ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Find the *Intel iSBC CSM/001* at our website: ***Click HERE***

ORDERING INFORMATION

Order Code	Shipping Media	Product Description
SD109CPM80A	A—Single-density 8" diskette	Multiplan spreadsheet program for use under CP/M* on 8080/8085 based small computers.
SD109CPM80B	B—Double-density 8" diskette	
SD109CPM80F	F—iPDS Format 5¼" diskette	

SUPPORT

Intel offers several levels of support for this product, depending on the system configuration in which it is used. Please consult the price list for a detailed description of the support options available.

AFN-00649B

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PASCAL 80
SOFTWARE PACKAGE

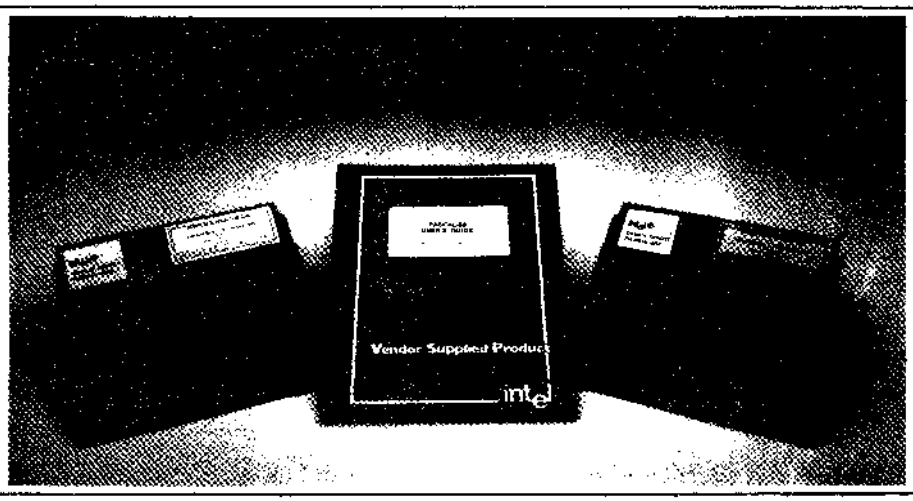
- **Offers a Superset of Standard Pascal**
- **Provides Highly Structured Language with Powerful Data Type Definitions to Suit Applications**
- **Compiles Pascal Source Code into Intermediate Code to Optimize Execution Speed and Storage**
- **Executes Compiler and Interprets the Intermediate Code on Intellec® Microcomputer Development Systems**
- **Provides a Utility to Produce Relocatable Object Modules Compatible with Other Intel® Languages**
- **Can Call Routines Written in PL/M 80, FORTRAN 80, or 8080/8085 Macro Assembler**
- **Allows Modular Breakdown of Large Programs and Separate Compilation of Individual Modules**
- **Gives Application Control Over Run-Time Errors by Providing User-Declared Error Procedures**

PASCAL 80 Software Package consists of a compiler and an interactive Run-Time System designed to provide the Pascal programming language as a software development tool for Intellec Development System Users.

Pascal is a highly-structured, block-oriented programming language that is now gaining wide acceptance as a powerful software development tool. Its rigid structure encourages and enforces good programming techniques, which, combined with a high level of readability, helps produce more reliable software.

Standard Intel development tools, such as CREDIT editor can be used to create and modify Pascal source programs. The compiler compiles this source and creates a P-Code file. The Run-Time System executes this P-Code in an interpretive manner under ISIS-II.

*Pascal language as defined in *PASCAL User Manual and Report*, Second Edition, Kathleen Jenson and Niklaus Wirth.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

LANGUAGE FEATURES

Data Structures

Pascal allows the user to define labels, constants, data types, variables, procedures, and functions.

Variable Types

Variables can be defined according to the following system-defined data types: boolean, integer, real, character, array, record, string, set, file, and pointer.

User-Defined Types

New types can be defined by the user for added flexibility.

File Handling Procedures

Pascal provides procedures to allow a user's program to interface with the ISIS-II file manager. Routines provided are: RESET, REWRITE, CLOSE, PUT, GET, SEEK, and PAGE.

Input/Output Procedures

Routines are provided to interact with the console or an ISIS file. These procedures are: READ, WRITE, READLN, WRITELN, plus BUFFER and BLOCK Read and Write.

Dynamic Memory Allocation

The procedures NEW, MARK, and RELEASE allow the user to obtain and release memory space at run-time for dynamically allocating variable storage.

String Handling

Pascal provides powerful tools for defining and manipulating strings and character arrays. These facilities enable concatenation of strings, character and pattern scans, insertion, deletion, and pointer manipulation.

Recursion

Pascal allows a PROCEDURE definition to include a call to itself, a powerful construct in many mathematical algorithms.

PROGRAM TRACING FACILITY

The PASCAL 80 System incorporates a program tracing facility which allows for selectively monitoring the execution of a Pascal program. When the TRACE flag is set, the line number of each program statement being executed is output to the console.

The TRACE flag may be manipulated in two ways:

—The TRACEON command (of the Run-Time System) will set the flag, and the TRACEOFF command will reset the flag.

—Pressing the Interrupt 4 switch on the Intellec System front panel will toggle the TRACE flag; i.e., the flag will be set if it was reset, and vice-versa.

COMPILER DIRECTIVES (PARTIAL LIST)

Compiler Command Line Directives

NOLIST
No list file is produced; used for fast compilation of "clean" programs.

NOCODE
No code file is produced; used for syntax error checking.

ERRLIST
List file is limited to only those Pascal lines that contain errors, along with the error messages produced.

LIST (file-name)
Specifies the name of the list file.

CODE (file-name)
Specifies the name of the code file.

NOECHO
Error lines are echoed on the console unless this directive is specified.

Embedded Compiler Directives

$C text
Causes text to appear in code file (allows for comments, copyrights, etc.).

$I+
Causes checking for I/O completion after each I/O transfer. Failure results in a run-time error. ($I− causes no checking, and no errors on I/O failure.)

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

$R+
Causes Range Checking to occur, so that an out-of-range value causes a Run-Time error. ($R− suppresses generation of code for Range Checking.)

$O+
Causes the compiler to operate in overlay mode. Overlays allow less source code to reside in memory. ($O− causes no overlays, which decreases compile time, since there are fewer disk accesses.)

$T+
Causes the compiler to generate tracing instructions to be used by the TRACE facility. ($T− suppresses tracing instructions.)

BENEFITS

Brings Pascal to Intellec Microcomputer Development Systems:

—Pascal is a block-structured, highly-readable programming language, suitable for a wide-range of applications.

—Pascal is being acclaimed as the programming language of the future; it is being taught in many colleges and universities around the country.

—PASCAL 80 Run-Time System provides great ease in programming formatted I/O operations.

PASCAL 80 provides a portable language for application programs running under ISIS-II.

PASCAL 80 can be used to evaluate complicated algorithms using a natural language.

PASCAL 80 compiler generates intermediate Pseudo-code.

—P-code is optimized for speed and storage space.

—P-code is approximately 50% to 70% smaller than corresponding machine code.

—P-code is machine independent, providing code portability to any CPU.

Makes the Intellec Development System a more valuable tool. Extension of software support to include Pascal makes software development and resource management more flexible.

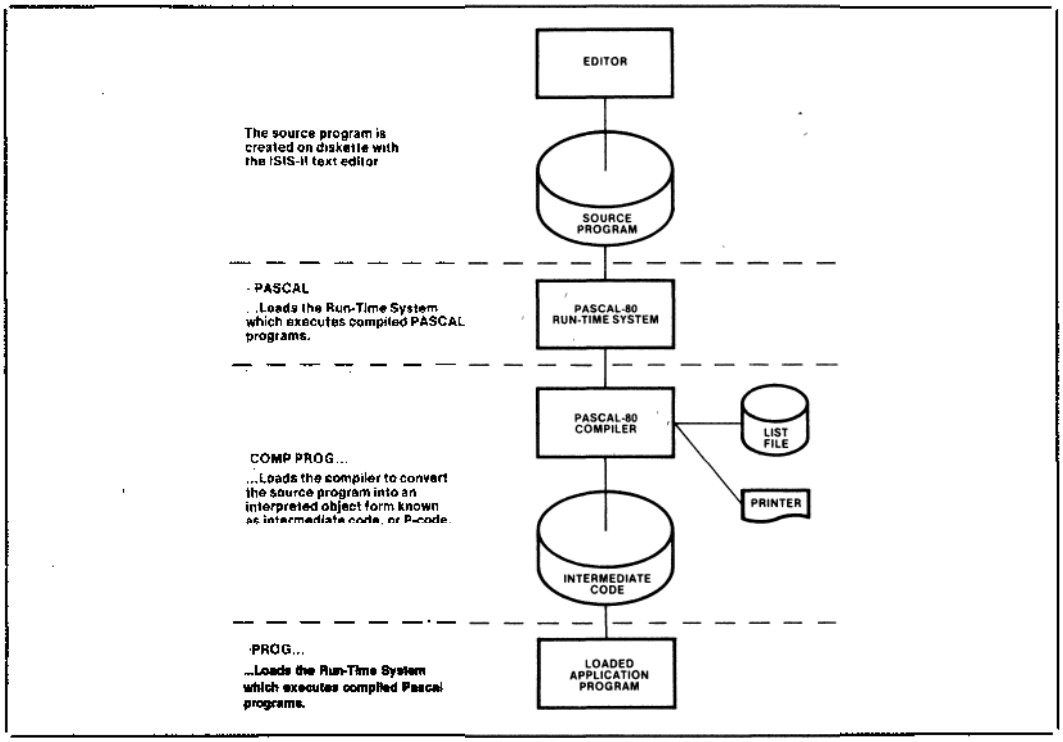


Figure 1. Program Development Cycle

AFN-012339

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 1. Sample Program Listing Showing Nesting Levels

