USERS MANUAL

MEMORY AREA: 0000 - FFFF

Please keep these points in mind while entering the instruction code:

1. The Opcode, Operand-1 and Operand-2 must be separated from each other using space or comma.
2. A line should contain only a single instruction.
3. Redundant spaces and commas in instructions are neglected.
4. Instructions are not case sensitive.
5. If not specified, the instructions are loaded at location 0000.
6. If not specified, execution starts from location 0000.
7. By default, Stack Pointer is FFFF.
8. Labels: Label names should end with a colon. There should NOT be a space between label name and colon.

Eg:

LABEL1: MOV A,B

9. Comments: Comments should be preceded by a semicolon. Everything that comes after a semicolon till the end of the line is regarded as a comment.

In cases where an instruction has two operands, everything that comes after the second operand is regarded as a comment. But still it is a good practice to use semicolon to start a comment.

Eg:

ADD C ;This is a comment

10. Some Assembler Directives can be used in this software. These directives MUST begin with a '@' symbol. The Assembler Directives supported are - @NEXT, @BEGIN and @EQU. These are described below.

ASSEMBLER DIRECTIVES

Assembler Directives supported are - @NEXT, @BEGIN and @EQU. These assembler directives are not compiled to generate hexcode.

@NEXT

Syntax: @NEXT <Memory_Address>

This assembler directive specifies where the next instruction is to be assembled in memory. The instructions after this directive are assembled at the memory location starting from the specified location.

This directive is used when one block of instructions have to be loaded at one location and another block of instructions have to be loaded at another location. The user has to just add the @NEXT directive before these blocks of instructions.

When a number of @NEXT is used and there is no @BEGIN in the program, then the location specified in the last @NEXT statement is taken as the execution start address.

@BEGIN

Syntax: @BEGIN <Memory_Address>

This assembler directive is used to specify the memory location from where execution begins. This directive can be given anywhere in the program. If more than one @BEGIN is used, then only the last one is considered.

@EQU

Syntax: <Label>: @EQU <Value>

This assembler directive is used to declare constants. The symbol name is given as a label. The 'Value' can be either a 2-digit or a 4-digit hexadecimal number. The Symbol Name can now be used any where in the program except Assembler Directives.

Eg:

```
PORT1: @EQU 02
OUT PORT1
```

User's Guide

The user can enter the program in the mnemonic textbox. To save the program and to open a file, the File Menu provides the Save and Open commands for the respective actions. The files, by default, end with '.sim' extension. But any text file is supported.

When the user types the program, it gets compiled automatically. User can also manually compile the program. The program is also compiled before execution. If there is any error in instructions then the errors along with their line number will be specified in the Messages window. If there are no errors, then the hexcodes are generated and loaded into memory. The user can execute the program as a whole or can perform Single Line Execution. The Stop Execution button can be used to stop the execution of the program midway.

While execution, Interrupts can be provided manually using the buttons in the 'Interrupts' area, which is found in the right hand side of the window.

To change the contents of Registers and Memory, user can use the corresponding actions found in 'Manage' Menu. This menu also provides facility to change Clock Frequency of the simulator.

To manage the contents of port use the 'Ports' tab in the right hand side of the window. There are also tabs which show the contents of Stack and Symbol Table. These two can be handy to the user while writing, executing and debugging the program.