehun
int 21h

mov ah,9
lea dx,one
int 21h

mov ah,9
lea dx,three
int 21h

mov ah,9
lea dx,five
int 21h

mov ah,9
lea dx,ten
int 21h

mov ah,9
lea dx,fiften
int 21h

mov ah,9
lea dx,tweenty
int 21h

mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
```

After transaction user will be displayed the remaining balance



```
Your Transaction is Successfull...


Your Total Balance is: 5,000 USD
Your Available Balance is: 4,900 USD


1. Back
2. Exit
_
```

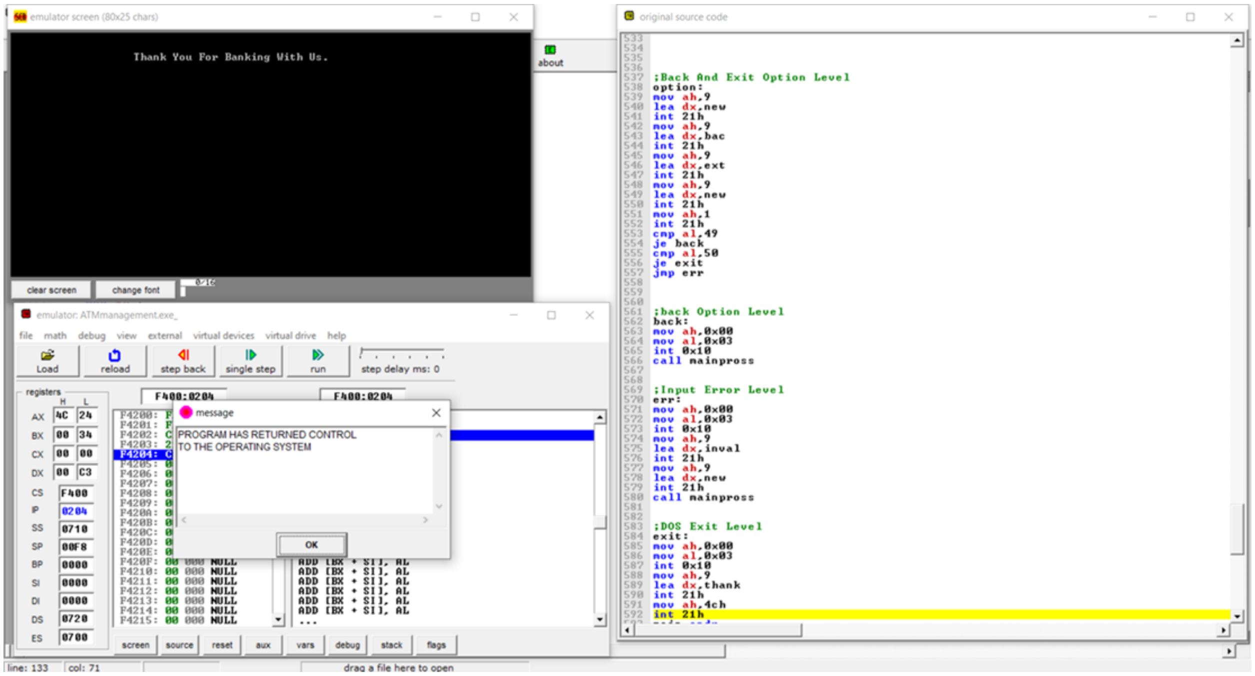
```
    H  L
AX  01 24
BX  00 37
CX  00 00
DX  01 0A
CS  F400
IP  0200
SS  0710
SP  00FA
BP  0000
SI  0000
DI  0000
DS  0720
ES  0700
```

```
F400:0200
F4200: FF 255 RES
F4201: FF 255 RES
F4202: CD 205 =
F4203: 21 033 !
F4204: CF 207 ±
F4205: 00 000 NULL
F4206: 00 000 NULL
F4207: 00 000 NULL
F4208: 00 000 NULL
F4209: 00 000 NULL
F420A: 00 000 NULL
F420B: 00 000 NULL
F420C: 00 000 NULL
F420D: 00 000 NULL
F420E: 00 000 NULL
F420F: 00 000 NULL
F4210: 00 000 NULL
F4211: 00 000 NULL
F4212: 00 000 NULL
F4213: 00 000 NULL
F4214: 00 000 NULL
F4215: 00 000 NULL
```

```
F400:0200
BIOS DI
INT 021h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

```
int 21h

mov ah,9
lea dx,fiften
int 21h

mov ah,9
lea dx,tweenty
int 21h

mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
mov bl,al

;Input Condition Cheek
cmp bl,49
je pross1

cmp bl,50
je pross2

cmp bl,51
je pross3

cmp bl,52
je pross4

cmp bl,53
je pross5

cmp bl,54
je pross6

cmp bl,55
je pross7
jmp err





;Back And Exit Option Level
option:
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,bac
int 21h
mov ah,9
lea dx,ext
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
```