BUFFER.PAS Program Listing

```
Line  Seg  Proc  Lev  Disp
   1    1     1          1   program example;
   2    1     1          3
   3    1     1          3   { Example using bufferread and bufferwrite with break characters }
   4    1     1          3
   5    1     1          3   var buffer: string,
   6    1     1         44       disk   storage: file;
   7    1     1         64       break  char;
   8    1     1         65       new   len, len: integer,
   9    1     1         67       buff   array: packed array[0..80] of char;
  10    1     1        108
  11    1     1    0    0   begin
  12    1     1    1    0     rewrite (disk   storage, 'data');
  13    1     1    1   27     writeln('Input a line of text: ');
  14    1     1    1   68     readln (buffer),
  15    1     1    1   87     len := bufferwrite(disk   storage, buffer[1], length(buffer)),
  16    1     1    1  109     repeat
  17    1     1    2  109       reset(disk   storage);
  18    1     1    2  116       writeln, writeln;
  19    1     1    2  132       write('Input break char [cntrl Z to stop]: '),
  20    1     1    2  179       readln(break);
  21    1     1    2  197       if not eof(input) then
  22    1     1    3  208         begin
  23    1     1    4  208           new   len  = bufferread(disk   storage, buff   array, len, ord(break)),
  24    1     1    4  226           writeln('The buffer read. '),
  25    1     1    4  262           writeln(copy(buffer, 1, abs(new   len))),
  26    1     1    4  292           writeln('Length. ', abs(new   len):0);
  27    1     1    4  331           if new   len < 0 then writeln('(Break char not found)');
  28    1     1    3  378         end,
  29    1     1    1  378     until eof(input),
  30    1     1    0  388   end
```

SPECIFICATIONS

Operating Environment

REQUIRED HARDWARE
Intellec® Microcomputer Development System
—Model 800
—Series II Model 220, Model 230, Model 240
64KB of Memory
Dual-Diskette Drives
—Single- or Double-Density*
System Console
—Intel® CRT or non-Intel® CRT

*Recommended.

REQUIRED SOFTWARE
ISIS-II Diskette Operating System
—Single- or Double-Density

OPTIONAL SOFTWARE
ISIS-II CREDIT™ (CRT-Based Text Editor)

Documentation Package

PASCAL 80 User's Guide (9801015-01)

PASCAL User Manual and Report, Second Edition, Kathleen Jensen and Niklaus Wirth

Shipping Media

Flexible Diskettes
—Single- and Double-Density

AFN-01233B

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

ORDERING INFORMATION

Part Number	Description
MDS-381*	PASCAL 80 Software Package

Requires Software License

*MDS is an ordering code only and is not used as a product name or trademark. MDS® is a registered trademark of Mohawk Data Sciences Corporation.

SUPPORT CATEGORY: Level D

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

WordStar* WORD PROCESSING SOFTWARE

- **Powerful, reliable, and user-friendly word processing software package**
- **Six on-screen menus and ten Help menus provide quick command reference**
- **Printout enhancements provide numerous combined print functions**
- **Simple formatting commands including Hyphen-Help**
- **Streamlines text entry**
- **Horizontal scrolling for wide pages**
- **Wordwrap removes need to worry about right margin**
- **On-screen formatting displays text exactly as it will be printed**
- **All functions easily controlled despite differences in printers and consoles**

WordStar, a popular word processing program written for use under the CP/M† operating system, gives screen editing capabilities in an easy to learn and use format. The program is in use by programmers, and engineers for documentation and program entry, as well as managers and secretaries.

With WordStar, the user can easily make insertions and deletions, move or copy blocks of text, and search for and replace a string of text. WordStar will automatically reformat text upon command as these editing functions are performed.

Documents produced by WordStar can include any combination desired of pagination (page numbers), right and left justification, subscripts, superscripts, underlining, boldface type, overstrikes, crossouts, and even accents for use in foreign languages. Commands for all of these are entered with simple control-character keystrokes which are well documented in the program's six help menus.

All WordStar commands are easily executed using the CTRL key and the standard typewriter keys. Using the CTRL key, the function of standard keys can be changed to perform useful editing commands. The cursor-movement diamond (a group of standard keys on the keyboard) allows fast access to any area of text.

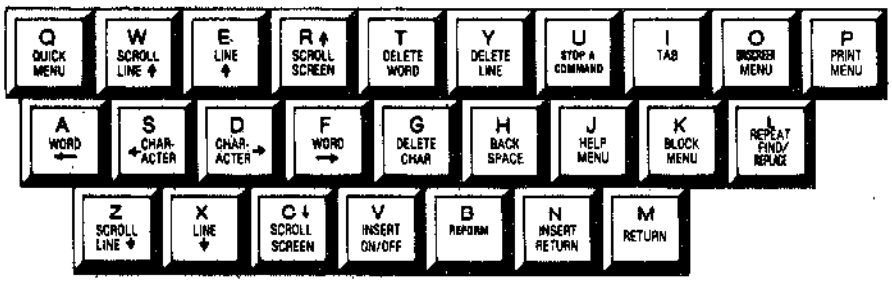


Figure 1. WordStar Keyboard Functions

The following are trademarks of Intel Corporation and its affiliates and may be used only to identify Intel products: BXP, CREDIT, I, ICE, iCS, im, Insite, Int$_e$l, INTEL, Intelevision, int$_e$ligent Identifier™, int$_e$ligent Programming™, intellink, Intellec, iMMX, iOSP, iPDS, iRMX, iSBC, iSBX, Library Manager, MCS, MULTIMODULE, Megachassis, Micromainframe, MULTIBUS, Multichannel, Plug-A-Bubble, PROMPT, Promware, RUPI, RMX/80, System 2000, UPI, and the combination of iCS, iRMX, iSBC, iSBX, ICE, iP ICE, MCS, or UPI and a numerical suffix. Intel Corporation Assumes No Responsibility for the use of Any Circuitry Other Than Circuitry Embodied in an Intel Product. No Other Patent Licenses are Implied. ©INTEL CORPORATION, 1983

MAY 1983
ORDER NUMBER:210762-002

*WordStar MailMerge and SpellStar are trademarks of MicroPro International.

†CP/M is a registered trademark of Digital Research Inc.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FEATURES

WordStar is designed to be simple for the novice to use, while remaining sophisticated enough to be appealing to even the most advanced user.

Standard typewriter keys are combined with the "Control" key to provide a wide variety of editing functions (Fig. 1). All cursor control is localized to the ten keys in the "Cursor-Movement Diamond" (Fig. 2), and the on-screen menu details the functions of the other keys, so the user can quickly find functions without memorizing them.

Wordwrap is a feature of WordStar that allows the typist to entirely disregard margins. When typed characters go beyond the right margin, WordStar brings the last full word down to the next line automatically. The only time the Return key needs to be used is between paragraphs. Margins can be automatically right and left justified both during and after entry.

Horizontal Scrolling give the flexibility in creating documents too wide to fit on the video screen. When Wordwrap is disabled and a line is being typed beyond normal screen width, the displayed lines are automatically scrolled offscreen to the left. A single keystroke can be used to move the lines back to their normal position. Editing functions can also use Horizontal Scrolling to examine and modify any part of a wide document.

The On-Screen Formatting feature displays the text on the screen as it will appear when it is printed. This allows the changing of margins, spacing, and other format variables without requiring the use of a number of intermediate printouts.

Hyphen-Help aids in reformatting by positioning the cursor over a word requiring hyphenation at the end of a line, and allowing the user to select a hyphenation point or decide not to hyphenate. Hyphens entered this way are "soft", and will not be printed if the document is reformatted and the hyphen is no longer required. Permanent or "hard" hyphens are inserted while typing and will always be printed.

WordStar's Find and Replace command allows the text to be scanned for a specified character string. Once the string is found it will be replaced quickly with the updated information. Options with this command allow the user to perform functions like finding the "n^{th}" occurrence, performing the operation "n" times, replacing the string without verification by the user each time, searching backward, or to compensate for differences in upper and lower case letters (i.e., at the beginning of a sentence).

Entire blocks of text can be marked at their beginning and ending, then moved to a new area as easily as moving the cursor. Different block control commands allow the duplication and deletion of blocks as well.

Column Move assists in the creation and editing of tables of data. With Column Move, a column can be taken from one table and moved to another table or to another place in the same table. Columns can also be easily duplicated or deleted.

Over 20 Page Formatting commands enable a range of functions from producing automatic page headings to overriding built-in parameters for line height and character width. Margins can be set and number of lines typed per page can be dictated via these very simple commands. These page commands are especially useful in long documents.

Decimal Tab is a feature that assists in aligning figures into columns. When a number is entered into a decimal tab position, it will be automatically aligned so that its decimal point is directly below the decimal point of the number on the line above.

Files can be combined with each other to form derivative documents. One file can be inserted at any point of another, beginning middle or end, with equal simplicity.

Print controls, single letters entered while editing to enhance the printout, permit the user of underscore, boldface, underlining, double-strike, superscript, subscript, overprint, and nearly any combination of the above. This facilitates the generation of mathematical formulas with subscripts and superscripts, and allows the text to include foreign words and phrases with accents above and below certain letters. Alternate character pitch, for italics, and even ribbon color selection can be controlled by WordStar if these options are available on the printer in use.

Can be used with MailMerge* to generate chained printing combining form letters with mailing lists. MailMerge allows names to be drawn from the address and inserted into the text of the form letter.

SpellStar* may also be used with WordStar to check the spelling in a document against both a 20,000-word standard dictionary and a user-generated supplemental dictionary which can be used to store names, buzz words, or abbreviations.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

WordStar is easily adapted to nearly any video terminal and document printer, despite the wide variation of options and communication standards used by these devices. At installation time the user is prompted through a series of questions which configure that WordStar installation to fit the hardware at hand.

When an existing file is edited and saved, any previous version of the file is saved under the original filename as a .BAK extension (i.e., after updating a file entitled "LETTER" there would be two files, LETTER and LETTER.BAK on the diskette). When a document is to be updated, its latest extension is automatically used as input. Whenever a new .BAK extension is created, the older .BAK version is destroyed.

WordStar allows documents of almost any size to be entered and edited. A Memory Management feature automatically transfers text to and from mass storage if the document is too large to be held in main memory at one time.

There are six Main Menus (Fig. 3A–3F) and ten Help Menus to guide even the most inexperienced user through a WordStar editing session. The Main or On-Screen Menus are displayed at the top of the screen along with the text being edited. Should the user desire to fill the entire screen with text, the menu displays can be turned on and off as desired.

The ten Help Menus guide the user through the use of all editing functions from Moving Text to Paragraph Reform.

