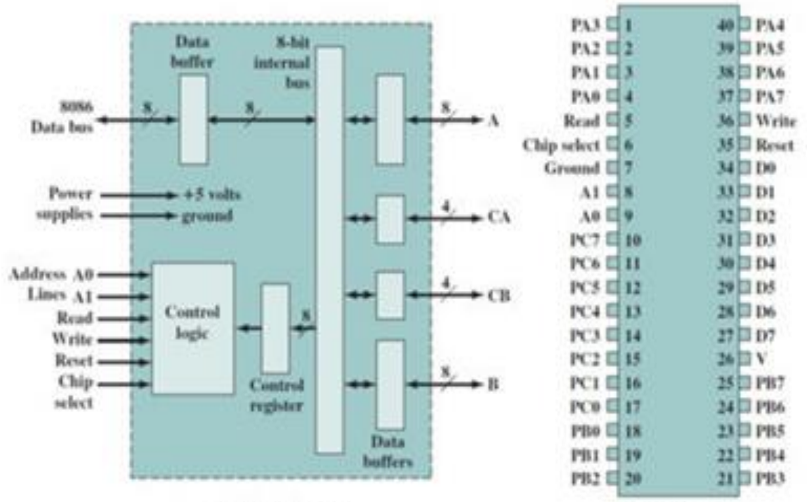
Session 5

Session Objective:

- To get familiar with 8255A Interface, PIN and port configuration.
- To understand the connectivity between 8255 with I/O ports (P3 and P4), LEDs, 7-segments display and DOT Matrix units.
- To simulate an example program to display 0-9 digits in 7-segment display units.
- To simulate an example program to display characters in DOT MATRIX units.

Experiment 1: Introduction to 8255A Programmable Peripheral Interface and Experiment with Seven (7)-Segments Display and LED Connection Program

8255A is a general purpose programmable I/O device used in microprocessors. It consists of three 8-bit bidirectional I/O ports with 24 I/O pins (figure 1) which may be individually programmed in 2 groups of 12 pins and used in 3 major modes of operation.



(a) Block diagram

(b) Pin layout

Figure 5.1: 8255 Block Diagram and PIN OUT Diagram

Carry control and status signal

A1	A0	Function
0	0	Port A
0	1	Port B
1	0	Port C
1	1	Command Register

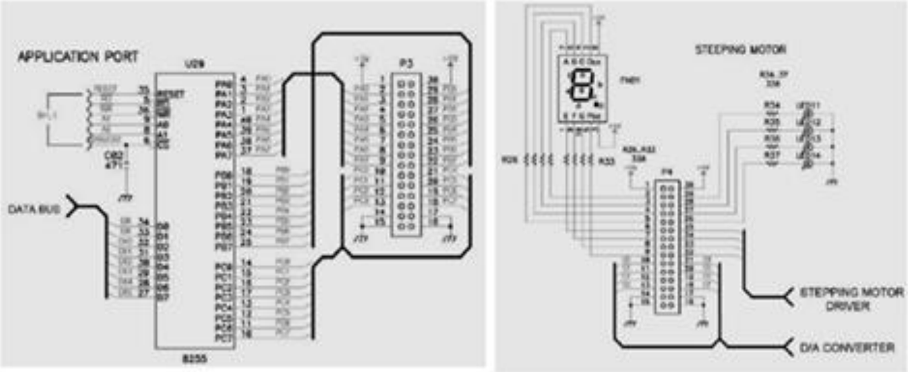


Figure 5.2: Interfacing 8255 with Seven Segment and LEDs

82C55 has three mode of operation including Mode 0, 1, 2.

Mode 0- Basic Input/Output Mode

Causes 82C55 to function either as a buffered input device the pins of Group B/Group A to be programmed as simple I/O ports.

