

XASM-85 8080/8085 MS-DOS CROSS-ASSEMBLER

XASM85.COM is an 8080/8085 cross-assembler that runs under MS-DOS. It will allow you to assemble source code statements prepared with an ASCII text editor into a HEX object file that can be downloaded into a Model 100 or OLIVETTI M-10 and then converted back to binary for execution in the target machine.

The object file format is also readable by DD85, an MS-DOS based 8080/8085 object file debugger. This documentation does not attempt to teach you how to program in assembly language. It assumes that you are familiar with assemblers in general, and the 8080/8085 assembly language in particular.

The execution of the program can be via one of two methods. Method 1 is the interactive mode, where XASM85 will prompt you. To use this mode simply enter the command:

```
XASM85
```

XASM85 will respond with the prompt:

```
Name of source file?
Name of listing file <NUL> for none?
Name of object file?
```

Enter the full drive, path, name AND extension of the various files. When this has been completed, XASM85 will display:

File type	Name
Source	C:\TEST.ASM
List	NUL
Object	C:\TEST.HEX

for example if the various files were as specified. Then the program will display a message indicating that it is on Pass 0. During this phase the source file is examined, and the various labels, and symbolic addresses are evaluated. If no errors are found, it will commence Pass 1, where the object file will be generated. If any errors are discovered, a message will be displayed. Errors are explicitly identified in the listing file, which should be examined with an editor to see the type and nature of the error.

To execute the program automatically, enter the command:

```
XASM85 source-filespec [list-filespec] [object-filespec]
```

If the latter two filespecs are missing, then the source file name will be used for the listing file with the extension .LST, and the source file name will be used for the object file with the extension .HEX.

Source file statements take the form of a line of text divided into 4 fields. The first field, which must start in the first column of the line, is the LABEL field, and can contain a label of up to 16 alpha-numeric characters, the first character of which MUST be alphabetic.

If the label field is not used, then spaces or tabs must be used to skip to the second field, the OP-CODE field. This field must contain legal mnemonic op-codes or pseudo-ops.

The third field is separated from the second by spaces or tabs, and is the OPERAND field, and should contain valid expressions for the operands or pseudo-ops used in field 2.

The fourth field is the comment field, and if present, must be separated from the third field by spaces or tabs, and must start with the semi-colon symbol. Any characters may be present in the comment field, as it is ignored by the assembler.

The following pseudo-ops are recognised:

DB Define byte
DW Define word
EQU Define equates
ORG Define program origin address
END Declare last statement of program

An example of a valid source file is:

```
;Test program to demonstrate the assembler.
;Standard Equates

space      equ    32
cr         equ    13
lf         equ    10

          org     0c000h      ;Program start

begin      jmp     start

message    db      cr,lf,'This is a test message.',cr,lf,0

start      lxi     h,message  ;Point HL at test message,
mov        a,m              ;and get the next character.
cpi        0                ;Have we reached the NUL byte?
rz         ;Yes, so quit.
rst        4                ;No, so display the character,
inx        h                ;increment the pointer,
jmp        start            ;and repeat.
end        begin
```

Numeric quantities may be specified either in decimal, (default), hex, by ensuring that the first symbol is in the range 0 – 9, and postfixing the quantity with the h symbol, or binary, by postfixing the quantity with the b symbol.

For a more complete picture, examine the .ASM files that come with this package.