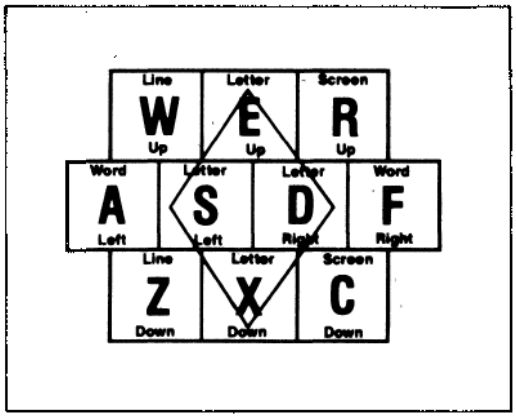


Figure 2. The Cursor Movement Diamond

```
        A:TEST.DOC  PAGE 1 LINE 1  COL 1              INSERT ON
                    < < <      M A I N   M E N U      > > >
*  *  Cursor  Movement  *  *|* Delete *|  * Miscellaneous * | * Other  Menus *
^S char left  ^D char right |^G  char   |^I Tab   ^B Reform  | (from Main only)
^A word left  ^F word right |DEL chr lf|^V Insert On or Off |^J Help  ^K Block
^E line  up   ^X line down  |^T word rt|^L Find/Replce again|^Q Quick ^P Print
*   *   Scrolling    *    *|^Y  line   |RETURN End paragraph|^O Onscreen
^Z line up    ^W line down  |          |^N  Insert a RETURN |
^C screen up  ^R screen down|          |^U  Stop a command  |
L-----!-----!-----!-----!-----!-----!-----!-----!-----!-----!--------R
```

Figure 3A. Main Menu: guide to the most frequently used commands. This menu —and all other menus—can be called up at any time, or dropped to allow full-screen viewing of the text.

```
^J      A:TEST.DOC  PAGE 1 LINE 1 COL 1              INSERT ON
                    < < <      H E L P   M E N U     > > >
                                |                     | * Other  Menus *
 H Display and set the help level | S Status line     | (from Main only)
 B Paragraph reform (CTRL B command) | R Ruler line    |^J Help  ^K Block
 F Flags in rightmost column of screen | M Margins and tabs |^Q Quick ^P Print
 D Dot commands, print ctrl(P command) | P Place markers  |^O Onscreen
                                | V Moving text       |Space bar returns
                                |                     |you to Main Menu.
L-----!-----!-----!-----!-----!-----!-----!-----!-----!-----!--------R
```

Figure 3B. Help Menu: a directory of commands that control help levels and show reference information.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
^Q      A:TEST.DOC  PAGE 1 LINE 1  COL 1              INSERT ON
                  < < <     Q U I C K   M E N U     > > >
*  *   Cursor Movement  *  *|* Delete *|  * Miscellaneous * | * Other  Menus *
S left side    D right side |Y line  rt|F Find text in file | (from Main only)
E top of scrn  X bottom scrn|DEL lin lf|A Find and  Replace |^J Help  ^K Block
R top of file  C end of file|*  *  *  *|L Find  misspelling |^Q Quick ^P Print
B top of block K end of block          |Q Repeat command or |   ^O Onscreen
0-9  marker    Z up    W down          |  key  until  space |Space bar returns
V last Find or block                   |  bar  or other key |you to Main Menu.
L----!----!----!----!----!----!----!----!----!----!----!--------R
```

Figure 3C. Quick Menu: expanded cursor movement, deletion, find/replace commands, and place marker commands.

```
^K      A:TEST.DOC  PAGE 1 LINE 1  COL 1              INSERT ON
                  < < <     B L O C K   M E N U     > > >
 * Saving Files * |* Block Operations *| * File Operations *| * Other  Menus *
 S Save and resume|B  Begin   K  End    |R  Read   P  Print  | (from Main only)
 D Save—done      |H  Hide / Display    |O  Copy   E  Rename |^J Help  ^K Block
 X Save and exit  |C  Copy    Y  Delete |J  Delete           |^Q Quick ^P Print
 Q Abandon file   |V  Move    W  Write  |* Disk  Operations *|^O Onscreen
* Place  Markers *|N  Column    off (ON)|L Change logged disk|Space bar returns
0-9 Set/hide # 0-9|                     |F Directory on (OFF)|you to Main Menu.
L----!----!----!----!----!----!----!----!----!----!----!--------R
```

Figure 3D: Block Menu: instructions for using block and place markers, saving and printing a file, and inserting other files.

