

**Faculty of Information Technology**

**Computer Systems Engineering Department**

**COMPUTER DESIGN LAB #ENCS411**

**EXP #2**

 Control of Dot Matrix and 7-Segment MDA-8086 Kit – PPI Application

**Name : Mohammad Modallal**

**ID : 1120174**

**Section# : 3**

**Dr.** [**Khader Mohammad**](https://ritaj.birzeit.edu/student/view_office_hr?employee_id=5442)

**Date:2/3/2015**

**Abstract:**

The object of this experiment is to understaning and testing the 8x8 LEDs dot matrix structure and the single 7-segment display using the 82C55 PPI (Programmable Peripheral Interface).

I used the dot matrix to display a beam of 8-lit LEDs scoring from left to right of the dot matrix (i.e. moving a column horizontally),then i used it to display a beam of 8-lit LEDs scoring from top to bottom of the dot matrix (i.e. moving a row down).Finally ,I use the dot matrix to display character ‘X’.

I used the single 7-segment display to the counter that will count from 0 to 9 and wraps back to 0.

**1.Introduction:**

* Dot matrix:

The dot matrix is a 2-dimensional patterned [array](http://en.wikipedia.org/wiki/Array_data_structure) consist of 8x8 leds,which lighted by red , green and orange color .I can control it by using all 82C55 PPI ports. PortA drives the green LEDs while portB drives the red LEDs. If the two LEDs with the same coordinates are enabled then the outcome will be an orange color.

 As shown in figure below[figure1.1], ports A and B select the rows while port C

 select the columns.



To display the LED[figure 1.2] the voltage on anode (port C) must be more than the voltage on cathode(port A or B),so if we want to display green LEDS we send one’s to port c (Anode logic 1) of the PPI and zero’s to Port A of the PPI (cathode logic 0*)* and we send one’s to port B.

****

The table below show the value of ports of PPI during different color display.

|  |  |
| --- | --- |
| *Port* | *Port value* |
| *A* | *00H* |
| *B* | *FFH* |
| *C* | *FFH* |

Table 1.2 display green leds

|  |  |
| --- | --- |
| *Port* | *Port value* |
| *A* | *FFH* |
| *B* | *00H* |
| *C* | *FFH* |

Table 1.3 display red leds

|  |  |
| --- | --- |
| *Port* | *Port value* |
| *A* | *00H* |
| *B* | *FFH* |
| *C* | *00H* |

Table 1.1 display green leds

|  |  |
| --- | --- |
| *Port* | *Port value* |
| *A* | *FFH* |
| *B* | *FFH* |
| *C* | *FFH or 00H* |

Table 1.4 leds OFF

In our experiment, the dot matrix is interfaced with the PPI. So the addresses of PPI ports are shown in the table 1.5.

|  |  |
| --- | --- |
|  | *address* |
| *Port A* | *18H* |
| *Port B* | *1AH* |
| *Port C* | *1CH* |
| *Control register* | *1EH* |

*Table 1.5*

* **Seven Segment Display:**

A seven-segment display (SSD) is a form of electronic [display device](http://en.wikipedia.org/wiki/Display_device) for displaying [decimal](http://en.wikipedia.org/wiki/Decimal) [numerals](http://en.wikipedia.org/wiki/Numeral_system) that is an alternative to the more complex [dot matrix](http://en.wikipedia.org/wiki/Dot_matrix) displays[2]. The display consists of seven active low segments (A, B, C, D, E, F, G) [figure 1.3]. The segments are driven by the PPI Port A signals as follow:

|  |  |
| --- | --- |
| *Segment* | *82C55 Port-A*  |
| *A* | *PA0* |
| *B* | *PA1* |
| *C* | *PA2* |
| *D* | *PA3* |
| *E* | *PA4* |
| *F* | *PA5* |
| *G* | *PA6* |
| *P* | *PA7* |

*Table 1.6*



In our experiment, the 7-segment is interfaced with the PPI. So the addresses of PPI ports are shown in the table 1.7.

|  |  |
| --- | --- |
|  | *address* |
| *Port A* | *19H* |
| *Port B* | *1BH* |
| *Port C* | *1DH* |
| *Control register* | *1FH* |

*Table 1.7*

**2.Procedure:**

**2.1 Task one:**

Using the dot matrix to display a beam of 8-lit LEDs scoring from left to right of the dot matrix (i.e. moving a column horizontally).

**2.1.1** First of all, we configure the P6 connector using two jumper caps as shown in figure 2.1.

**

Figure 2.1

**2.1.2** Then we write the following assembly code and store it into the file name MATRIX\_1.ASM .We run it using Kit MDA-8086 and WinComm software.



**2.1.3 The result will be as shown in figure 2.2(red color)**

****

We done this by sending zero`s to port B(LEDs display red color) and sending one`s to port A.we using loop to send one`s (logic 1) to one of port C pins[PC0-PC7] each time which lead to lighted one column and turn of the others .we use function TIMER to delay the display to be visible to human.

**2.2 Task two:**

Using the dot matrix to display a beam of 8-lit LEDs scoring from top to bottom of the dot matrix (i.e. moving a row down)

**2.2.1** we write the following assembly code and store it into the file name MATRIX\_2.ASM. We run it using Kit MDA-8086 and WinComm software.



We send zero`s to port B to display LEDS red and we send one`s to port A to turn off green LEDS.we send ones`s to port C. we using loop to send one`s (logic 1) to one of port C pins[PC0-PC7] each time which lead to lighted one column and turn of the others.

**2.3 Task three:**

Using dot matrix to display character ‘X’.

**2.3.1** we write the following assembly code and store it into the file name MATRIX\_3.ASM. We run it using Kit MDA-8086 and WinComm software.

****

**2.3.2 The result will be as shown in figure 2.3(red color)**

****

Figure2.3

**2.4 Task four**:

useing the single 7-segment display to the counter that will count from 0 to 9 and wraps back to 0.

**2.4.1** we write the following assembly code and store it into the file name MATRIX\_4.ASM. We run it using Kit MDA-8086 and WinComm software.



**2.4.2 The result will be as shown in figure 2.4**

figure 2.4

**3.Conclusion:**

In this experiment I learn many things such as:

1. I become familiar with MDA-8086 kit and Webcomm software.
2. I understand functionality of Dot matrix and 7-segmant display.
3. I learned how to make .hex file using CMD Command line which is used to loaded in MDA kit to be executed[file.asm file.exe file.bin file.hex].
4. I study some application for PPI such as DOTmatrix and 7-sigmant.

**Reference:**

[1]THE INTEL MICROPROCESSORS,seven edition.

[2] <http://en.wikipedia.org/wiki/Seven-segment_display>.