SSN College of Engineering Department of Computer Science and Engineering

III year - UCS1512 – Microprocessors Lab

Display system date and time

Exp No: 11

Name: Rahul Ram M

Register Number: 185001121

Date: 13/10/2020

11a) Display system date:

Aim:

To design 8086-program for displaying system date.

Procedure for executing MASM:

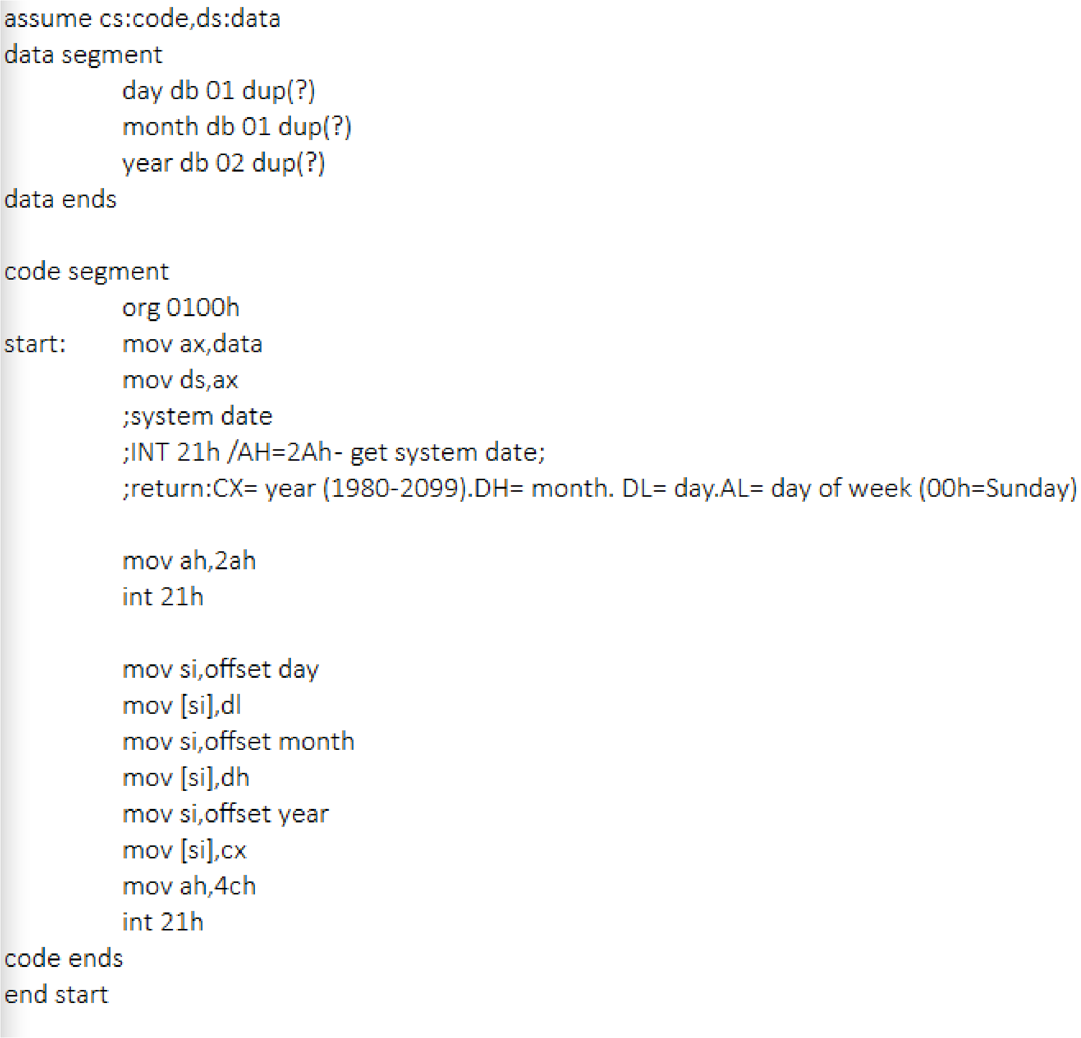
1. Run Dosbox and mount your masm folder to a drive in dosbox.
2. Goto the mounted drive.
3. Save the 8086 program with extension .asm in the same folder using command “edit”
4. After creating the file, assemble it using the command “masm filename.asm”
5. Link the file using the command “link filename.obj;”
6. Use debug command with filename.exe to execute and analyse the memory contents, “debug filename.exe”.
7. In debug, command “u” will display the unassembled code.
8. Use command “d segment:offset” to see the content of memory locations starting from segment:offset address.
9. To change the value in memory, use the command “e segment:offset”
10. Verify the memory contents to ensure the updates (using command “d”).
11. . Execute using the command “g” and check the outputs.
12. “q” to exit from debug and “exit” to exit from command prompt and to close the Dosbox.

