Command: **dis88 – disassembler [IBM]**
Syntax: **dis88** [**–o**] *infile* [*outfile*]
Flags: **–o** List the object code along with the assembly code
Examples: dis88 a.out >listing # Disassemble *a.out*
dis88 –o a.out listing # Ditto, but with object code

Dis88 disassembles 8088 object code to the assembly language format used by MINIX. It makes full use of symbol table information, supports separate instruction and data space, and generates synthetic labels when needed. It does not support 8087 mnemonics, symbolic data segment references, or the ESC mnemonic.

The program is invoked by:

```
dis88 [–o] infile [outfile]
```

The –o flag causes object code to be listed. If no outfile is given, *stdout* is used.

The text segment of an object file is always padded to an even address. In addition, if the file has split I/D space, the text segment will be padded to a paragraph boundary (i.e., an address divisible by 16). Due to padding, the disassembler may produce a few spurious, but harmless, instructions at the end of the text segment.

Because the information to which initialized data refers cannot generally be inferred from context, the data segment is treated literally. Byte values (in hexadecimal) are output, and long stretches of null data are represented by appropriate *.zerow* pseudo-ops. Disassembly of the bss segment, on the other hand, is quite straightforward, because uninitialized data is all zero by definition. No data is output in the bss segment, but symbolic labels are output as appropriate.

The output of operands in symbolic form is complicated somewhat by the existence of assembler symbolic constants and segment override opcodes. Thus, the program's symbol lookup routine attempts to apply a certain amount of intelligence when it is asked to find a symbol. If it cannot match on a symbol of the preferred type, it may output a symbol of some other type, depending on preassigned (and somewhat arbitrary) rankings within each type. Finally, if all else fails, it will output a string containing the address sought as a hex constant. For user convenience, the targets of branches are also output, in comments, as hexadecimal constants.

Error Messages

Various error messages may be generated as a result of problems encountered during the disassembly. They are listed below

Cannot access input file	– Input file cannot be opened or read
Cannot open output file	– Output file cannot be created
Input file not in object format	– Bad magic number
Not an 8086/8088 object file	– CPU ID of the file header is incorrect
Reloc table overflow	– Relocation table exceeds 1500 entries
Symbol table overflow	– Symbol table exceeds 1500 entries

Lseek error	– Input file corrupted (should never happen)
Warning: no symbols	– Symbol table is missing (use ast)
Cannot reopen input file	– Input file was removed during execution

Author

Dis88 was written and copyrighted by G. M. Harding and is included here by permission. It may be freely redistributed provided that complete source code, with all copyright notices, accompanies any redistribution. This provision also applies to any modifications you may make. You are urged to comment such changes, giving, as a minimum, your name and complete address.