Fall 1997 Professor Yap

V22.0201.002 Machine Organization I

Assignment 1

Due: Wednesday, September 24, 1997

INTRODUCTION Your assignment is to write an assembly language program that reads a sequence of input characters and then echos the sequence in the next line, and also prints the sequence in reverse in the following line. This assignment will familiarize you with the turbo assembler and its computing environment.

DETAILS The overall assignment is worth 30 points. Note that some instructions carry specific points. E.g., You get 2 points just for giving your program the suggested title.

- Make the title of your program "MY_ECHO, HW 1" [2 points].
- Your MY_ECHO program is to first print any reasonable introduction message followed by the prompt ">>" [2 points]. Note: prompts always begin on a new line.
- Subsequently, MY_ECHO reads a sequence of 16 printable ASCII characters from the keyboard. You are not responsible if the user types more than 16 characters.
- Then MY_ECHO echos the input (i.e., prints in the next line the string of 16 characters), and in the following line, MY_ECHO prints the same string but in reverse order.

E.g. Here is a sample run:

```
:> myecho
Welcome to the Echo Program
>> 0123456789ABCDEF
Here is your input:  0123456789ABCDEF
Here is it's reverse:  FEDCBA9876543210
:>
```

Here, :> marks the DOS prompt, and >> marks your prompt. The user input is shown underlined.

TO BE HANDED IN:

- A printed copy of a well-commented program, which should include your identification (NAME AND NYU ID) at the top. [2 points]
- Two sample runs of your program [2 points].
- A 3.5" disk containing the source file and an executable file. Put your name and class identification on the diskette. The source and executable files should be called (respectively) hw1.asm and myecho.exe. [2 points]
- Hand in everything in an envelop with your name on it [2 points]. As always, you are responsible to keep an extra copy of your work.

NOTES AND HINTS

- You must read chapter 4 for this assignment.
- The pseudo-operators `EQU` and `DUP` are useful. [4 points if you use both of them]
- Interrupt 21h, function 9 (p. 73) for printing a string is useful.
- The **based addressing mode** as discussed in class is necessary. E.g.,

  ```
  MOV [Input+BX],AL
  ```

  where `Input` is a variable name. (If you really must, you may refer to p.181 for more details.)

Before doing this exercise, we suggest that you first try out the assembler. Feel free to send me emails me know if you run into snags. Use your favorite editor (e.g., Microsoft `edit`) to input a small program from our lectures or the book (e.g., PGM4_2.ASM on page 74). Then assemble, link and run it. Details for assembling your program and linking it are found chapter 4.9. We recommend that you do this under DOS rather than under Windows. E.g., Say you are using the Turbo Assembler and your program is in the file `hw1.asm`:

```
:> tasm hw1              ; this produces the object file hw1.obj.
:> tlink hw1             ; this converts hw1.obj to hw1.exe.
:> mov hw1.exe myecho.exe
:> myecho                ; this executes the file myecho.exe.
...
```