230-923

XDASM Cross-Disassembler

XDASM is a powerful, MS-DOS based Cross-Disassembler which is used to reconstruct or debug source level code for various processor types. Its unique table-driven structure and output format adaptibility, makes XDASM the most universal program disassembler available.

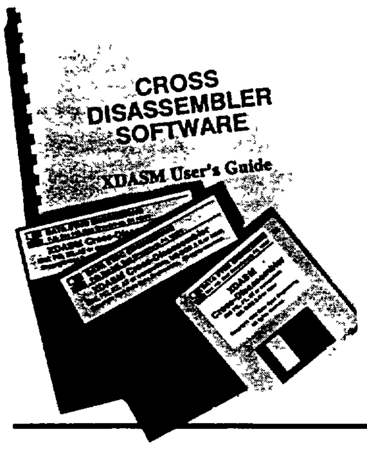
XDASM's disassembly process can be directly controlled by a user generated TAG file. The output source file is created from an Intel/Motorola Hex or Binary coded input file of up to 65K bytes.

XDASM is very easy to use and produces an "Assembler-ready" source file that can be immediately re-assembled with little or no editing.

Optional comment fields are attached to each disassembled line which show the current Program Counter, Instruction HEX bytes and ASCII character translations.

XDASM automatically inserts Origin, Define Byte and Equate directives when required. Labels are assigned to every instruction address that is referenced. De-blocking is used in the output listing to allow easier program interpretation.

Cross-Reference lists can be optionally appended to the output file to provide further program detail. The sorted reference lists are maintained as comment fields which are ignored by the assembler. Based on the processor type, up to four cross-reference lists may be generated. Label name address references are always provided. Other reference lists may include, Immediate Values, I/O Addresses and Special Address Registers. XDASM is capable of producing up to 130K of sorted references.



*** PC/MS-DOS based**
*** Table-driven disassembly**
*** Hex or Binary input files**
*** Creates assembly source**
*** Direct disassembly control**
*** Manufacturers Mnemonics**
*** Assigns Label names**
*** Inserts assembler directives**
*** Deblocks output listing**
*** Generates Cross-References**

Command Line

XDASM filename,type /options

filename = Hex/Binary input file
type = Processor name

Options:

B	Binary file input
C	Include line comments
L	Lower-case output
M	Mask 7-bit ASCII
R	Append cross-references
T	Use TAG file control
X	Cross-references only

Tables Included

* 1802 1805 1806
* 64180 Z180
* 6502 65C02
* 6800 6802 6808
* 6801 6803
* 6301 6303
* 6805
* 6809
* 68HC11
* 8048
* 8051
* 8085 8080
* 8096
* COP400
* COP800
* Z8
* Z80
* . . . call for others

Output Format Control

Format file = type.FMT

(Current default setting)

;	Insert directive, line 1
;	Insert directive, line 2
DFB	Define Hex byte directive
DFB	Define Text directive
DWH	Define Word, MSB first
DWL	Define Word, LSB first
ORG	Origin directive
EQU	Equate directive
END	End directive
SRC	Output file extension
L	Start of label character
:	End-of-label character
:	End-of-equate character
;	Comment field delimiter
"	Text string delimiters
0	Hex notation format

Tag File Control

Tag file = filename.TAG

Disassembly commands:

aaaa=B	Define Byte
aaaa=H	Hex load addr offset
aaaa=I	Instruction
aaaa=S	Skip byte
aaaa=T	Text byte
aaaa=>	Define Word, LSB
aaaa=<	Define Word, MSB

(aaaa = starting address)

Condensed Sample Listing

```
;============================================================;
; Disassembled Using XDASM   --   (C)1990 Data Sync Engineering ;
;============================================================;
;
;       Unresolved Address Reference list
;
L00CD:  EQU   000CDH
L00EF:  EQU   000EFH
L00F5:  EQU   000F5H
;
        ORG   00000H
;
L0000:  LD    HL,08000H          ;0000 21 00 80       !..
;
L0003:  DEC   HL                 ;0003 2B             +
        LD    A,H                ;0004 7C             |
        OR    L                  ;0005 B5             .
        JP    NZ,L0003           ;0006 C2 03 00       ...
;
;       Cross-references to LABEL Addresses
;
; L0000= 1D18
; L0003= 0006
; L0009= 01E1  01FC  02A3  02BE  0506  0524  06E4  06FF  085B  0AA3
;        0B0A  0B51  0CC5  0DC1  1172
```

Automatic Directives

ORG - Origin directives are used to specify the memory location of where the program resides. XDASM inserts origin directives in accordance with the input files specifications. Any time the Hex load address changes, XDASM will insert a new origin statement. An address offset can be added by using the "H" command in the TAG file.

DFB - Define Byte directives are inserted whenever an Unassigned Opcode or an Incomplete Instruction is encountered from the input file. The "B" command in the TAG file will also cause Define Byte statements.

EQU - If an address is referenced and is not found within the program, XDASM will equate a label to that address and will show it in the Unresolved Address Reference list. The key to a successful disassembly is to eliminate all unresolved references. If the address is valid, it can be manually equated by using the "G" command in the TAG file. This removes that address from the unresolved list.

Line Comments

Line comments are enabled by using the "C" option in the command line. Each disassembled line will contain a comment field showing the Instruction Address, the HEX bytes that made up the instruction and the ASCII character equivalent of the Hex bytes. This is very useful for distinguishing between code and text. In some cases the high bit may be set as a flag or to disguise text. The "M" option will mask the high bit for ASCII character display.

The TAG File

The TAG file is a separate ASCII character file that may be generated using a standard text editor. It is used to control the disassembly process by switching to various modes at specified addresses. Byte, Word, Text, Skip and Instruction modes are easily selected by entering the start address followed by a command character.

System Requirements

MS-DOS version 2.0 or later.
512 Kilobytes RAM.
5.25" / 3.50" floppy drive.

Your local sales representative is:

Available From:
MicroProcessor Engineering Ltd
133 Hill Lane
Southampton
SO1 5AF

Tel: 0703 631441
Fax: 0703 339691

*** Ask about the companion Cross-Assembler.**

DATA SYNC ENGINEERING
P.O. Box 146, East Stroudsburg, PA 18301 **(717) 421-1977**