29 Nov 86

HELP FILE for DazzleStar Disassembler

Introduction

This document is the file DZENGHLP.OVR and serves as a brief introduction to DazzleStar. It's provided in case you don't yet have the full manual, and as a form of on-line help.

DazzleStar takes a CP/M-80 binary file (for example a .COM or .OVR file), and produces Zilog Z80 mnemonics, either as a listing on your printer or as a .MAC file for Microsoft M80 or other assemblers.

This file serves as an introduction, sufficient to get you started and do useful work. It can be browsed with ^J, or printed using WordStar. To obtain the full manual, refer to DZ--READ.ME.

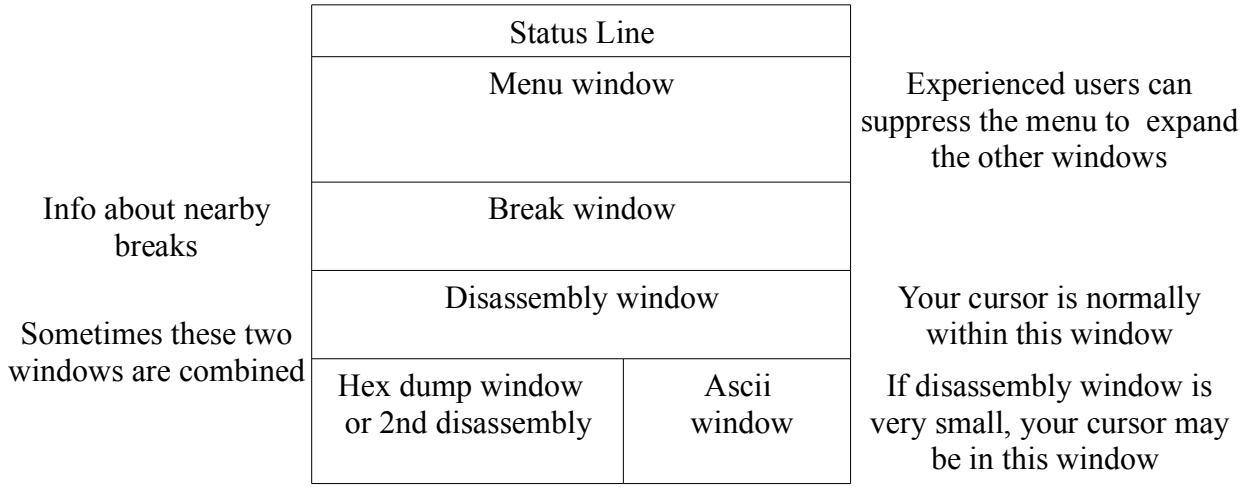
Few programs contain nothing but instructions, so DazzleStar keeps a break table, to specify how each part of your program is to be disassembled. There is also a symbol table, so that labels and values can be given whatever names (or phrases, even) that the user wishes.

Starting up (the command line)

The file to be disassembled must be specified in the command tail. Since the .COM extension is the most frequent, .COM is added if no extension is explicitly given. For example, to disassemble a file THATPROG.COM, type "DZ THATPROG" ("DZ THATPROG.COM" also works). Other files, for example THATPROG.OVR, can be specified by giving the name in full. A drive letter in front of the filename is accepted (e.g. "DZ B:THATPROG").

Windows on the screen

The screen is arranged with windows which are independently updated.



Moving thru the file

The command keys chosen are based on WordStar, so you should quickly find a lot of old friends on your keyboard.

^W	Scroll up 1 line
^E	Cursor up
^R	Scroll to previous page (or thereabouts)
^Z	Scroll down 1 line
^X	Cursor down
^C	Next page
^^	(up) also work, which allows use of
^_	(down) cursor arrows on some machines

Actually, just the letters alone also work OK, no need to use <CTRL>.

Moving further

To move a short distance through the file, use the scroll commands (especially ^C and ^R). Often you'll want to move further, so there are handy ways to do this:

Go to address	^A	hex address <return>
and display	^F	goes to the address in the 2nd and 3rd bytes of
screenful		the instruction specified by the cursor
		(to remember: A for Address, F for Follow)

Command groups

Some of the command codes are two characters long (similar to WordStar where analogies are possible). They are loosely grouped, according to the first character of the pair. The command groups are:

^K	-	various file commands
^O	-	Onscreen: how the screen is arranged
^D	-	Display other information
^Q	-	Quick: move quickly to another part of the file
^S	-	build Symbol table
^B	-	build Break table
^P	-	inline Parameter definition
^J	-	Help

"Quick" is not a very good mnemonic for the ^Q family, but many of the uses are similar to WordStar, for example ^QR, ^QC, ^QF, ^QA, ^QV are all inspired by WordStar.

HELP!

When the first character of a two-character command has been typed, if the second character is not typed within a reasonable time then the user is prompted by the display of an appropriate secondary menu.

^JH2	clears the top of the screen to make more room for displaying code, but turns off the menu.
^JH3	turns the menu back on again.
^J <space>	returns to the main menu.
^J	followed by anything else displays this file.

(Actually, JH2 and JH3 can be abbreviated to J2 and J3, not so analagous to WordStar, but quicker to type!).

Break table

DazzleStar builds a table of "breaks" (called the break table) which decides how each part of the input file is displayed. This terminology might confuse you if you're a DDT or ZSID user. Don't misunderstand this term -- it has NOTHING to do with things like DDT breakpoints.

The break table can be modified with ^Bx sequences, summarized in the ^B menu, and more fully described in the manual.