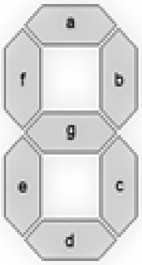
Mode 1- Strobe Input/Output Mode

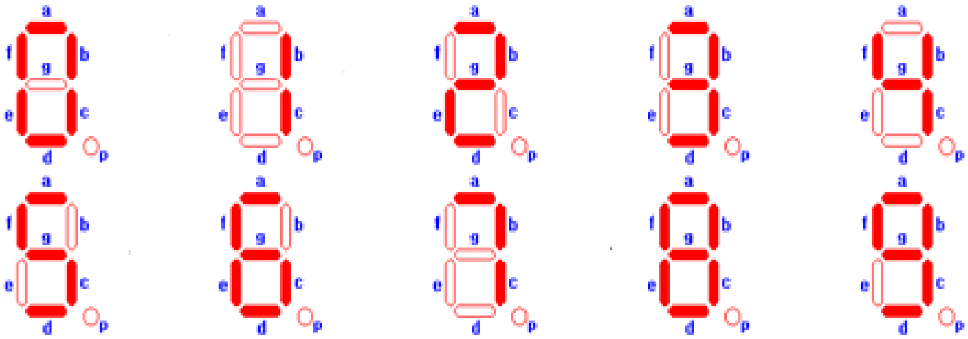
Causes operation port A and/or port B to function as latching input devices. Similar to mode 0 but data are transferred through port A/port B and handshaking (DATA READY, ACKNOWLEDGE) and interrupt request signals are provided by port C. Strobe inputs signal to microprocessor retrieve data that are stored into the port registers

The address of the control register, port A, port B and port C are given below:

```
PPIC_C    EQU    1FH
PPIC      EQU    1DH
PPIB      EQU    1BH
PPIA      EQU    19H
```

Experiment 2: Write an assembly code to display 0-9 in Seven Segment Display (SSD)





- For seven segments display we use 0 for ON and 1 for OFF.
- Control register values will be the column headings of the following table:

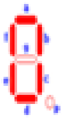
D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	0	0	0
Control Register 0- BSR mode 1- I/O mode	Mode selection for group A 00- I/O 01- Handshaking		Port A 0- Output 1- Input	Upper 4 bit of port C	Mode selection for group B 0- I/O 1- Handshaking	For port B	For lower 4 bit of port C

Assembly Code:

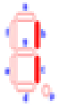
```
S SEGMENT PARA PUBLIC 'CODE'
ASSUME CS: S
ORG 1000H

START:
  ;control register turn on
   MOV AL,80H OUT 1FH,AL
SSD:
    ;display 0
    MOV AL,0C0H
    OUT 19H,AL
    ;for delay
    MOV CX,0FFFFH
    L0:LOOP L0
    ;display 1
    MOV AL,0F9H
    OUT 19H,AL
    ;for delay
    MOV CX,0FFFFH
    L1:LOOP L1
    ;display 2
    MOV AL,0A4H
    OUT 19H,AL
    ;for delay
    MOV CX,0FFFFH
    L2:LOOP L2
    ;display 3
    MOV AL,0B0H
    OUT 19H,AL
    ;for delay
    MOV CX,0FFFFH
    L3:LOOP L3
    ;display 4
    MOV AL,099H
    OUT 19H,AL
    ;for delay
    MOV CX,0FFFFH
    L4:LOOP L4
    ;display 5
    MOV AL,092H
    OUT 19H,AL
    ;for delay
```

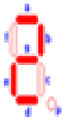
	g	f	e	d	c	b	a
1	1	0	0	0	0	0	0



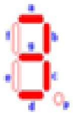
	g	f	e	d	c	b	a
1	1	1	1	1	0	0	1



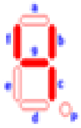
	g	f	e	d	c	b	a
1	0	1	0	0	1	0	0



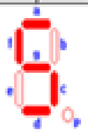
	g	f	e	d	c	b	a
1	0	1	1	0	0	0	0



	g	f	e	d	c	b	a
1	0	0	1	1	0	0	1

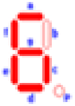


	g	f	e	d	c	b	a
1	0	0	1	0	0	1	0

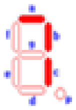


```
  MOV CX,0FFFFH
  L5:LOOP L5
  ;display 6
  MOV AL,082H
  OUT 19H,AL
  ;for delay
  MOV CX,0FFFFH
  L6:LOOP L6
  ;display 7
  MOV AL,0F8H
  OUT 19H,AL
  ;for delay
  MOV CX,0FFFFH
  L7:LOOP L7
  ;display 8
  MOV AL,080H
  OUT 19H,AL
  ;for delay
  MOV CX,0FFFFH
  L8:LOOP L8
  ;display 9
  MOV AL,090H
  OUT 19H,AL
  ;for delay
  MOV CX,0FFFFH
  L9:LOOP L9
  JMP SSD
S ENDS
END START
```