```
^O      A:TEST.DOC  PAGE 1 LINE 1  COL 1              INSERT ON
                  < < <  O N S C R E E N   M E N U  > > >
* Margins & Tabs * |* Line  Functions *| *  More Toggles  * | * Other  Menus *
L Set left  margin |C Center text       |J Justify   off (ON)| (from Main only)
R Set right margin |S Set line spacing  |V Vari-tabs off (ON)|^J Help  ^K Block
X Release  margins |                    |H Hyph-help off (ON)|^Q Quick ^P Print
I Set   N Clear tab|   *  Toggles  *    |E Soft hyph on (OFF)|^O Onscreen
G Set paragraph tab|W Wrd wrap off (ON)|D Prnt disp off (ON)|Space bar returns
F Ruler from line  |T Rlr line off (ON)|P Pge break off (ON)|you to Main Menu.
L----!----!----!----!----!----!----!----!----!----!----!--------R
```

Figure 3E. Onscreen Menu: functions that perform onscreen document formatting (such as line spacing, tabs, margins, justification, and wordwrap).

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

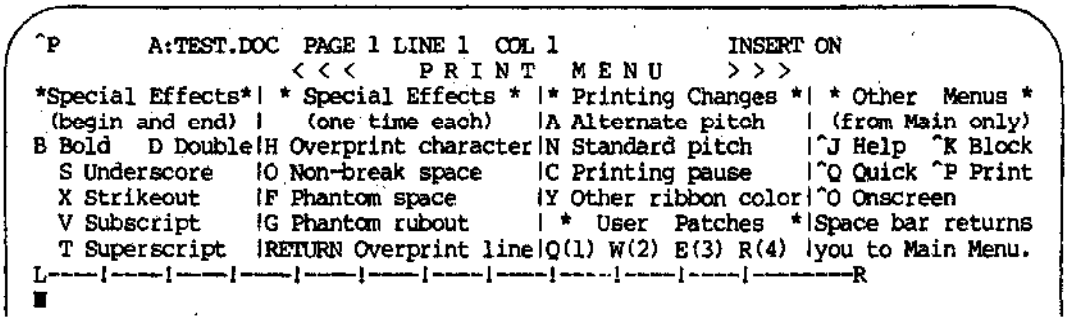


Figure 3F. Print Menu: special print control characters including subscripts, superscripts, boldface, double strike, and strikeout.

BENEFITS

WordStar is an advanced word-processing program that can turn any CP/M based personal computer into a sophisticated yet easy to learn and use text processor. It takes very little time for even the least-trained user to learn to productively generate documentation with WordStar.

The simplifying features of WordStar do not detract from its acceptance by advanced users. Menus and other features are designed to be unobtrusive when they are not needed. WordStar's sophistication means that it will not run out of horsepower as the user progresses, but will always be an appealing and highly productive tool.

With WordStar there is no question about the appearance of the printed output, since the text can be displayed on the screen exactly as it is to be printed.

Time savings when using WordStar will be considerable. Generation of new text is easier than by handwritten/typed means. When WordStar is used for program editing it supplies powerful features unavailable in other editors. With WordStar, both code and documentation can be generated at the same time within the same environment.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 1.

EDITING COMMAND INDEX

^	hold CTRL key, type letter
^A	cursor left word
^B	reform paragraph
^C	scroll up screenful
^D	cursor right character
^E	cursor up line
^F	cursor right word
^G	delete character right
^H	cursor left character
^I	tab
^J	help PREFIX
^K	editing PREFIX
^L	find/replace again
^M	(Same as RETURN)
^N	insert hard carriage return
^O	formatting PREFIX
^P	print control PREFIX
^Q	editing PREFIX
^R	scroll down screenful
^S	cursor left character
^T	delete word right
^U	interrupt
^V	insert on/off
^W	scroll down line
^X	cursor down line
^Y	delete line
^Z	scroll up line
^JB	explain reform
^JD	summarize print directives
^JF	explain Flags
^JH	set Help level
^JI	command index
^JM	explain tabs and Margins
^JP	explain Place markers
^JR	explain Ruler line
^JS	explain Status line
^JV	explain moVing text
^K0–^K9	set/hide marker 0–9
^KB	mark/hide Block beginning
^KC	Copy block
^KD	Done edit (save)
^KE	rEname file
^KF	File directory on/off
^KH	Hide/display marked block
^KJ	delete additional file
^KK	mark blocK end
^KL	change Logged disk
^KN	column mode on/off
^KO	cOpy file
^KP	Print
^KQ	abandon edit
^KR	Read additional file
^KS	Save and reedit
^KV	moVe block
^KW	Write block to additional file
^KX	save and eXit
^KY	delete block
^OC	Center cursor line
^OD	print control display on/off
^OE	soft hyphen Entry on/off
^OF	margins & tabs from line
^OG	paraGraph tab
^OH	Hyphen-Help on/off
^OI	set tab stop
^OJ	Justification on/off
^OL	set left margin
^ON	clear tab stop
^OP	Page break display on/off
^OR	set Right margin
^OS	set line Spacing
^OT	ruler display on/off
^OW	Wordwrap
^PA–^PZ	enter ^A–^Z
^PM	make next line overprint
^PO	enter non-break space
^Q0–^Q9	cursor to marker 0–9
^QA	find and replace
^QB	cursor Block beginning
^QC	cursor end file
^QD	cursor right end line
^QE	cursor top screen
^QF	Find
^QK	cursor blocK end
^QL	find misspelling
^QP	cursor Previous position
^QQ	repeat next command
^QR	cursor beginning of file
^QS	cursor left Side screen
^QV	cursor source
^QW	continuous downward scroll
^QX	cursor bottom of screen
^QY	delete to end line
^QZ	continuous upward scroll
^Qdel	delete to beginning line
^DEL	delete character left
^ESCAPE	error release
^LINE FEED	(same as ^J)
^RETURN	hard carriage return
^TAB	tab

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 1. (Continued)

NO-FILE COMMANDS			
D	open a Document file	**O**	cOpy file
E	rEname file	**P**	Print
F	File directory on/off	**R**	Run program
H	set Help level	**S**	run SpellStar (optional)
L	change Logged disk	**X**	eXit to operating system
M	run MailMerge (optional)	**Y**	delete file
N	open a Non-document file		

SPECIFICATIONS

OPERATING ENVIRONMENT

Hardware Required

8080 or 8085 CPU
5¼" or 8" Diskette drive
Printer
64K Bytes of memory
Console with absolute cursor addressing
Note Intellec Series II and III require iMDX-511

Optional hardware

Additional mass storage

Software Required

CP/M 2.2 operating system

DOCUMENTATION PACKAGE

Wordstar Training Guide
Wordstar Operator's Guide

Wordstar General Information Manual
Wordstar Reference Manual
Wordstar Installation Manual

SUPPORT

Intel offers several levels of support for this product, depending on the system configuration in which it is used. Please consult the price list for a detailed description of the support options available.

Intel software license is required.

ORDERING INFORMATION

Description

WordStar word processing software package for use under the CP/M operating system

Order Code	Shipping Media
SD111CPM80ASU	A—Single-density 8" diskette
SD111CPM80BSU	B—Double-density 8" diskette
SD111CPM80FSU	F—iPDS Format 5¼" diskette

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

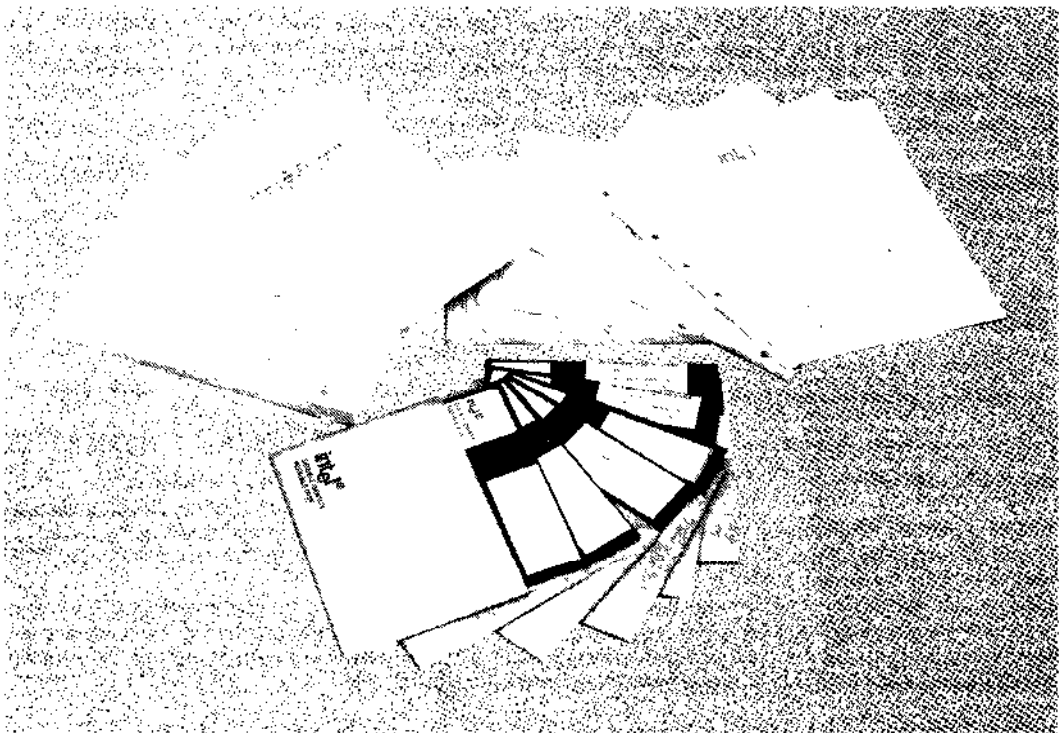
iAPX 86,88
SOFTWARE DEVELOPMENT PACKAGES FOR SERIES II/PDS

- **PL/M 86/88 High Level Programming Language**
- **ASM 86/88 Macro Assembler for iAPX 86,88 Assembly Language Programming**
- **LINK 86/88 and LOC 86/88 Linkage and Relocation Utilities**
- **CONV 86/88 Converter for Conversion of 8080/8085 Assembly Language Source Code to iAPX 86, 88 Assembly Language Source Code**
- **OH 86/88 Object-to-Hexadecimal Converter**
- **LIB 86/88 Library Manager**

The iAPX 86,88 Software Development Packages for Series II provide a set of software development tools for the iAPX 86/88 CPUs and the iSBC 86/12A single board computer. The packages operate under the ISIS-II operating system on Intel Microcomputer Development Systems—Model 800, Series II or the Personal Development System (PDS)—thus minimizing requirements for additional hardware or training for Intel Microcomputer Development System users.

These packages permit 8080/8085 users to efficiently upgrade existing programs into iAPX 86/88 code from either 8080/8085 assembly language source code or PL/M 80 source code.

For the new Intel Microcomputer Development System user, the packages operating on a PDS or an Intellec Series II, such as a Model 235, provide total iAPX 86,88 software development capability.



© INTEL CORPORATION 1983

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PL/M 86/88 COMPILER FOR SERIES II/PDS

- **Language is Upward Compatible from PL/M 80, Assuring MCS-80/85™ Design Portability**
- **Supports 16-bit Signed Integer and 32-bit Floating Point Arithmetic in Accordance with IEEE Proposed Standard**
- **Easy-to-Learn, Block-Structured Language Encourages Program Modularity**
- **Produces Relocatable Object Code Which is Linkable to All Other 8086 Object Modules**
- **Supports Full Extended Addressing Features of the iAPX 86/10 and 88/10 Microprocessors (Up to 1 Mbyte)**
- **Code Optimization Assures Efficient Code Generation and Minimum Application Memory Utilization**

Like its counterpart for MCS-80/85 program development, PL/M 86/88 is an advanced, structured high-level programming language. The PL/M 86/88 compiler was created specifically for performing software development for the Intel iAPX 86,88 Microprocessors.

PL/M 86/88 has significant new capabilities over PL/M 80 that take advantage of the new facilities provided by the iAPX 86,88 microsystem, yet the PL/M 86/88 language remains compatible with PL/M 80.

With the exception of hardware-dependent modules, such as interrupt handlers, PL/M 80 applications may be recompiled with PL/M 86/88 with little need for modification. PL/M 86/88, like PL/M 80, is easy to learn, facilitates rapid program development, and reduces program maintenance costs.