There are many types of break, giving you great flexibility. The ones to try first, satisfying the majority of needs, are:

^BB	Bytes (as in DB or DEFB)
^BI	Instructions (the kind of display you get on starting DZ)
^BW	Word values, 16-bit numbers (as in DW or DEFW)

Disk files etc

^KS	Write .DZ file (text file containing all symbols, comments, and breaks, so disassembly can continue another day)
^KR	Read .DZ file (produced by ^KS)
^KQ	return to CP/M -- nothing is saved
^KX	save break & symbol tables (if altered) and return to CP/M
^KP	print part or whole disassembly on LST: (or .PRN file)
^KW	write part or whole disassembly to disk (.MAC file)
^KG	Read "Ken Gielow" .BRK file, from Z80DIS21.

REMEMBER:	^KS does not take long. You can't turn back the clock and type ^KS after a power cut, or a head crash.
^KP, ^KW:	Just press <return> to obtain the most sensible defaults (the values displayed within square brackets).

Symbol table

The symbol table can be thought of as several independent symbol tables -- when an instruction is decoded DazzleStar decides what kind of symbol is involved (for example, a byte reference, a word reference, a label, a relative label, etc). When the instruction is displayed, your symbol name is only used if you asked for it to be used for this kind of symbol. So it is easy to tell DazzleStar to turn this:

```
LD HL,0100        (size of 256 byte block)
 ...
JP 0100           (restart the program)
```

into:

```
LD HL,blocksize
 ...
JP start
```

without any confusion between the two uses of 0100.

The commands are described more formally in the reference section of the manual, and much fuller examples are given in the tutorial section. For now let's just go through that example in a little more detail:

```
LD HL,0100
 ...
JP 0100
```

Move the cursor to the LD HL line. Type ^SIblocksize<return>. Move the cursor to the JP line. Type ^SIstart<return>. Now, when you display somewhere else in the file with a reference to 0100 it will show 'start' or 'blocksize' as appropriate.

^SI stands for "symbol indirect". Less frequently you will want to define a symbol where you are now, rather than the place an instruction references. There are lots of ^S commands to do this.

Auto-build breaks and symbol names

^SA auto-build symbol table (names derived from hex addresses)

^SA is the fastest way to build a good disassembly. Position the cursor to the start of the program (^QR) and use ^SA. DazzleStar starts disassembling, and adds names to the symbol table as it goes. If an area follows an unconditional transfer, it is provisionally marked as unreachable and not disassembled (to avoid entering false labels in the symbol table). Each time a call to a new routine is found, DazzleStar gives you back full control. This enables advanced users to examine the routine, and perhaps use the ^P family. Each time a reference to an undefined T-symbol (within the program area) is found you also gain control, again giving advanced users fine control. To continue auto-build from where it was halted, press "." (as in "to be continued...").

The name given by ^SA uses the letter appropriate for the use of the symbol and the four hex digits of its value. For example, decoding C3 34 12 in an area starting with an "I" break (Instructions) would generate a symbol L1234.

When no more passes are required the cursor will stay at the end of the program when '.' is pressed. Clean up the break table with ^BY. For a large disassembly this will take some time.

Fine tuning the display

^OF, ^OS, and ^OU give you finer control over the way in which the display is presented. This is discussed further in the manual. Moreover, DZINSTAL allows you to specify the initial settings of these and many other things to your liking.

If you have the manual, and you still can't get the display, printouts, and files exactly the way you like them (minus pickles, extra ketchup, and 734 sesame seeds on each bun) then you're hard to please! Mail me a letter and I'll be happy to see if I can satisfy your gourmet demands next time around. Very few people take up this offer -- so don't just say to yourself "someone is bound to have pointed THAT out already".

Comments

There are two sorts of comment, those between lines and those alongside the code. I like to call them major comments and minor comments.

Inserting a major comment:

> Move the cursor to the line following where you want the comment.
> Press <semicolon>. (The screen will be redisplayed, with a gap for your new comment).
> Type comment, terminating with <return>

Backspace is implemented, and can be used before <return> has been typed, but otherwise the comment cannot be edited. However, you can later retype it completely, and delete the old one. There's no limit to the number of lines of comment at any point. Comments can be inserted in any order.

Inserting a minor comment: much the same, but start by pressing <slash>.

Moving forward or back by just a few bytes

A digit (DON'T press the <CTRL> key at the same time, some computers get upset if you do) causes redisplay from the address of the current line plus the value of the digit you typed.

For example, move the cursor to a line (perhaps the first line of a routine) and type '0'. The address of this line (plus 0 = nothing) is used as the first line of the screen and the whole screen is redisplayed. So '0' slides your window down the code a bit (depending where your cursor is) but not as much as ^C would (unless your cursor was on the bottom line, in which case the result would be identical).

> '1' is useful if you think you're looking at instructions, but the decoding is "out of sync".
>
> '-' followed by a digit causes... (you guessed?)

These digit commands are particularly useful when you use ^QA.

WARNING

I would be very surprised if DazzleStar overwrote your system tracks, or arbitrary parts of the disk (and would VERY much want to hear about it if it happened). The most likely cause of accidents is through the lack of any warning before overwriting an existing file, for example, using ^KW when a .MAC file of that name already exists.

No precautions against computer or disk error are included. In particular, DazzleStar will fail if the disk it is writing to becomes full. As with any other software, make **BACK-UP COPIES OF ANY FILES YOU'D REGRET LOSING**. Do it **REGULARLY**. Nothing can eliminate head crashes.