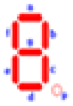
	g	f	e	d	c	b	a
1	0	0	0	0	0	1	0



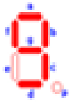
	g	f	e	d	c	b	a
1	1	1	1	1	0	0	0



	g	f	e	d	c	b	a
1	0	0	0	0	0	0	0



	g	f	e	d	c	b	a
1	0	0	1	0	0	0	0



Steps to Run code in MDA-8086 through PC:

- At first copy paste the .ASM file in the mda folder of computer
- Then open cmd and write cd\ and press enter
- Then type cd mda and press enter
- Then type MASM and press enter
- Then write the file_name.ASM and press enter. For our example we will write S.ASM
- Then write the file_name.OBJ and press enter. For our example we will write S.OBJ
- Then write the file_name.LST and press enter. This step is used for error checking. For our example we will write S.LST
- Then when it wants .CRF file simply press enter
- If there is any error in the file, then after this line we can see the number of errors.
- If any error is found, then type EDIT file_name.LST and press enter.
- If no error is found, then type LOD186 and press enter
- Then type file_name.OBJ and press enter. For our example we will write S.OBJ
- Then type file_name.ABS and press enter. For our example we will write S.ABS
- Then type COMM and press enter.
- Then a blue window will occur
- We will now turn on the kit and we will select PC mode from kit mode

- Then press RESET
- If your kit is ok, then it will show up in the blue screen
- Then type L from keyboard and press enter
- If L does not show up, then it means your PC is not connected and you have to try in different PC
- Otherwise press F3 and in the pop-up screen write filename.ABS and press enter. For our example we will write S.ABS
- Then in the kit select kit mode from PC mode
- Then press RESET
- After that press AD
- Then Press GO
- Then you can see the output in the seven segments display

Example 1: Display digits 0–9 and some characters on a 7-segment display (Kit mode).

To display the digits 0 – 9 the bit-patterns are given below.

```
P G F E D C B A
1 1 0 0 0 0 0 0 B
1 1 1 1 1 0 0 1 B
1 0 1 0 0 1 0 0 B
1 0 1 1 0 0 0 0 B
1 0 0 1 1 0 0 1 B
1 0 0 1 0 0 1 0 B
1 0 0 0 0 0 1 0 B
1 1 1 1 1 0 0 0 B
1 0 0 0 0 0 0 0 B
1 0 0 1 0 0 0 0 B
```

<u>Task 1: Write and run the following program to display 0-9</u>

```
B0 80        MOV AL, 10000000B
E6 1F        OUT 1F, AL          ;GOES TO CONTROL REG
B0 F0        MOV AL,11110000B
E6 1B        OUT 1B, AL          ;GOES TO PORT B
B0 00        MOV AL,00000000B
E6 1D        OUT 1D,AL           ; GOES TO PORT C

;DISPLAY STARTS HERE
LEVEL:  ;should be 100CH if the offset address of the code is 1000H
B0 C0        MOV AL, 11000000B
E6 19        OUT 19,AL           ; '0' GOES TO PORT A
B0 F9        MOV AL, 11111001B
E6 19        OUT 19,AL           ;'1'  GOES TO PORT A
B0 A4        MOV AL, 10100100B
E6 19        OUT 19,AL           ; '2' GOES TO PORT A
B0 B0        MOV AL, 10110000B
E6 19        OUT 19,AL           ; '3' GOES TO PORT A
B0 99        MOV AL, 10011001B
E6 19        OUT 19,AL           ; '4' GOES TO PORT A
B0 92        MOV AL, 10010010B
E6 19        OUT 19,AL           ; '5' GOES TO PORT A
B0 82        MOV AL, 10000010B
E6 19        OUT 19,AL           ; '6' GOES TO PORT A
```

```
B0 F8           MOV AL, 11111000B
E6 19           OUT 19,AL           ; ‘7’ GOES TO PORT A
B0 80           MOV AL, 10000000B
E6 19           OUT 19,AL           ; ‘8’ GOES TO PORT A
B0 90           MOV AL, 10010000B
E6 19           OUT 19,AL           ;  ‘9’ GOES TO PORT A

EA  0C                                      ; GIVE THE ADDRESS OF THE INSTRUCTION
MOV AL, 11000000B
10 00           JMP [LEVEL]
00
```

