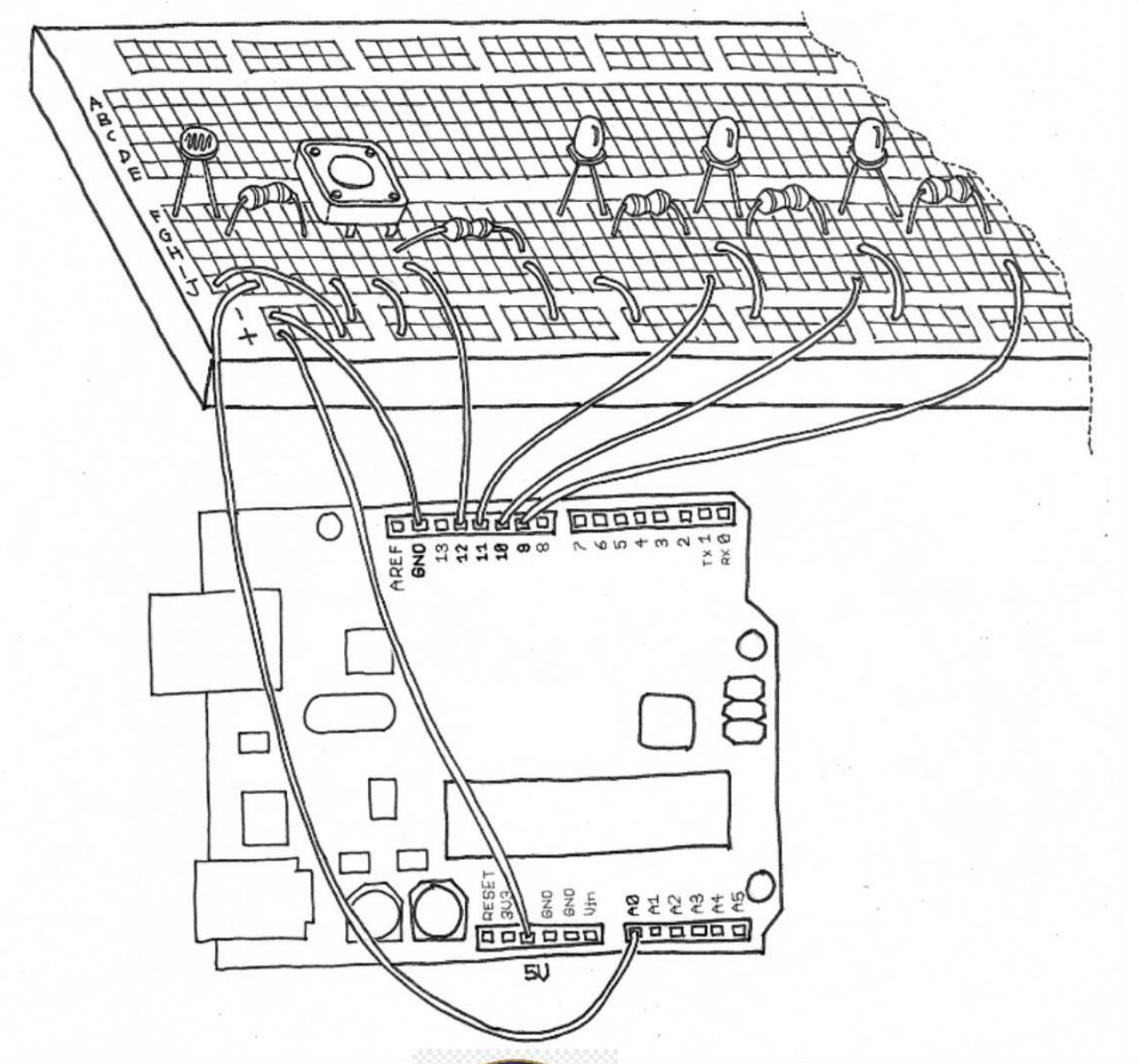
CS/EEE/INSTR F241
Microprocessor Programming and Interfacing

Lab 1- Introduction to DEBUG & DEBUGX



Dr. Vinay Chamola and Anubhav Elhence

Lab 1- Introduction to DEBUG & DEBUGX

What is DebugX?

DEBUG, supplied by MS-DOS, is a program that traces the 8086 instructions. Using DEBUG, you can easily enter 8086 machine code program into memory and save it as an executable MS-DOS file (in .COM/.EXE format). DEBUG can also be used to test and debug 8086 and 8088 programs. The features include examining and changing memory and register contents including the CPU register. The programs may also be single-stepped or run at full speed to a break point.

You will be using DEBUGX which is a program similar to DEBUG but offers full support for 32-bit instructions and addressing. DEBUGX includes the 80x86 instructions through the Pentium instructions.

