Daffodil International University (DIU)

Department of Electrical and Electronic Engineering

EEE 324: Microprocessors and Interfacing Laboratory

Experiment No: 01

Name of the Experiment:

INTRODUCTION TO EMULATOR8086 AND DATA ADDRESSING MODE.

Objectives:

The objectives of the experiment are-

- To know how to run a program in emulator8086
- To know how to copy data from register to register, memory to register and register to memory.

Requirements:

1. PC with Windows XP.
2. Emulator8086 Software.

Theory:

There are four fields in an instruction:

- Label
- Op-code (Operation code)
- Operands
- Comments

Example: xxx: MOV CX, DX; Copy

Data addressing modes of 8086:

- Register Addressing mode
- Immediate Addressing mode
- Register Indirect Addressing mode
- Direct Addressing mode
- Indexed Addressing mode
- Base Relative Addressing mode
- Base Indexed Addressing mode

Register Addressing Mode:

Transfers a copy of a byte or word from the source register or memory location to the destination register or memory location.
For example:
MOV CL, BL means move 8-bit data from BL to CL.
MOV DX, AX means move 16-bit data from AX to DX register.

Immediate Addressing Mode:

Transfers the source immediate byte or word of data into the destination register or memory location.
For example:
MOV AL, 25H means move 8-bit data 25H into AL register.
MOV CX, 23F6H means move 16-bit data 23F6 into CX register.

Direct addressing mode:

Moves a byte or word between a memory location and a register.
For example:
MOV CL, LIST means the byte sized content of memory location LIST is copied into CL.

Register Indirect addressing mode:

Transfers a byte or word of data between a register and a memory location addressed by an index or base register.
For example:
MOV AX, [BX] copies the word sized content of the data segment offset address indexed by BX into register AX.

Base plus Indexed Addressing mode:

Transfers a byte or word of data between a register and a memory location addressed by a base register (BP or BX) plus an index register (DI or SI).
For example:
MOV [BX+DI], CL copies the byte sized content of register CL into the data segment memory location addressed by BX plus DI.

Register Relative addressing mode:

Moves a byte or word between a register and a memory location addressed by a base register (BP or BX) or an index register (DI or SI) plus a displacement.
For example:
MOV AX, [BX+4] loads AX from the data segment address formed by BX plus 4.

Base Relative plus Index addressing mode:

Transfers a byte or word of data between a register and a memory location addressed by a base register (BP or BX) and an index register (DI or SI) plus a displacement.

For example:

MOV AX, [BX+DI+4] loads AX from the data segment address formed by BX plus DI plus 4.

Procedure:

- **To run an example code follow the following steps-**

1. First of all you need to start emu8086. Double click emu8086 icon or run c:\emu8086\emu8086.exe
   If you don't have this program yet, you can download it from http://www.emu8086.com/

2. When it starts, click code examples and select Hello, world.

3. A code example with a lot of comments should open, all comments are green and they take about 90% of all text, so don't be scared by this tiny "Hello Word" code. The compiled executable has length of just about 100 bytes, this is even less than the size of this sentence.

4. To run this example in the emulator click emulate (or press F5). This makes the assembler to try to assemble and save the executable to c:\emu8086\MyBuild and then, if assembler succeeds making the file, the emulator will automatically load it into the memory.

5. You can click single step (or press F8) to executed the code and see how it works, you can also click step back (or press F6) and see how changes are reversed.

6. Actually, there are far more than a single way to print "Hello World" in assembly language, and this is not the shortest way. Click examples and browse c:\emu8086\examples, there is HelloWorld.asm that is assembled in only 30 bytes, it is so small because, unlike the previous example that does everything itself, the shorter one uses built-in interrupt function of the operating system. Interrupts save a lot of work and time, but unlike the previous example, they often do not let you see how they do what they do.

- **To write a new code follow the following steps-**

1. Start emulator8086.
2. To write a new code click on New and choose code template as empty workspace.
3. Write the code in the text area.
4. Click on File\Save and save it.
5. Click on Compile\Save\Run. After the operation is completed the emulator will display a message.
6. Click on debug and verify the output on debug log.

Program 01:

Write the following code in the text area and verify the output.

```
ORG 1000H
MOV AX, 1223H
MOV BX, 3456H
MOV CX, 6478H
MOV DX, 3386H
HLT
```

OUTPUT:

```
AX = 1223H
BX = 3456H
CX = 6478H
DX = 3386H
```

Program 02:

Now again write the following code in the text area and find out the problems and solve it.

```
ORG 1000H
MOV [AX], 1223H
MOV SS, 3456H
MOV CX, 6478H
MOV DX, 3386H
HLT
```

REPORT:

1. Describe how to write a new program in emulator8086.
2. Describe which instructions are not supported in 8086 microprocessor.