Algorithm:

1. START: Move the starting address of data segment to AX register and move the data from AX register to DS register.
2. Move 2ah to AH register.
3. Calling int 21H with 2a in AH register will return year in CX register, month in DH register, day in DL register and day of the week in AL register.
4. Move the offset of the variable DAY in SI register.
5. Move the contents stored in DL register to the location in SI register.

6. Move the offset of the variable MONTH in SI register.
7. Move the contents stored in DH register to the location in SI register.
8. Move the offset of the variable YEAR in SI register.
9. Move the contents stored in CX register to the location in SI register.
10. Move the hexadecimal value 4C into AH register. INT 21H means invoke the interrupt identified by the hexadecimal number 21. In MS-DOS, invoking interrupt 21h while AH = 4Ch causes the current process to terminate and uses the value of register AL as the exit code of the process.

Program:

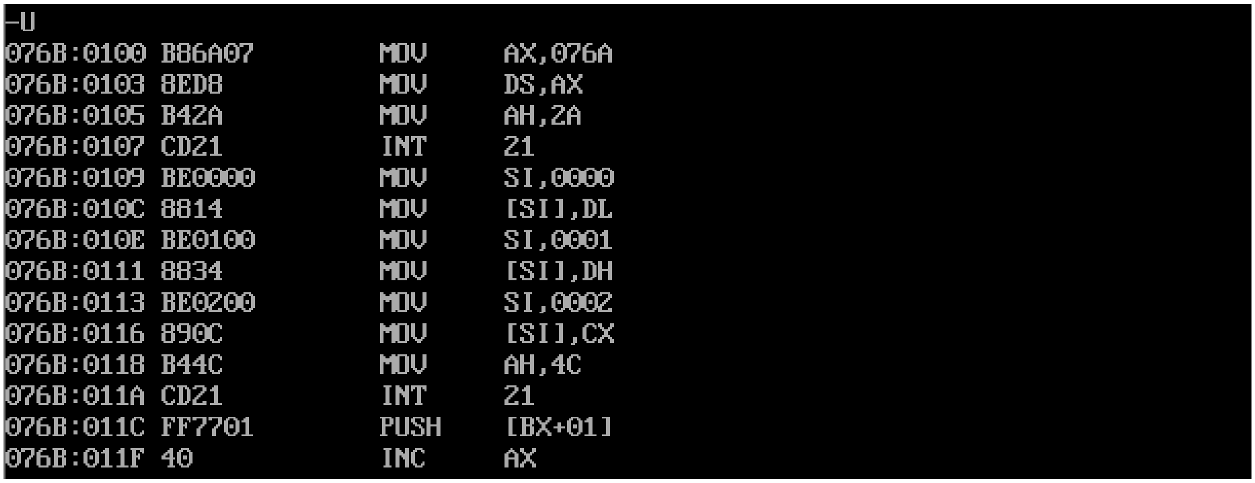


```
assume cs:code,ds:data
data segment
            day db 01 dup(?)
            month db 01 dup(?)
            year db 02 dup(?)
data ends

code segment
            org 0100h
start:      mov ax,data
            mov ds,ax
            ;system date
            ;INT 21h /AH=2Ah- get system date;
            ;return:CX= year (1980-2099).DH= month. DL= day.AL= day of week (00h=Sunday)

            mov ah,2ah
            int 21h

            mov si,offset day
            mov [si],dl
            mov si,offset month
            mov [si],dh
            mov si,offset year
            mov [si],cx
            mov ah,4ch
            int 21h
code ends
end start
```

	Program	**Comments**
START:	MOV AX, DATA MOV DS, AX	Transferring the data from DATA to AX register and from AX register to DS register.
	MOV AH, 2AH	AH <- 2AH
	INT 21H	INT 21h /AH=2Ah - get system date
	MOV SI, OFFSET DAY	SI <- DAY
	MOV [SI], DL	[SI] <- DL