Installing DosBox and DebugX

To install the software which are prerequisites for x86 programming, follow the link below and these steps:-.

Link:- BITS IoT Lab Download Link

[For Windows]

1. On the above page, click on "*Link to DosBox*" installer, and install DosBox (There should be an icon on the desktop after this).
2. Now, click on the "*Link to MASM*", and extract the MASM611 folder.
3. Copy the MASM611 folder directly in the 'D' drive (Remember the drive, this will be important later)
4. Make sure "debugx" is present in the MASM611→BIN folder.
5. You don't need to install anything from/within MASM611 folder (Everything is already pre-compiled in the folder)

How to Start DebugX

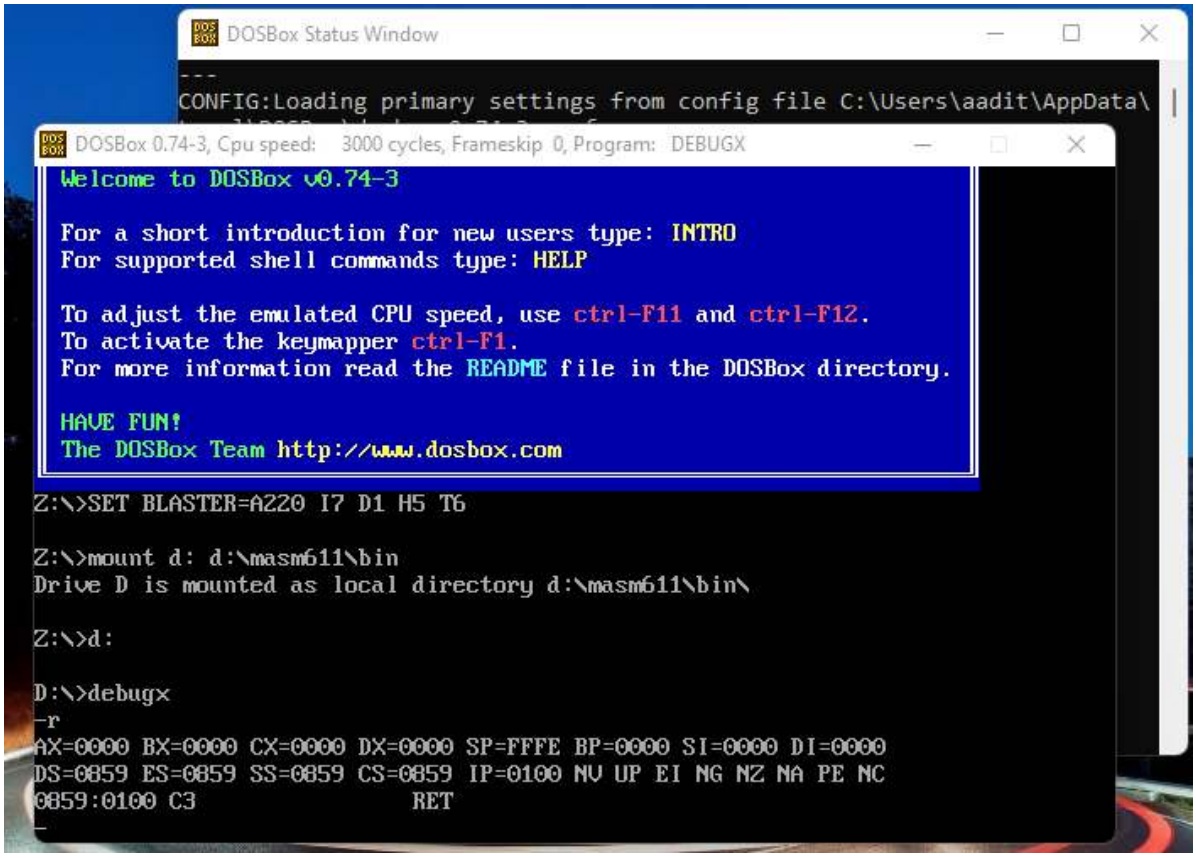
Once you have all the software installed, this is the procedure you'll need to follow to start DebugX. In short, we need to mount the drive where the MASM611 folder is stored (D Drive here). *You will have to do this many times, please note it down*:-

1. Click on the **DosBox icon** on your desktop (This opens two windows, you only need to focus on the one with the command prompt).
2. The command prompt will read 'Z:\>', type in "**mount d: d:\masm611\bin**" (If successful you should get a message 'Drive D is mounted as local directory')
3. Now type "**d:**" to change the command prompt to 'D:\>'
4. Type "**debugx**" to start the program (If succsessful, the cursor moving to the next line.)
5. You can return to DosBox, by typing "q" (Quit command)

In Summary (See figure below):-

- **DOSBox icon**
- **mount d: d:\masm611\bin**
- **d:**
- **debugx**

Great! You are ready to go. To check if everything worked, type r (Register command) in DEBUGX to see all the registers.