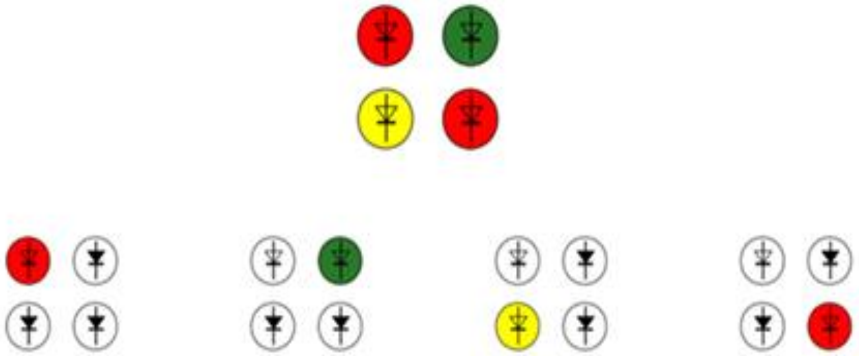
Assignment 1: Write a program to display the hexadecimal digits.

Assignment 2: Write a program to glow the four LEDs of Figure 2 one by one.

Hints: You need to change command register to get the LED on.

```
B0 0F           MOV AL, 00001111B  ; B NEED TO BE WORK AS OUTPUT
E6 1B           OUT 1B, AL          ;GOES TO CONTROL REG
```

Experiment 3: Write an assembly code to glow R1, G, Y and R2 in LED Display respectively.



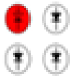
- For LED display we use 1 for ON and 0 for OFF
- Control register value will be the column headings of the following table:

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	0	0	0
Control Register 0- BSR mode 1- I/O mode	Mode selection for group A 00- I/O 01- Handshaking		Port A 0- Output 1- Input	Upper 4 bit of port C	Mode selection for group B 0- I/O 1- Handshaking	For port B	For lower 4 bit of port C

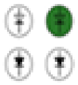
Assembly Code:

```
L SEGMENT PARA PUBLIC 'CODE'
ASSUME CS: L
ORG 1000H
START:
;control register turn on
MOV AL,80H
OUT 1FH,AL
;segment address forcefully off
MOV AL,0FFH
OUT 19H,AL
LED:
;R1 LED turn on
MOV AL,01H
OUT 1BH,AL
;for delay
MOV CX,0FFFFH
LR1:LOOP LR1
;G LED turn on
MOV AL,02H
OUT 1BH,AL
;for delay
MOV CX,0FFFFH
LG:LOOP LG
;Y LED turn on
MOV AL,04H
OUT 1BH,AL
;for delay
MOV CX,0FFFFH
LY:LOOP LY
;R2 LED turn on
MOV AL,08H
OUT 1BH,AL
;for delay
MOV CX,0FFFFH
LR2:LOOP LR2
JMP LED
L ENDS
END START
```

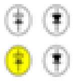
				R2	Y	G	R1
0	0	0	0	0	0	0	1



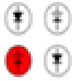
				R2	Y	G	R1
0	0	0	0	0	0	1	0



				R2	Y	G	R1
0	0	0	0	0	1	0	0



				R2	Y	G	R1
0	0	0	0	1	0	0	0



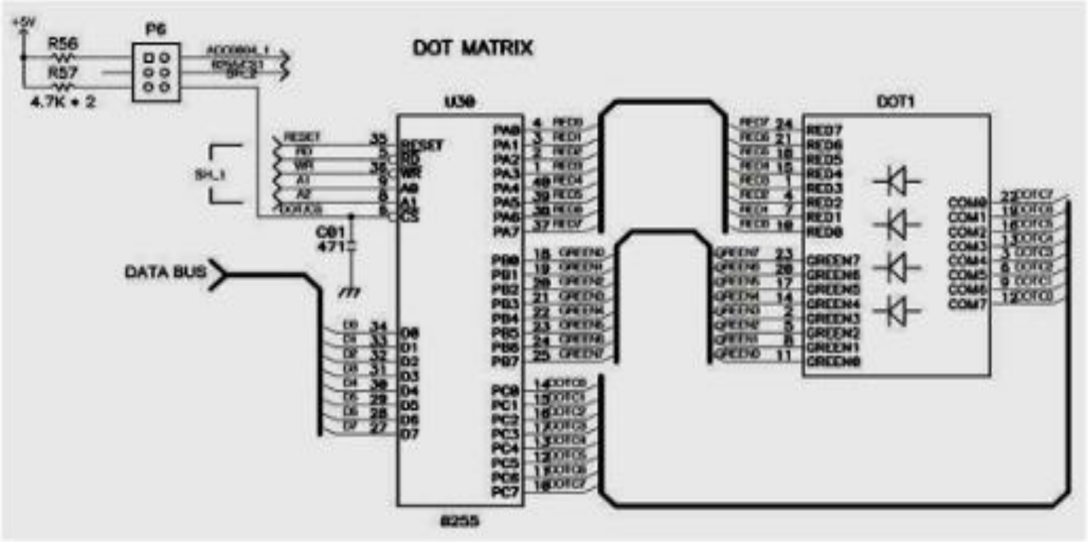
Steps to run code in MDA-8086 through PC:

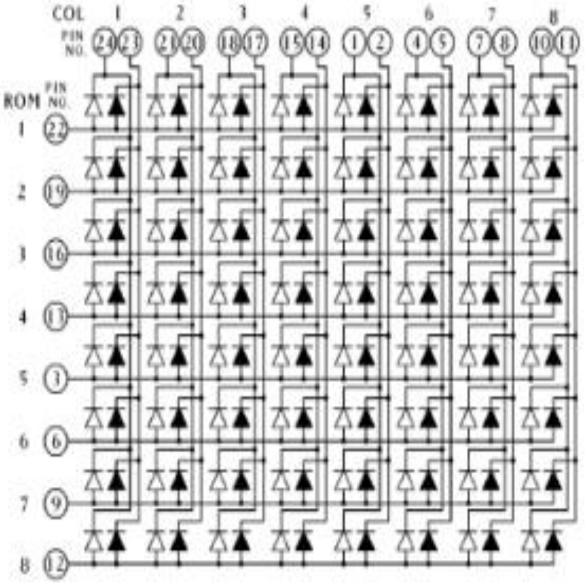
- At first copy paste the .ASM file in the mda folder of computer
- Then open cmd and write cd\ and press enter
- Then type cd mda and press enter
- Then type MASM and press enter
- Then write the file_name.ASM and press enter. For our example we will write L.ASM
- Then write the file_name.OBJ and press enter. For our example we will write L.OBJ

- Then write the file_name. LST and press enter. This step is used for error checking. For our example we will write L.LST
- Then when it wants. CRF file simply press enter
- If there is any error in the file, then after this line we can see the number of errors.
- If any error is found, then type EDIT file_name.LST and press enter.
- If no error is found, then type LOD186 and press enter
- Then type file_name.OBJ and press enter. For our example we will write L.OBJ
- Then type file_name. ABS and press enter. For our example we will write L.ABS
- Then type COMM and press enter.
- Then a blue window will occur
- We will now turn on the kit and we will select PC mode from kit mode
- Then press RESET
- If your kit is ok, then it will show up in the blue screen
- Then type L from keyboard and press enter
- If L does not show up, then it means your PC is not connected and you have to try in different PC
- Otherwise press F3 and in the pop-up screen write filename.ABS and press enter. For our example we will write L.ABS
- Then in the kit select kit mode from PC mode
- Then press RESET
- After that press AD
- Then Press GO
- Then you can see the output in the LED display

Experiment 4: Dot Matrix Display with The Microprocessor Through Peripheral Programmable Interface 82555a

8255A is a general purpose programmable I/O device used in microprocessors. It consists of three 8-bit bidirectional I/O ports with 24 I/O pins (figure 1) which may be individually programmed in 2 groups of 12 pins and used in 3 major modes of operation.





To formulate (8x8) DOTT MATRIX – two (2) color LEDs – including RED and GREEN. Port C HIGH (1) and Port A is LOW (0) glows corresponding RED LED, Port C HIGH (1) and Port B is LOW (0) glows corresponding GRREEN LED.

The address of the control register, port A, port B and port C of the 8255 IC are 1E, 18, 1A and 1C respectively.

Figure 5.3: Circuit diagram of a DOT MATRIX

Task : Understand the basic configuration of 8255 and the DOT matrix.

LEDs are a particular type of diode that converts electrical energy into light. In fact, LED stands for "Light Emitting Diode". The following figure is taken from TechTerms.com and shows the basic forward bias electricity flow from Anode (Positive +) to Cathode (Negative -). If both side of diode have same voltage value (1/0) then no conduction, means no current. However, if potential difference is equal to or greater than threshold (0.7 for germanium) then there will be conduction.