PL/M is a powerful, structured, high-level system implementation language in which program statements can naturally express the program algorithm. This frees the programmer to concentrate on the logic of the program without concern for burdensome details of machine or assembly language programming (such as register allocation, meanings of assembler mnemonics, etc.).

The PL/M 86/88 compiler efficiently converts free-form PL/M language statements into equivalent 86/88 machine instructions. Substantially fewer PL/M statements are necessary for a given application than if it were programmed at the assembly language or machine code level.

The use of PL/M high-level language for system programming, instead of assembly language, results in a high degree of engineering productivity during project development. This translates into significant reductions in initial software development and follow-on maintenance costs for the user.

FEATURES

Major features of the Intel PL/M 86/88 compiler and programming language include:

Block Structure

PL/M source code is developed in a series of modules, procedures, and blocks. Encouraging program modularity in this manner makes programs more readable, and easier to maintain and debug. The language becomes more flexible by clearly defining the scope of user variables (local to a private procedur global to a public module, for example).

The use of procedures to break down a large problem is paramount to productive software development. The PL/M 86/88 implementation of a block structure allows the use of REENTRANT which is especially useful in system design.

Language Compatibility

PL/M 86/88 object modules are compatible with object modules generated by all other 86/88 translators. This means that PL/M programs may be linked to programs written in any other 86/88 language.

Object modules are compatible with ICE-88 and ICE-86 units; DEBUG compiler control provides the In-Circuit Emulators with symbolic debugging capabilities.

PL/M 86/88 Language is upward-compatible with PL/M 80, so that application programs may be easily ported to run on the iAPX 86 or 88.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Supports Five Data Types

PL/M makes use of five data types for various applications. These data types range from one to four bytes, and facilitate various arithmetic, logic, and addressing functions:

—Byte: 8-bit unsigned number
—Word: 16-bit unsigned number
—Integer: 16-bit signed number
—Real: 32-bit floating point number
—Pointer: 16-bit or 32-bit memory address indicator

Another powerful facility allows the use of BASED variables that map more than one variable to the same memory location. This is especially useful for passing parameters, relative and absolute addressing, and memory allocation

Two Data Structuring Facilities

In addition to the five data types and based variables, PL/M supports two data structuring facilities. These add flexibility to the referencing of data stored in large groups.

—Array: Indexed list of same type data elements
—Structure: Named collection of same or different type data elements
—Combinations of Each: Arrays of structures or structures of arrays

8087 Numerics Support

PL/M programs that use 32-bit REAL data may be executed using the Numeric Data Processor for improved performance. All floating-point operations supported by PL/M may be executed on the 8087 NDP, or the 8087 Emulator (a software module) provided with the package. Determination of use of the chip or emulator takes place at link-time, allowing compilations to be run-time independent.

Built-In String Handling Facilities

The PL/M 86/88 language contains built-in functions for string manipulaiton. These byte and word functions perform the following operations on character strings: MOVE, COMPARE, TRANSLATE, SEARCH, SKIP, and SET.

Interrupt Handling

PL/M has the facility for generating interrupts to the iAPX 86 or 88 via software. A procedure may be defined with the INTERRUPT attribute, and the compiler will automatically initialize an interrupt vector at the appropriate memory location. The compiler will also generate code to same and restore the processor status, for execution of the user-defined interrupt handler routine. The procedure SET$INTERRUPT, the function retuning an INTERRUPT$PTR, and the PL/M statement CAUSE$INTERRUPT all add flexibility to user programs involving interrupt handling.

Segmentation Control

The PL/M 86/88 compiler takes full advantage of program addressing with the SMALL, COMPACT, MEDIUM, and LARGE segmentation controls. Programs with less than 64KB total code space can exploit the most efficient memory addressing schemes, which lowers total memory requirements. Larger programs can exploit the flexibility of extended one-megabyte addressing.

Code Optimization

The PL/M 86/88 compiler offers four levels of optimization for significantly reducing overall program size.

—Combination or "folding" of constant expressions; and short-circuit evaluation of Boolean expressions.
—"Strength reductions" (such as a shift left rather than multiply by 2); and elimination of common sub-expressions within the same block.
—Machine code optimizations; elimination of superfluous branches; re-use of duplicate code; removal of unreadable code.
—Byte comparisons (rather than 20-bit address calculations) for pointer variables; optimization of based-variable operations.

Compiler Controls

The PL/M 86/88 compiler offers more than 25 controls that facilitate such features as:

—Conditional compilation
—Intra- and Inter-module cross reference
—Corresponding assembly language code in the listing file
—Setting overflow conditions for run-time handling

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

BENEFITS

PL/M 86/88 is designed to be an efficient, cost-effective solution to the special requirements of iAPX 86 or 88 Microsystem Software Development, as illustrated by the following benefits of PL/M use:

Low Learning Effort

PL/M 86/88 is easy to learn and to use, even for the novice programmer.

Earlier Project Completion

Critical projects are completed much earlier than otherwise possible because PL/M 86/88, a structured high-level language, increases programmer productivity.

Lower Development Cost

Increases in programmer productivity translate immediately into lower software development costs because less programming resources are required for a given programmed function.

Increased Reliability

PL/M 86/88 is designed to aid in the development of reliable software (PL/M 86/88 programs are simple statements of the program algorithm). This substantially reduces the risk of costly correction of errors in systems that have already reached full production status, as the more simply stated the program is, the more likely it is to perform its intended function.

Easier Enhancements and Maintenance

Programs written in PL/M tend to be self-documenting, thus easier to read and understand. This means it is easier to enhance and maintain PL/M programs as the system capabilities expand and future products are developed.

iAPX 86,88 MACRO ASSEMBLER FOR SERIES II/PDS

- **Powerful and Flexible Text Macro Facility with Three Macro Listing Options to Aid Debugging**
- **Highly Mnemonic and Compact Language, Most Mnemonics Represent Several Distinct Machine Instructions**
- **"Strongly Typed" Assembler Helps Detect Errors at Assembly Time**
- **High-Level Data Structuring Facilities Such as "STRUCTUREs" and "RECORDs"**
- **Over 120 Detailed and Fully Documented Error Messages**
- **Produces Relocatable and Linkable Object Code**

ASM 86/88 is the "high-level" macro assembler for the iAPX 86,88 assembly language. ASM 86/88 translates symbolic 86/10, 88/10 assembly language mnemonics into 86/10, 88/10 relocatable object code.

ASM 86/88 should be used where maximum code efficiency and hardware control is needed. The iAPX 86,88 assembly language includes approximately 100 instruction mnemonics. From these few mnemonics the assembler can generate over 3,800 distinct machine instructions. Therefore, the software development task is simplified, as the programmer need know only 100 mnemonics to generate all possible 86/10, 88/10 machine instructions. ASM 86/88 will generate the shortest machine instruction possible given no forward referencing or given explicit information as to the characteristics of forward referenced symbols.