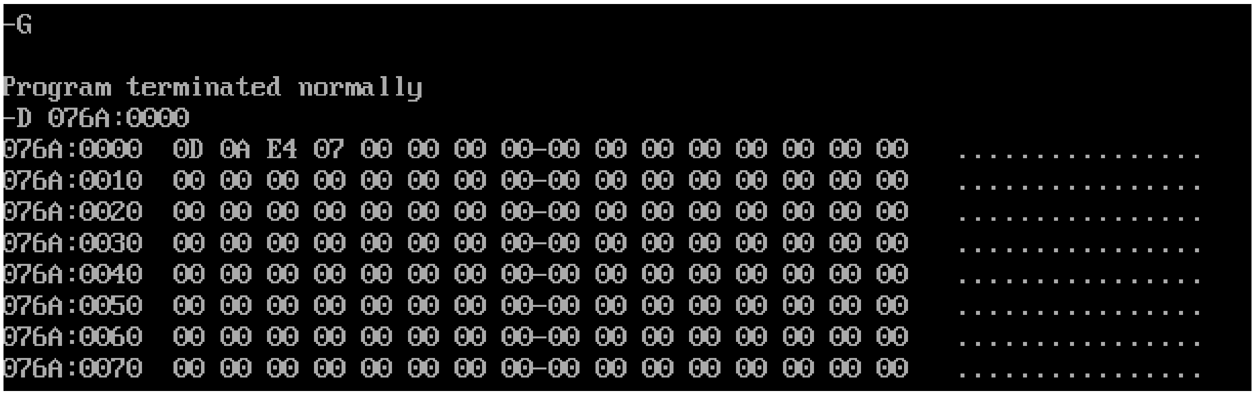
	MOV SI, OFFSET MONTH	SI <- MONTH
	MOV [SI], DH	[SI] <- DH
	MOV SI, OFFSET YEAR	SI <- YEAR
	MOV [SI], CX	[SI] <- CX
	MOV AH,4CH	Setup function-4C of the int21.
	INT 21H	Call BIOS int21 to return to DOS.

Unassembled Code:



```
-U
076B:0100 B86A07        MOV     AX,076A
076B:0103 8ED8          MOV     DS,AX
076B:0105 B42A          MOV     AH,2A
076B:0107 CD21          INT     21
076B:0109 BE0000        MOV     SI,0000
076B:010C 8814          MOV     [SI],DL
076B:010E BE0100        MOV     SI,0001
076B:0111 8834          MOV     [SI],DH
076B:0113 BE0200        MOV     SI,0002
076B:0116 890C          MOV     [SI],CX
076B:0118 B44C          MOV     AH,4C
076B:011A CD21          INT     21
076B:011C FF7701        PUSH    [BX+01]
076B:011F 40            INC     AX
```