DebugX Basic Conventions

- **DebugX is NOT case sensitive.**
- **DEBUGX recognizes ONLY Hexadecimal numbers (without a trailing 'H' by default, so '11' is interpreted as seventeen, not eleven!!)**
- **DEBUGX displays a list of the commands and their parameters by entering a "?" at the command prompt.**
- **Segment and offset are specified as Segment:Offset**
- **Spaces in commands are only used to indicate separate parameters**

List of DebugX Commands

These are some vital commands you will use in DEBUGX, to perform tasks like viewing the registers, to executing and verifying your assembling language programs.

We recommend you to try out these commands, in the following tasks, to get accustomed to using DEBUGX. More details on the commands will be shared in the upcoming lab sessions.

Command Syntax	Description
Register	
RX	Activates 32-bit registers ('386 regs on')
R	Shows the 16-bit registers (Default) or the 32-bit registers, if rx was used before
R <register>	**View a register** and change its value at the prompt
Execution	
A :<offset>	Assemble- Prompts the code segment to **write instructions** (assembled into machine code)
U <offset>	Unassemble- **Displays the Symbolic code** (instructions) written at the offset (from CS)
T	Trace- Execute commands at CS:IP, i.e. **one at a time** (debugging)
G <address of last instruction>	Go- Executes commands **all at once**, until the address specificed (Change IP Value using R first!)
Data	
D :<offset>	Dump- **View the data** at this address (Little Endian)
E :<offset>	Enter- **Edit data** at this address by changing the value at the prompt
Misc.	
?	**View** all debugx commands
Q	**Quit** debugx

Vital DEBUG Commands

A: Assemble

The Assemble command (A) is used to enter assembly mode. In assembly mode, DEBUG prompts for each assembly language statement and converts the statement into machine code that is then stored in memory. The optional start address specifies the address at which to assemble the first instruction. The default start address is 100h. A blank line entered at the prompt causes DEBUG to exit assembly mode.

Syntax: A [address]

D: Dump

The Dump command (D), when used without a parameter, causes DEBUG to display the contents of the 128-byte block of memory starting at CS:IP if a target program is loaded, or starting at CS:100h if no target program is loaded. The optional range parameter can be used to specify a starting address, or a starting and ending address, or a starting address and a length. Subsequent Dump commands without a parameter cause DEBUG to display the contents of the 128-byte block of memory following the block displayed by the previous Dump command.

Syntax: D [range]

R: Register

The Register command (R), when used without a parameter, causes DEBUG to display the contents of the target program's CPU registers. The optional register parameter will cause DEBUG to display the contents of the register and prompt for a new value.

Syntax: R [register]

Syntax: R [register] [value]

T: Trace

The Trace command (T), when used without a parameter, causes DEBUG to execute the instruction at CS:IP. Before the instruction is executed, the contents of the register variables are copied to the actual CPU registers. After the instruction has executed, and updated the actual CPU registers (including IP), the contents of the actual CPU registers are copied back to the register variables and the contents of these variables are displayed. The optional address parameter can be used to specify the starting address. The optional count parameter can be used to specify the number of instructions to execute. To differentiate the address parameter from the count parameter, the address parameter must be preceded with an "=". Note that the first byte at the specified address must be the start of a valid instruction.

Syntax: T [=address] [number]

Tasks to be Completed

1. Using three addressing modes (Immediate, Register, Register-Indirect), write instructions to

- **Move the value 1133H into the register AX.**
- **Swap the lower and higher bytes in AX and move them into BX (If AX is pqrs, BX should be rspq)**
- **Move the value in BX to the memory location at an offset of 20 (from BX)**

Note down the machine code equivalents of the four MOV statements.

(*Hint:* You need to use the following commands- A to write the instructions, and U to view the machine code and unassembled instructions, T to execute and D to view the memory location)

2. Move the first letter of your name (ASCII Character) to the location DS:0120
(*Hint*: Recall the rules for the Immediate addressing mode)

3.Fill 32 (decimal) bytes of the Extra Segment with ASCII characters for the first two letters of your name. (Like "ABABAB...")

(*Hint:* Use the F (Fill) command to fill a memory region with a byte pattern

To fill, for example, the first 8000h bytes of the current data segment with pattern 55:

F 0 L 8000 55

[Syntax: F <start-address> L <range> <pattern>])

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

ज्ञानं परमं बलम्