ASM 86/88 offers many features normally found only in high-level languages. The iAPX 86,88 assembly language is strongly typed. The assembler performs extensive checks on the usage of variables and labels. The assembler uses the attributes which are derived explicitly when a variable or label is first defined, then makes sure that each use of the symbol in later instructions conforms to the usage defined for that symbol. This means that many programming errors will be detected when the program is assembled, long before it is being debugged on hardware.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FEATURES

Major features of the Intel iAPX 86,88 assembler and assembly language include:

Powerful and Flexible Text Macro Facility

— Macro calls may appear anywhere
— Allows user to define the syntax of each macro
— Built-in functions
conditional assembly (IF-THEN-ELSE, WHILE)
repetition (REPEAT)
string processing functions (MATCH)
support of assembly time I/O to console (IN, OUT)
— Three Macro Listing Options include a GEN mode which provides a complete trace of all macro calls and expansions

High-Level Data Structuring Capability

— STRUCTURES: Defined to be a template and then used to allocate storage. The familiar dot notation may be used to form instruction addresses with structure fields.
— ARRAYS: Indexed list of same type data elements.
— RECORDS: Allows bit-templates to be defined and used as instruction operands and/or to allocate storage.

Fully Supports iAPX 86,88 Addressing Modes

— Provides for complex address expressions involving base and indexing registers and (structure) field offsets.
— Powerful EQU facility allows complicated expressions to be named and the name can be used as a synonym for the expression throughout the module.

Powerful STRING MANIPULATION INSTRUCTIONS

— Permit direct transfers to or from memory or the accumulator.
— Can be prefixed with a repeat operator for repetitive execution with a count-down and a condition test.

Over 120 Detailed Error Messages

— Appear both in regular list file and error print file.
— User documentation fully explains the occurrence of each error and suggests a method to correct it.

Support for ICE-86™ Emulation and Symbolic Debugging

— Debug options for inclusion of symbol table in object modules for In-Circuit Emulation with symbolic debugging.

Generates Relocatable and Linkable Object Code—Fully Compatible with LINK 86/88, LOC 86/88 and LIB 86/88

— Permits ASM 86/88 programs to be developed and debugged in small modules. These modules can be easily linked with other ASM 86/88 or PL/M 86/88 object modules and/or library routines to form a complete application system.

BENEFITS

The iAPX 86,88 macro assembler allows the extensive capabilities of the 86/88 CPU's to be fully exploited. In any application, time and space critical routines can be effectively written in ASM 86/88. The 86,88 assembler outputs relocatable and linkable object modules. These object modules may be easily combined with object modules written in PL/M 86/88—Intel's structured, high-level programming language. ASM 86/88 compliments PL/M 86/88 as the programmer may choose to write each module in the language most appropriate to the task and then combine the modules into the complete applications program using the iAPX 86,88 relocation and linkage utilities.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

CONV 86/88
MCS®-80/85 to iAPX 86,88 ASSEMBLY LANGUAGE CONVERTER UTILITY PROGRAM

- **Translates 8080/8085 Assembly Language Source Code to iAPX 86,88 Assembly Language Source Code**
- **Provides a Fast and Accurate Means to Convert 8080/8085 Programs to the iAPX 86/88 Facilitating Program Portability**
- **Automatically Generates Proper ASM 86/88 Directives to Set Up a "Virtual 8080" Environment that is Compatible with PL/M 86/88**

In support of Intel's commitment to software portability, CONV 86/88 is offered as a tool to move 8080/8085 programs to the iAPX 86/88. A comprehensive manual, "MCS-86 Assembly Language Converter Operating Instructions for ISIS-II Users," covers the entire conversion process. Detailed methodology of the conversion process is fully described therein.

— CONV 86/88 will accept as input an error-free 8080/8085 assembly-language source file and optional controls, and produce as output, optional PRINT and OUTPUT files.

— The PRINT file is a formatted copy of the 8080/8085 source and the 86/88 source file with embedded caution messages.

— The OUTPUT file is an 86/88 source file.

— CONV 86/88 issues a caution message when it detects a potential problem in the converted 86/88 code.

— A transliteration of the 8080/8085 programs occurs, with each 8080/8085 construct mapped to its exact 86/88 counterpart:

Registers
Condition flags
Instruction
Operands
Assembler directives
Assembler control lines
Macros

Because CONV 86/88 is a transliteration process, there is the possibility of as much as a 15%–20% code expansion over the 8080/8085 code. For compactness and efficiency it is recommended that critical portions of programs be re-coded in iAPX 86,88 assembly language.

Also, as a consequence of the transliteration, some manual editing may be required for converting instruction sequences dependent on:

—instruction length, timing, or encoding
—interrupt processing*
—PL/M parameter passing conventions*

*Mechanical editing procedures for these are suggested in the converter manual.

The accompanying figure illustrates the flow of the conversion process. Initially, the abstract program may be represented in 8080/8085 or iAPX 86,88 assembly language to execute on that respective target machine. The conversion process is porting a source destined for the 8080/8085 to the 86/88 via CONV 86/88.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

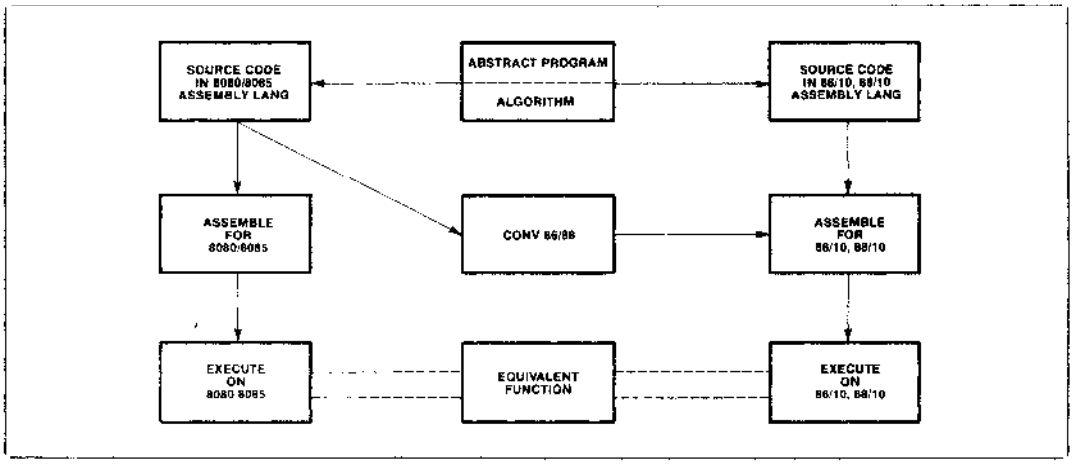


Figure 1. Porting 8080/8085 Source Code to the iAPX 86/10 and 88/10

LINK 86/88

- **Automatic Combination of Separately Compiled or Assembled iAPX 86, 88 Programs Into a Relocatable Module**
- **Automatic Selection of Required Modules from Specified Libraries to Satisfy Symbolic References**
- **Extensive Debug Symbol Manipulation, Allowing Line Numbers, Local Symbols, and Public Symbols to be Purged and Listed Selectively**
- **Automatic Generation of a Summary Map Giving Results of the LINK 86/88 Process**
- **Abbreviated Control Syntax**
- **Relocatable Modules may be Merged into a Single Module Suitable for Inclusion in a Library**
- **Supports "Incremental" Linking**
- **Supports Type Checking of Public and External Symbols**

LINK 86/88 combines object modules specified in the LINK 86/88 input list into a single output module. LINK 86/88 combines segments from the input modules according to the order in which the modules are listed.

LINK 86/88 will accept libraries and object modules built from PL/M 86/88, ASM 86/88, or any other translator generating Intel's iAPX 86/88 Relocatable Object Modules.

Support for incremental linking is provided since an output module produced by LINK 86/88 can be an input to another link. At each stage in the incremental linking process, unneeded public symbols may be purged.

LINK 86/88 supports type checking of PUBLIC and EXTERNAL symbols reporting an error if their types are not consistent.

LINK 86/88 will link any valid set of input modules without any controls. However, controls are available to control the output of diagnostic information in the LINK 86/88 process and to control the content of the output module.

LINK 86/88 allows the user to create a large program as the combination of several smaller, separately compiled modules. After development and debugging of these component modules the user can link them together, locate them using LOC 86/88 and enter final testing with much of the work accomplished.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

LIB 86/88

- **LIB 86/88 is a Library Manager Program which Allows You to:**
  - **Create Specially Formatted Files to Contain Libraries of Object Modules**
  - **Maintain These Libraries by Adding or Deleting Modules**
  - **Print a Listing of the Modules and Public Symbols in a Library File**
- **Libraries Can be Used as Input to LINK 86/88 Which Will Automatically Link Modules from the Library that Satisfy External References in the Modules Being Linked**
- **Abbreviated Control Syntax**

Libraries aid in the job of building programs. The library manager program LIB 86/88 creates and maintains files containing object modules. The operation of LIB 86/88 is controlled by commands to indicate which operation LIB 86/88 is to perform. The commands are:

CREATE: creates an empty library file
ADD: adds object modules to a library file