Snapshot of sample input and output:



```
-G

Program terminated normally
-D 076A:0000
076A:0000  0D 0A E4 07 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

Result:

Thus the 8086-program for displaying system date is executed successfully in DOS-BOX.

11b) Display system time:

Aim:

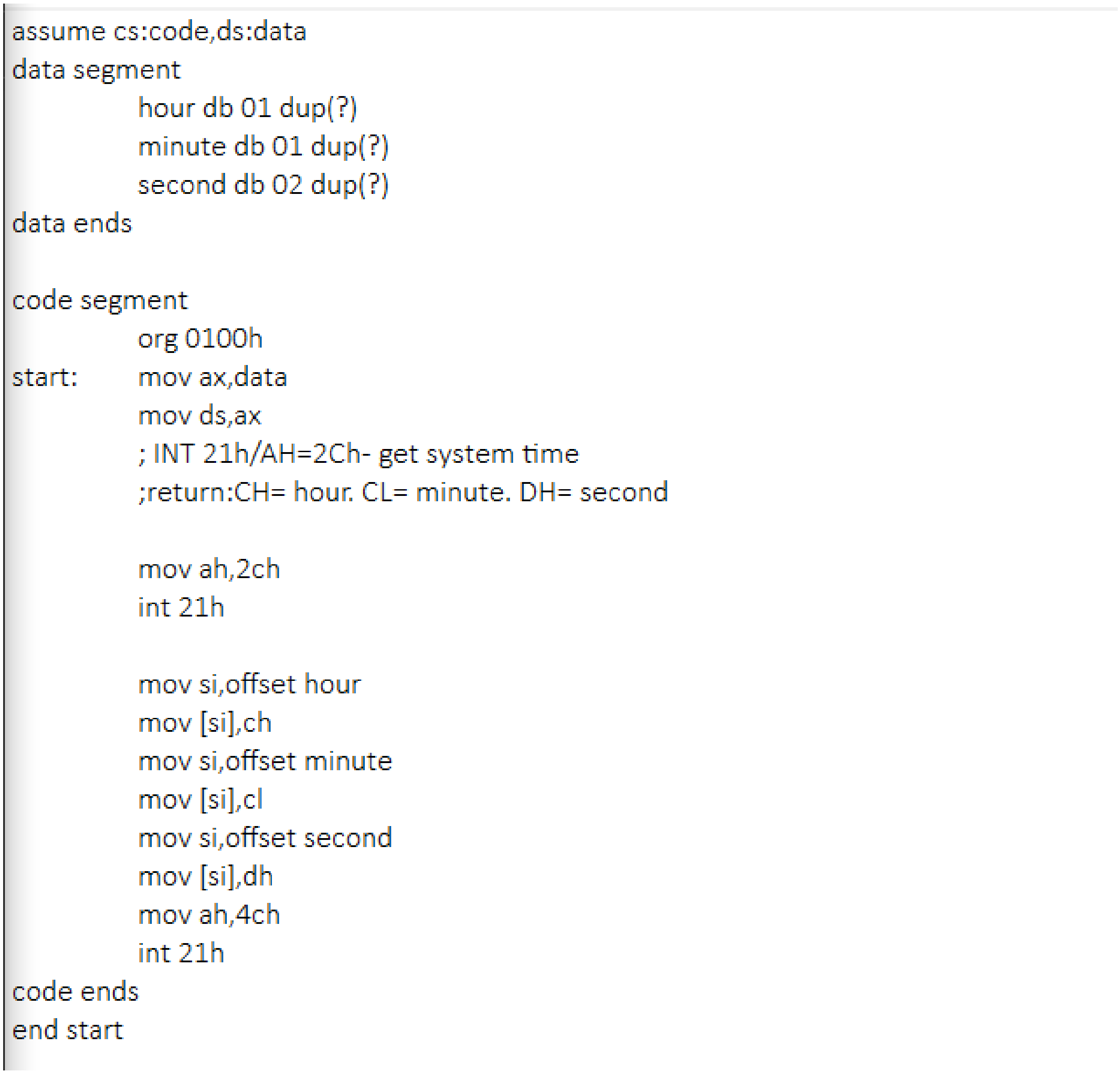
To design 8086-program for displaying system time.

Algorithm:

1. START: Move the starting address of data segment to AX register and move the data from AX register to DS register.

2. Move 2ch to AH register.
3. Calling int 21H with 2c in AH register will return hour in CH register, minute in CL register and second in DH register.
4. Move the offset of the variable HOUR in SI register.
5. Move the contents stored in CH register to the location in SI register.
6. Move the offset of the variable MINUTE in SI register.
7. Move the contents stored in CL register to the location in SI register.
8. Move the offset of the variable SECOND in SI register.
9. Move the contents stored in DH register to the location in SI register.
10. Move the hexadecimal value 4C into AH register. INT 21H means invoke the interrupt identified by the hexadecimal number 21. In MS-DOS, invoking interrupt 21h while AH = 4Ch causes the current process to terminate and uses the value of register AL as the exit code of the process.

Program:

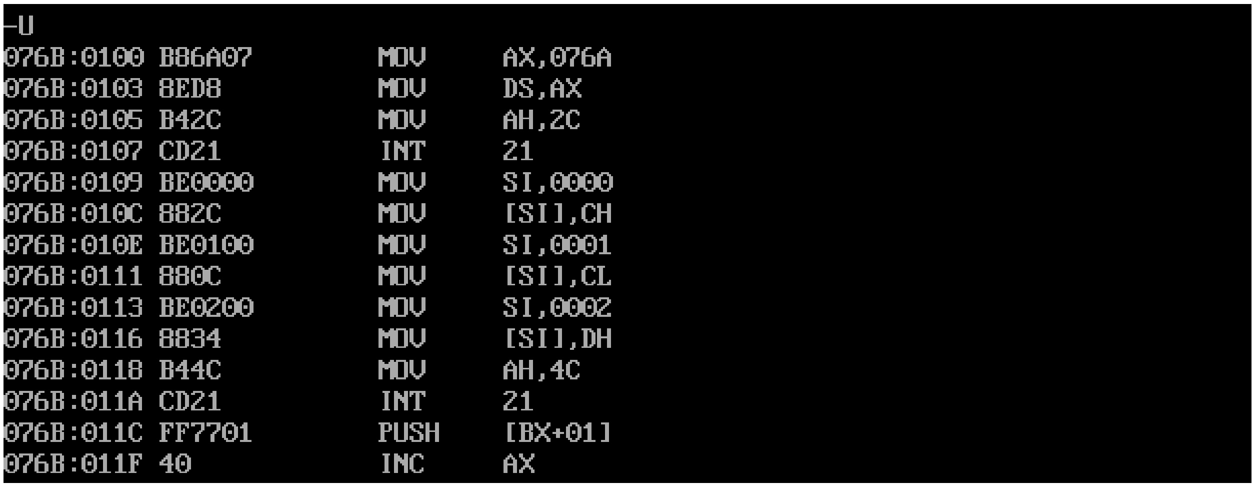