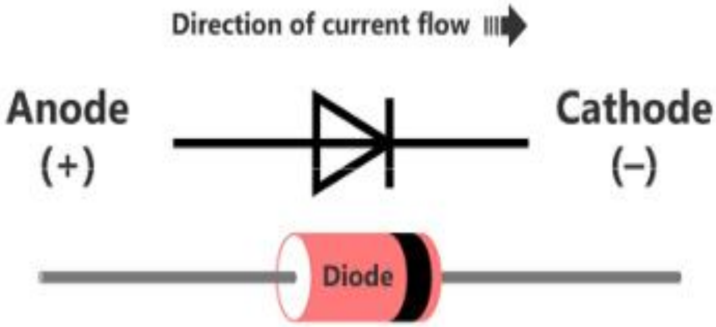


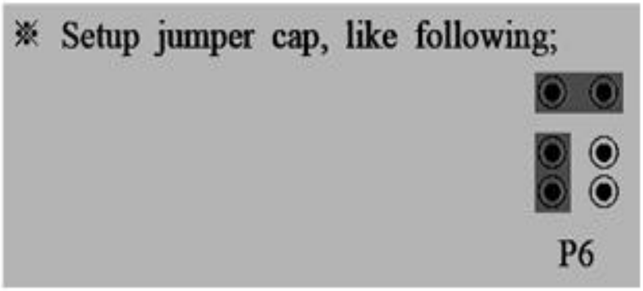
Figure 5.4: Direction of Current Flow

For LEDs in MATRIX: Glow 1st ROW LEDs

You will need to select the column A0/B0 (which means A0/B0 is pulled low), and deselect other columns by blocking their ground paths (by pulling A1/B1 through A7/B7 pins to logic high). Now, the first column is active, and you will need to turn on the LEDs in the rows C0 through C7 of this column, which can be done by applying forward bias voltages (HIGH) to all rows.

Jumper Setup:

You need to set the jumper as shown below before running any program with MDA kit.


※ Setup jumper cap, like following;

Task: Understand the Setting of the Jumper.

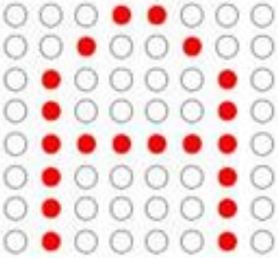
Go to top right corner of DOT MATRIX figure.

Experimental tools:

MDA-Win8086, Computer, Microprocessor emulator Software with Integrated Assembler.

The following rules are needed to perform the lab work.

Task: Run the HEX CODE of the following program to display the letter 'A' on the LED matrix.



```
            ; PPIC_C      EQU   1EH    ; control register
            ; PPIC EQU    1CH    ; c port
            ; PPIB EQU    1AH    ; b port
            ; PPIA EQU    18H    ; a port

Address: Op code      instructions

                      ORG   1000H
1000:  B0 80          MOV   AL, 10000000B
1002:  E6 1E          OUT   PPIC_C, AL    ; program PPI
                                  ;
1004:  B0 FF          MOV   AL, 11111111B        ; OFF LEDs connected to port A
1006:  E6 18          OUT   PPIA, AL
                                  ;
1008: BE 2C 10    L1:       MOV  SI, OFFSET FONT
100B: B4 01           MOV   AH, 00000001B
100D: 2E 8A 04   L2:MOV   AL, BYTE PTR CS:[SI]
1010:  E6 1A          OUT   PPIB, AL
```

```
1012:  8A C4          MOV   AL, AH
1014:  E6 1C          OUT   PPIC, AL
1016:  E8 09 00       CALL TIMER

1019:  46             INC   SI
101A:  F8             CLC
101B:  D0 C4          ROL   AH, 1
101D:  73 EF(EE)      JNC   L2
101F:  EB E7          JMP   L1
1021:  CC             INT   3
1022:  B9 2C 01       TIMER:      MOV  CX, 300
1025:  90             TIMER1: NOP
1026:  90                   NOP
1027:  90                   NOP
1028:  90                   NOP
1029:  E2 FA                LOOP TIMER1
102B:  C3                   RET
                                  ;
102C:  FFFONT: DB           11111111B
102D:  C0             DB    11000000B
102E:  B7             DB10110111B
102F:  77             DB01110111B
1030:  77             DB01110111B
1031:  B7             DB10110111B

1032:  C0             DB    11000000B
1033:  FF             DB    11111111B
```