DELETE: deletes modules from a library file
LIST: lists the module directory of library files
EXIT. terminates the LIB 86 program and returns control to ISIS-II

When using object libraries, the linker will call only those object modules that are required to satisfy external references, thus saving memory space.

LOC 86/88

- **Automatic Generation of a Summary Map Giving Starting Address, Segment Addresses and Lengths, and Debug Symbols and their Addresses**
- **Extensive Capability to Manipulate the Order and Placement of Segments in iAPX 86/88 Memory**
- **Abbreviated Control Syntax**
- **Automatic and Independent Relocation of Segments. Segments May Be Relocated to Best Match Users Memory Configuration**
- **Extensive Debug Symbol Manipulation, Allowing Line Numbers, Local Symbols, and Public Symbols to be Purged and Listed Selectively**

Relocatability allows the programmer to code programs or sections of programs without having to know the final arrangement of the object code in memory.

LOC 86/88 converts relative addresses in an input module to absolute addresses. LOC 86/88 orders the segments in the input module and assigns absolute addresses to the segments. The sequence in which the segments in the input module are assigned absolute addresses is determined by their order in the input module and the controls supplied with the command.

LOC 86/88 will relocate any valid input module without any controls. However, controls are available to control the output of diagnostic information in the LOC 86/88 process, to control the content of the output module, or both.

The program you are developing will almost certainly use some mix of random access memory (RAM), read-only memory (ROM), and/or programmable read-only memory (PROM). Therefore, the location of your program affects both cost and performance in your application. The relocation feature allows you to develop your program on the Intellec development system and then simply relocate the object code to suit your application.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

OH 86/88

- **Converts an iAPX 86/88 Absolute Object Module to Symbolic Hexadecimal Format**
- **Facilitates Preparing a File for Later Loading by a Symbolic Hexadecimal Loader, such as the iSBC™ Monitor SDK-86 Loader, or Universal PROM Mapper**
- **Converts an Absolute Module to a More Readable Format that can be Displayed on a CRT or Printed for Debugging**

The OH 86/88 utility converts an 86/88 absolute object module to the hexadecimal format. This conversion may be necessary to format a module for later loading by a hexadecimal loader such as the iSBC 86/12 monitor or Universal PROM Mapper. The conversion may also be made to put the module in a more readable format than can be displayed or printed.

The module to be converted must be in absolute format; the output from LOC 86/88 is in absolute format.

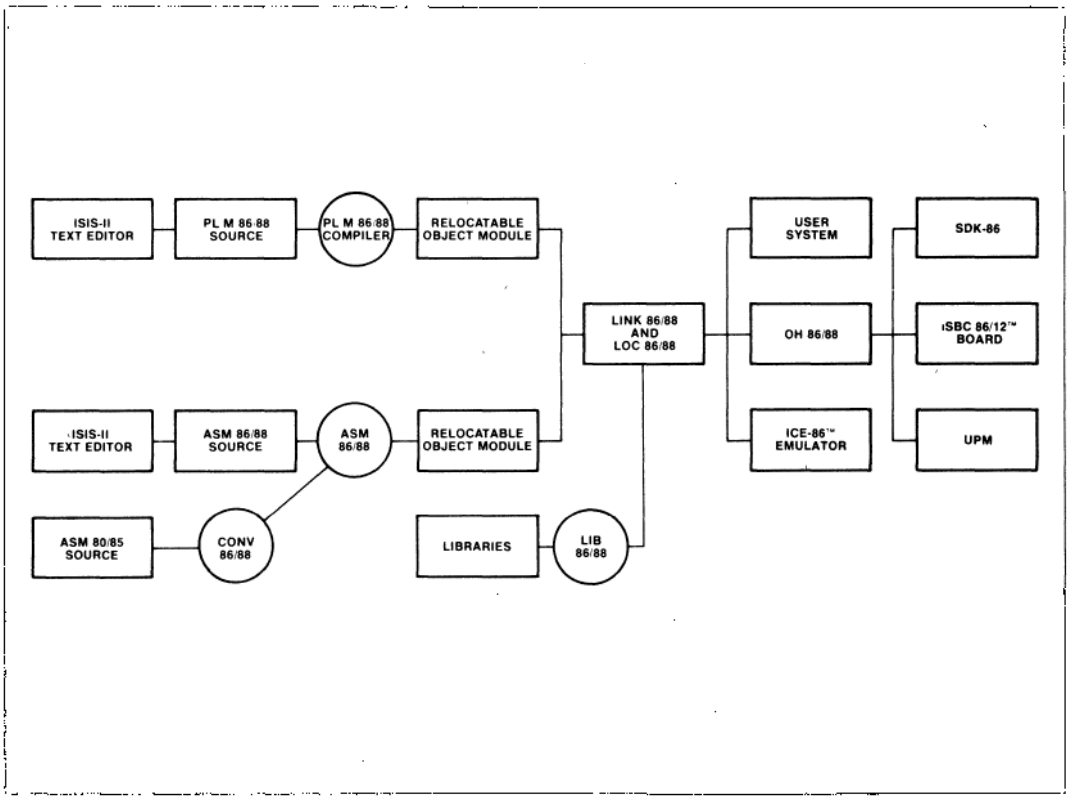


Figure 2. iAPX 86,88 Software Development Cycle

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SPECIFICATIONS

Operating Environment

Intel Microcomputer Development Systems
Intel Personal Development System

Documentation

PL/M-86 Programming Manual

ISIS-II PL/M-86 Compiler Operator's Manual

MCS-86 User's Manual

MCS-86 Software Development Utilities Operating Instructions for ISIS-II Users

MCS-86 Macro Assembly Language Reference Manual

MCS-86 Macro Assembler Operating Instructions for ISIS-II Users

MCS-86 Assembly Language Converter Operating Instructions for ISIS-II Users

Universal PROM Programmer User's Manual

ORDERING INFORMATION

iAPX 86,88 Software Development Packages for Series II:

Part No.	Description
MDS-308*	Assembler and Utilities Package
MDS-309*	PL/M compiler and Utilities Package
MDS-311*	PL/M compiler, Assembler, and Utilities Package

All Packages Require Software Licenses

SUPPORT:

Hotline Telephone Support, Software Performance Reports (SPR), Software Updates, Technical Reports, Monthly Newsletters are available

*MDS is an ordering code only and is not used as a product name or trademark. MDS® is a registered trademark of Mohawk Data Sciences Corporation.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

86/88/186/188 SOFTWARE PACKAGES

FORTRAN 86/88 Software Package

- Features High-Level Language Support for Floating-Point Calculation, Transcendentals, Interrupt Procedures, and run-time exception handling
- Meets ANS FORTRAN 77 Subset Language Specifications
- Supports Complex Data Types

PASCAL 86/88 Software Package

- Resident on iAPX 86 Based Intel Microcomputer Development Systems
- Object Compatible and Linkable with PL/M 86/88, ASM 86/88 and FORTRAN 86/88
- Supports Large Array Operation

PL/M 86/88/186/188 Software Package

- Advanced Structured System Implementation Language for Algorithm Development
- Supports 16-bit Signed Integer and 32-bit Floating Point Arithmetic in Accordance with IEEE Proposed Standard
- Easy-to-Learn Block-Structured Language Encourages Program Modularity

iC-86 C Compiler for the 8086

- Implements Full C Language
- Produces High Density Code Rivaling Assembler
- Supports Intel Object Module Format (OMF)

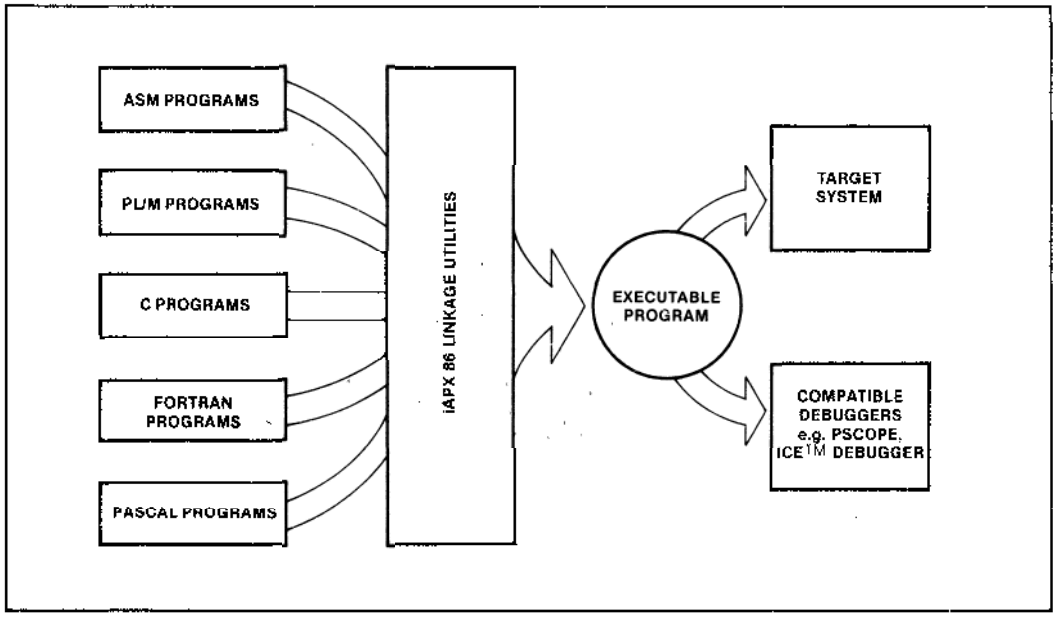


Figure 1. Program modules compiled with any of the iAPX 86 languages may be linked together. Each language is compatible with Intel's debug tools.

Intel Corporation Assumes No Responsibility for the Use of Any Circuitry Other Than Circuitry Embodied in an Intel Product. No Other Circuit Patent Licenses are Implied. Information Contained Herein Supercedes Previously Published Specifications On These Devices From Intel.
INTEL CORPORATION, 1983

SEPTEMBER 1984
ORDER NUMBER: 210689-003

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

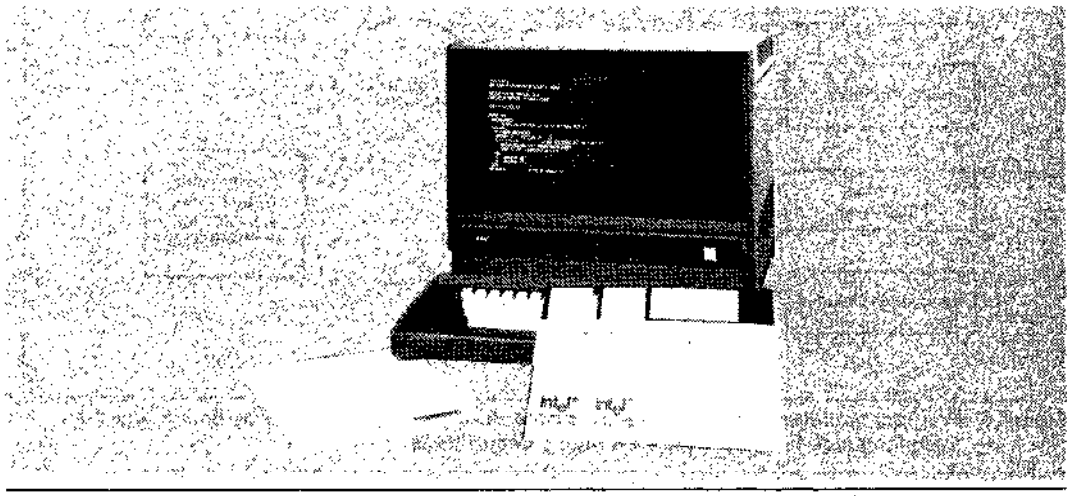
FORTRAN 86/88
SOFTWARE PACKAGE

- **Features high-level language support for floating-point calculations, transcendentals, interrupt procedures, and run-time exception handling**
- **Meets ANS FORTRAN 77 Subset Language Specifications**
- **Supports iAPX 86/20, 88/20 Numeric Data Processor for fast and efficient execution of numeric instructions**
- **Uses REALMATH Floating-Point Standard for consistent and reliable results**
- **Supports Arrays Larger Than 64K**
- **Unlimited User Program Symbols**
- **Offers powerful extensions tailored to microprocessor applications**
- **Offers upward compatibility with FORTRAN 80**
- **Provides FORTRAN run-time support for iAPX 86,88,186,188-based design**
- **Provides users ability to do formatted and unformatted I/O with sequential or direct access methods**
- **ICE™ Symbolic Debugging Fully Supported**
- **PSCOPE Source Level Debugging Fully Supported**
- **Supports complex data types**

FORTRAN 86/88 meets the ANS FORTRAN 77 Language Subset Specification and includes many features of the full standard. Therefore, the user is assured of portability of most existing ANS FORTRAN programs and of full portability from other computer systems with an ANS FORTRAN 77 Compiler.

FORTRAN 86/88 programs developed and debugged on the Intel Microcomputer Development Systems may be tested with the prototype using ICE symbolic debugging, and executed on an RMX-86 operating system, or on a user's iAPX 86,88,186,188-based operating system.

FORTRAN 86/88 is one of a complete family of compatible programming languages for iAPX 86,88,186,188 development: PL/M, Pascal, FORTRAN, and Assembler. Therefore, users may choose the language best suited for a specific problem solution.



©INTEL CORPORATION, 1983

MAY 1983

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FEATURES

Extensive High-Level Language Numeric Processing Support

Single (32-bit), double (64-bit), and double extended precision (80-bit) floating-point data types

REALMATH Proposed IEEE Floating-Point Standard) for consistent and reliable results

Full support for all other data types: integer, logical, character

Ability to use hardware (iAPX 86/20, 88/20 Numeric Data Processor) or software (simulator) floating-point support chosen at link time

ANS FORTRAN 77 Standard

Intel® Microprocessor Support

FORTRAN 86/88 language features support of iAPX 86/20, 88/20 Numeric Data Processor

Compiler generates in-line iAPX 86/20, 88/20 Numeric Data Processor object code for floating-point arithmetic (See Figure 1)

Intrinsics allow user to control iAPX 86/20, 88/20 Numeric Data Processor

iAPX 86,88,186,188 architectural advantages used for indexing and character-string handling

Symbolic debugging of application using ICE emulators

Source level debugging using PSCOPE.

FLOATING-POINT-STATMENT