```
assume cs:code,ds:data
data segment
        hour db 01 dup(?)
        minute db 01 dup(?)
        second db 02 dup(?)
data ends

code segment
        org 0100h
start:  mov ax,data
        mov ds,ax
        ; INT 21h/AH=2Ch- get system time
        ;return:CH= hour. CL= minute. DH= second

        mov ah,2ch
        int 21h

        mov si,offset hour
        mov [si],ch
        mov si,offset minute
        mov [si],cl
        mov si,offset second
        mov [si],dh
        mov ah,4ch
        int 21h
code ends
end start
```

	Program	**Comments**
START:	MOV AX, DATA MOV DS, AX	Transferring the data from DATA to AX register and from AX register to DS register.
	MOV AH, 2CH	AH <- 2CH
	INT 21H	INT 21h /AH=2Ch - get system date
	MOV SI, OFFSET HOUR	SI <- HOUR
	MOV [SI], DL	[SI] <- CH
	MOV SI, OFFSET MINUTE	SI <- MINUTE
	MOV [SI], DH	[SI] <- CL
	MOV SI, OFFSET SECOND	SI <- SECOND
	MOV [SI], CX	[SI] <- DH
	MOV AH,4CH	Setup function-4C of the int21.

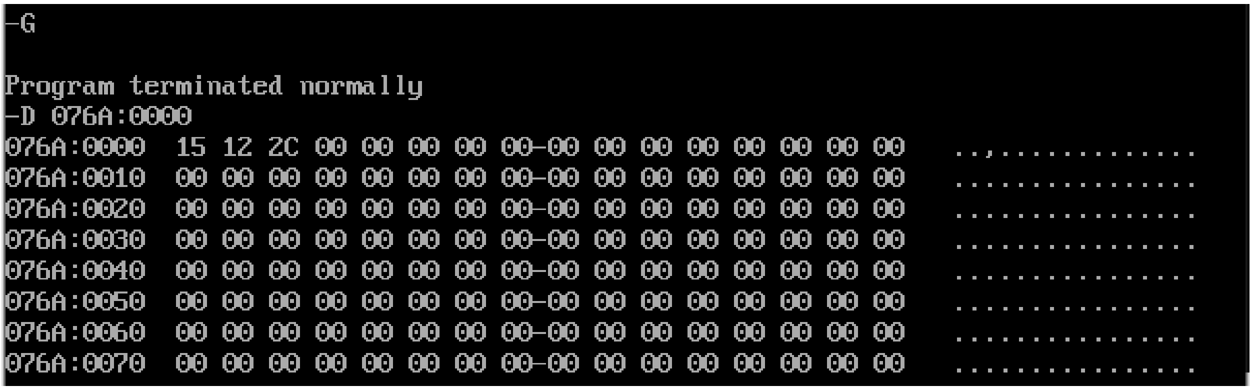
	INT 21H	Call BIOS int21 to return to DOS.

Unassembled Code:



```
-U
076B:0100 B86A07        MOV     AX,076A
076B:0103 8ED8          MOV     DS,AX
076B:0105 B42C          MOV     AH,2C
076B:0107 CD21          INT     21
076B:0109 BE0000        MOV     SI,0000
076B:010C 882C          MOV     [SI],CH
076B:010E BE0100        MOV     SI,0001
076B:0111 880C          MOV     [SI],CL
076B:0113 BE0200        MOV     SI,0002
076B:0116 8834          MOV     [SI],DH
076B:0118 B44C          MOV     AH,4C
076B:011A CD21          INT     21
076B:011C FF7701        PUSH    [BX+01]
076B:011F 40            INC     AX
```

Snapshot of sample input and output:



```
-G

Program terminated normally
-D 076A:0000
076A:0000  15 12 2C 00 00 00 00 00-00 00 00 00 00 00 00 00   ..,.............
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

Result:

Thus the 8086-program for displaying system time is executed successfully in DOS-BOX.