If the user selects exit he will be displayed Thank you for banking with us.



```
Thank You For Banking With Us.
```

```
PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM
```

```
     H  L
AX  4C 24
BX  00 34
CX  00 00
DX  00 C3
CS  F400
IP  0204
SS  0710
SP  00F8
BP  0000
SI  0000
DI  0000
DS  0720
ES  0700
```

```
F400:0204                      F400:0204
F420F: 00 000 NULL             ADD [BX + SI], AL
F4210: 00 000 NULL             ADD [BX + SI], AL
F4211: 00 000 NULL             ADD [BX + SI], AL
F4212: 00 000 NULL             ADD [BX + SI], AL
F4213: 00 000 NULL             ADD [BX + SI], AL
F4214: 00 000 NULL             ADD [BX + SI], AL
F4215: 00 000 NULL             ...
```

```
;Back And Exit Option Level
option:
mov ah,9
lea dx,new
int 21h
mov ah,9
lea dx,bac
int 21h
mov ah,9
lea dx,ext
int 21h
mov ah,9
lea dx,new
int 21h
mov ah,1
int 21h
cmp al,49
je back
cmp al,50
je exit
jmp err



;back Option Level
back:
mov ah,0x00
mov al,0x03
int 0x10
call mainpross


;Input Error Level
err:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,inval
int 21h
mov ah,9
lea dx,new
int 21h
call mainpross


;DOS Exit Level
exit:
mov ah,0x00
mov al,0x03
int 0x10
mov ah,9
lea dx,thank
int 21h
mov ah,4ch
int 21h
```

Conclusion:

The Internet has a good impact on bank management in general, and delivery channel management in particular. Electronic and internet-based banking services such as ATMs, Net Banking, and Mobile Banking have given the banking industry a whole new look. Banks all around the world have acknowledged the importance of responding to changes in the way they give services to their consumers.

Approximately a million ATMs have been placed by various banks throughout the world. Because of their benefits, ATMs are becoming increasingly popular. In India, too, it is believed that automating banking operations through ATMs will save clients time and effort by eliminating the need for time-consuming paperwork and procedures.

India has just over 35,000 ATMs (as of 2009), or just over 1 ATM per million people. In India, ATM services have yet to gain traction. At the same time, India's low ATM penetration presents a huge opportunity for the expansion of ATM services. The increased acceptance of ATMs as a mechanism of expansion and better responsiveness among banks and customers is being matched by the demand for ATM services caused by rising middle-class earnings.

We attempted to create an online ATM system and exhibit all of its features. Essentially, in this project, we designed an assembly language programme to implement the functions that the atm system provides to the customer, such as knowing their account balance, withdrawing money from the atm, and transferring money to different bank accounts.

To implement these various functions, we performed various operations on general-purpose registers and then displayed the results to the users.

The microprocessor 8086 has been successfully shown in the use and demonstration of ATM systems. We eliminated the conventional dependencies and overheads that arise during ATM transactions by employing an efficient codebase.

We've also included a functional prototype of an interfacing script that uses 8255 and keypad interfaces to connect the system's many working components and make them work together in a standalone environment.

As hardware becomes available, the project can be scaled to real-life operational hardware components and therefore integrated into actual ATM systems.

References:

1. CAI, W. X., SHI, Y. P., & SUN, H. Q. (2015). Application of EMU8086 in the Teaching of Microcomputer Theory and Interface Technology. Journal of Shijiazhuang University, (3), 3.
2. Jegede, C. A. (2014). Effects of automated teller machines on the performance of Nigerian banks. American Journal of Applied Mathematics and Statistics, 2(1), 40-46.
3. Hitchcock, R. C. (1992). U.S. Patent No. 5,157,717. Washington, DC: U.S. Patent and
Trademark Office.
4. Cash matters: Now, more than ever in the time of coronavirus | ATM Marketplace
5. Are contactless ATM transactions a solution during the Covid-19 pandemic? | Deccan Herald
6. How Do ATMs Work? - YouTube
7. ATM Machine System in C++ (engineersgarage.com)

8.https://www.researchgate.net/publication/333365920_THE_EFFECT_OF_ATM_SERVICE_QUALITY_ON_CUSTOMER'S_SATISFACTION_AND_LOYALTY_AN_EMPIRICAL_ANALYSIS

9.https://www.researchgate.net/publication/321623369_A_Study_on_Services_Quality_in_ATM_Services_in_Public_Sector_Banks_in_Kanchipuram_Town

10. https://www.nxp.com/docs/en/application-note/AN2060.pdf