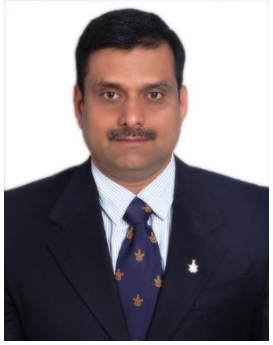
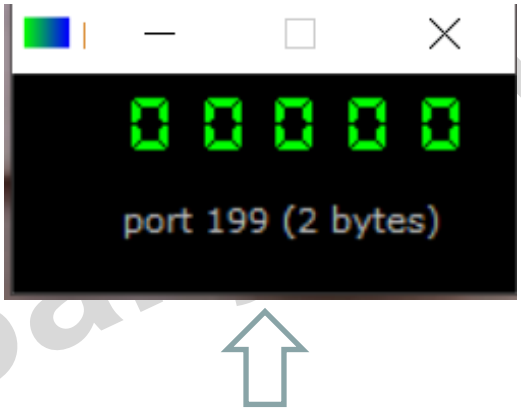
Microprocessors and Interfaces: 2021-22
Lab 11
Program to Display numbers in LED Display

By Dr. Sanjay Vidhyadharan



LED Display in Emulator

- A LED display is available in EMU8086 with port address 199.
- It consists of 5 LED based display which can be used to show a number (both positive and negative).
- The display number can be provided using any 8086 ALP or using a simple loop.



Port Address

Display number using LED array

ALP for LED based Display

```
#start=led_display.exe#
#make_bin#
name "led"

mov ax,0
out 199,ax

mov ax, zzzzz
out 199, ax

mov ax, vwxyz
out 199, ax

Eternal loop to write values to port:

mov ax, yy
x1:
  out 199, yy
  inc zz
jmp yy

hlt
```

Initializes the virtual LED display

Load 0 in the display

Store an arbitrary positive number in AX and send it IO

Store an arbitrary negative number in AX and send it IO

Initialize AX by 0.

Start from 0. Each step increases one value and load the updated number in the LED display. Continue to increase the value.

The output file type directives

#make_com# - the oldest and the simplest format of an executable file, such files are loaded with 100h prefix (256 bytes). Select **Clean** from the **New** menu if you plan to compile a COM file. Compiler directive **ORG 100h** should be added before the code. Execution always starts from the first byte of the file. This file type is selected automatically if **org 100h** directive is found in the code. supported by DOS and Windows Command Prompt.

#make_exe# - more advanced format of an executable file. not limited by size and number of segments. stack segment should be defined in the program. you may select **exe template** from the **new** menu in to create a simple exe program with pre-defined data, stack, and code segments. the entry point (where execution starts) is defined by a programmer. this file type is selected automatically if **stack** segment is found. supported by dos and windows command prompt.

The output file type directives

#make_bin# - a simple executable file. You can define the values of all registers, segment and offset for memory area where this file will be loaded. When loading "**MY.BIN**" file to emulator it will look for a "**MY.BINF**" file, and load "**MY.BIN**" file to location specified in "**MY.BINF**" file, registers are also set using information in that file (open this file in a text editor to edit or investigate). in case the emulator is not able to find "**MY.BINF**" file, current register values are used and "**MY.BIN**" file is loaded at current **CS:IP**. the execution starts from values in **CS:IP**. bin file type is not unique to the emulator, however the directives are unique and will not work if .bin file is executed outside of the emulator because their output is stored in a separate file independently from pure binary code.

.BINF file is created automatically if assembler finds any of the following directives.

Thankyou