```
 TEMPER = (PRESS - VOLUM / QUEK) - 3.45 / (PRESS - VOLUM / QUEK)
&    - (PRESS - VOLUM / QUEK) * (PRESS - VOLUM / QUEK)
```

OBJECT CODE GENERATED

```
Intel FORTRAN-86 Compiler

                                                    ; STATEMENT # 2
0013  9BD9060C00        FLD     VOLUM
0018  9B08360000        FDIV    QUEK
0010  9B082E0800        FSUBR   PRESS
0022  9BDDD1            FST     TOS+1H
0025  9B2ED83E0000      FDIVR   CS:@CONST
002B  9BD9C9            FXCHG   TOS+1H
002E  9BDDD2            FST     TOS+2H
0031  9BDEE9            FSUBRP
0034  9BD9C1            FLD     TOS+1H
0037  9B08C8            FMUL    TOS
003A  9BDDC2            FFREE   TOS+2H
003D  9BDEE1            FSUBP
0040  9BD91E0400        FSTP    TEMPER
0045  9B                WAIT
```

iAPX 86/20, 88/20 MACHINE CODE

ASSEMBLER MNEMONICS

Figure 2. Object Code Generated by FORTRAN 86/88 for a Floating-Point Calculation Using iAPX 86/20, 88/20 Numeric Processor.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Microprocessor Application Support

—Direct byte- or word-oriented port I/O

—Reentrant procedures

—Interrupt procedures

Flexible Run-Time Support

Application object code may be executed in iAPX 86,88,186,188-based environment of user's choice:

—a Series III or Series IV Intellec Development System

—an iAPX 86,88,186,188-based system with iRMX-86 Operating System

—an iAPX 86,88,186,188-based system with user-designed Operating System

Run-time exception handling for fixed-point numerics, floating-point numerics, and I/O errors

Relocatable object libraries for complete run-time support of I/O and arithmetic functions. In-line code execution is generated for iAPX 86/20, 88/20 Numeric Data Processor

BENEFITS

FORTRAN 86/88 provides a means of developing application software for the Intel iAPX 86,88,186,188 products lines in a familiar, widely accepted, and industry-standard programming language. FORTRAN 86,88 will greatly enhance the user's ability to provide cost-effective software development for Intel microprocessors as illustrated by the following:

Early Project Completion

FORTRAN is an industry-standard, high-level numerics processing language. FORTRAN programmers can use FORTRAN 86/88 on microprocessor projects with little retraining. Existing FORTRAN software can be compiled with FORTRAN 86/88 and programs developed in FORTRAN 86/88 can run on other computers with ANS FORTRAN 77 with little or no change. Libraries of mathematical programs using ANS 77 standards may be compiled with FORTRAN 86/88.

Application Object Code Portability for a Processor Family

FORTRAN 86/88 modules "talk" to the resident Intellec development operating system using Intel's standard interface for all development-system software. This allows an application developed under the ISIS-II operating system to execute on iRMX/86, or a user-supplied operating system by linking in the iRMX/86 or other appropriate interface library. A standard logical-record interface enables communication with non-standard I/O devices.

Comprehensive, Reliable and Efficient Numeric Processing

The unique combination of FORTRAN 86/88, iAPX 86/20, 88/20 Numeric Data Processor, and REALMATH (Proposed IEEE Floating-Point Standard) provide universal consistency in results of numeric computations and efficient object code generation.

SPECIFICATIONS

Operating Environment

Intel Microcomputer Development Systems (Series III/Series IV)

Documentation Package

FORTRAN 86/88 User's Guide

AFN-01653B

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

ORDERING INFORMATION

Part Number	Description
MDS*-315	FORTRAN 86/88 Software Package

Requires Software License

SUPPORT

Intel offers several levels of support for this product which are explained in detail in the price list. Please consult the price list for a description of the support options available.

*MDS is an ordering code only and is not used as a product name or trademark. MDS is a registered trademark of Mohawk Data Sciences Corporation.

AFN-01653B

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

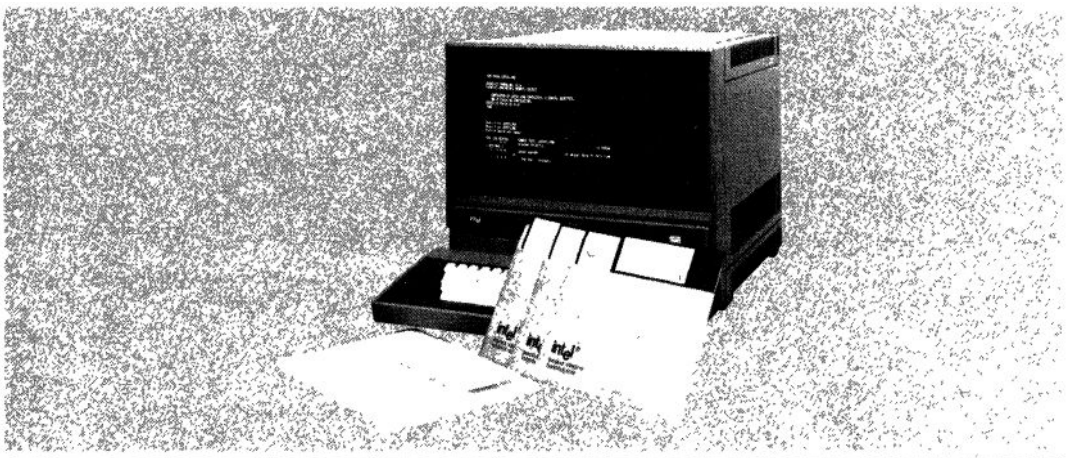
PASCAL 86/88
SOFTWARE PACKAGE

- **Resident on iAPX 86 Based Intel Microcomputer Development Systems**
- **Object Compatible and Linkable with PL/M 86/88, ASM 86/88 and FORTRAN 86/88**
- **ICE™ Symbolic Debugging Fully Supported**
- **PSCOPE Source Level Debugging Fully Supported**
- **Implements REALMATH for Consistent and Reliable Results**
- **Supports large array operation**
- **Unlimited User Program Symbols**
- **Supports iAPX86/20, 88/20 Numeric Data Processors**
- **Strict Implementation of ISO Standard Pascal**
- **Useful Extensions Essential for Microcomputer Applications**
- **Separate Compilation with Type-Checking Enforced Between Pascal Modules**
- **Compiler Option to Support Full Run-Time Range-Checking**

PASCAL 86/88 conforms to and implements the ISO Draft Proposed Pascal standard. The language is enhanced to support microcomputer applications with special features, such as separate compilation, interrupt handling and direct port I/O. To assist the development of portable software, the compiler can be directed to flag all non-standard features.

The PASCAL 86/88 compiler runs on Series III and Series IV Microcomputer Development Systems. A well-defined I/O interface is provided for run-time support. This allows a user-written operating system to support application programs as an alternate to the development system environment. Program modules compiled under PASCAL 86/88 are compatible and linkable with modules written in PL/M 86/88, ASM 86/88 or FORTRAN 86/88. With a complete family of compatible programming languages for the iAPX 86, 88, 186, 188 one can implement each module in the language most appropriate to the task at hand.

PASCAL 86/88 object modules contain symbol and type information for program debugging using ICE™ emulators and PSCOPE source language debugger. For final production version, the compiler can remove this extra information and code.



Intel Corporation Assumes No Responsibility for the Use of Any Circuitry Other Than Circuitry Embodied in an Intel Product. No Other Circuit Patent Licenses are Implied. Information Contained Herein Supercedes Previously Published Specifications of These Devices from Intel.
JUNE 1984
©INTEL CORPORATION, 1983

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FEATURES

Includes all the language features of Jensen & Wirth Pascal as defined in the ISO Draft Proposed Pascal Standard.

Supports required extensions for microcomputer applications.

—Interrupt handling
—Direct port I/O

Separate compilation extensions allow:

—Modular decomposition of large programs

—Linkage with other Pascal modules as well as PL/M 86/88/186/188, ASM 86/88/186/188 and FORTRAN 86/88.

—Enforcement of type-checking at LINK-time

Supports numerous compiler options to control the compilation process, to INCLUDE files, flag non-standard Pascal statements and others to control program listings and object modules.

Utilizes the IEEE standard for Floating-Point Arithmetic (the Intel REALMATH standard) for arithmetic operations.

Well-defined and documented run-time operating system interfaces allow the user to execute the applications under user-designed operating systems.

Predefined type extensions allow:

—Create precision in read, integer, and unsigned calculations.

—Means to check 8087 errors

—Circumvention of rigid type checking on calls to non-Pascal routines

BENEFITS

Provides a standard Pascal for iAPX 86, 88, 186, 188 based applications.

—Pascal has gained wide acceptance as the portable application language for microcomputer applications

—It is being taught in many colleges and universities around the world

—It is easy to learn, originally intended as a vehicle for teaching computer programming

—Improves maintainability: Type mechanism is both strictly enforced and user extendable

—Few machine specific language constructs

Strict implementation of the proposed ISO standard for Pascal aids portability of application programs. A compile time option checks conformance to the standard making it easy to write conforming programs.

PASCAL 86/88 extensions via predefined procedures for interrupt handling and direct port I/O make it possible to code an entire application in Pascal without compromising portability.

Standard Intel REALMATH is easy to use and provides reliable results, consistent with other Intel languages and other implementations of the IEEE proposed Floating-Point standard.

Provides run-time support for co-processors. All real-type arithmetic is performed on the 86/20 numeric data processor unit or software emulator. Run-time library routines, common between Pascal and other Intel languages (such as FORTRAN), permit efficient and consistently accurate results.

Extended relocation and linkage support allows the user to link Pascal program modules with routines written in other languages for certain parts of the program. For example, real-time or hardware dependent routines written in ASM 86/88/186/188 or PL/M 86/88/186/188 can be linked to Pascal routines, further extending the user's ability to write structured and modular programs.

PASCAL 86/88 programs "talk" to the resident operating system using Intel's standard interface for translated programs. This allows users to replace the development operating system by their own operating systems in the final application.

PASCAL 86/88 takes full advantage of iAPX 86, 88, 186, 188 high level language architecture to generate efficient machine code

Compiler options can be used to control the program listings and object modules. While debugging, the user may generate additional information such as the symbol record information required and useful for debugging using PSCOPE or ICE emulation. After debugging, the production version may be streamlined by removing this additional information.

AFN-01652B

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SPECIFICATIONS

Operating Environment

REQUIRED HARDWARE

Intel Microcomputer Development Systems (Series III, Series IV)

Documentation Package

PASCAL 86 User's Guide

ORDERING INFORMATION

Part Number	Description
MDS*-314	PASCAL 86/88 Software Package

Requires software license.

* MDS is an ordering code only and is not used as a product name or trademark. MDS® is a registered trademark of Mohawk Data Science.

SUPPORT:

Hotline Telephone Support, Software Performance Report (SPR), Software Updates, Technical Reports, and Monthly Technical Newsletters are available.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

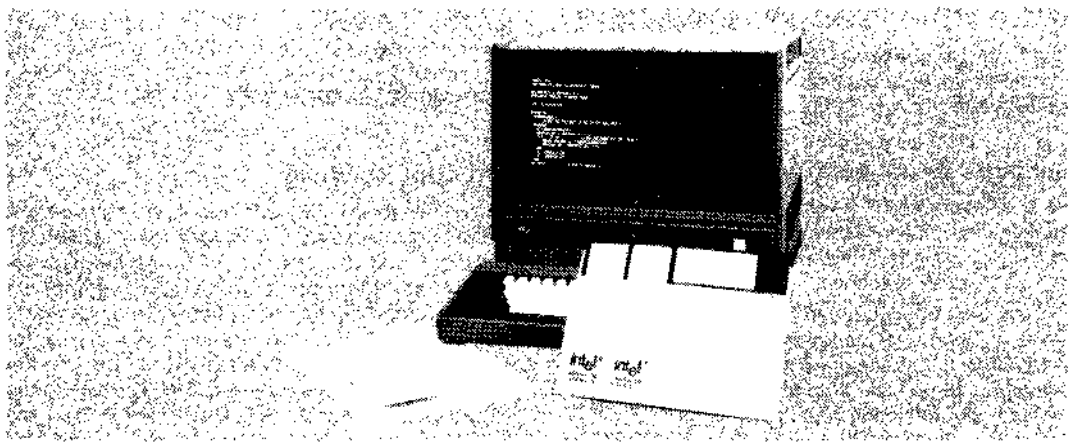
PL/M 86/88/186/188 Software Package

- **Systems Programming Language for the iAPX 86/88/186/188 Processors**
- **Language Is Upward Compatible from PL/M 80, Assuring MCS®-80/85 Design Portability**
- **Advanced Structured System Implementation Language for Algorithm Development**
- **Supports 16-Bit Signed Integer and 32-Bit Floating Point Arithmetic in Accordance with IEEE Proposed Standard**
- **Easy-to-Learn Block-Structured Language Encourages Program Modularity**
- **Improved Compiler Performance Now Supports More User Symbols and Faster Compilation Speeds**
- **Produces Relocatable Object Code Which Is Linkable to All Other 8086 Object Modules**
- **Code Optimization Assures Efficient Code Generation and Minimum Application Memory Utilization**
- **Built-In Syntax Checker Doubles Performance for Compiling Programs Containing Errors**
- **Resident on iAPX 86 Intel Microcomputer Development Systems**

PL/M 86 is an advanced, structured, high-level systems programming language. The PL/M 86 compiler was created specifically for performing software development for the Intel 8086, 8088, 80186 and 80188 Microprocessors. PL/M was designed so that program statements naturally express the program algorithm. This frees the programmer to concentrate on the logic of the program without concern for burdensome details of machine or assembly language programming (such as register allocation, meanings of assembler mnemonics, etc.).

The PL/M 86 compiler efficiently converts free-form PL/M language statements into machine instructions. Substantially fewer PL/M statements are necessary for a given application than if it were programmed at the assembly language or machine code level.

The use of PL/M high-level language for system programming, instead of assembly language, results in a high degree of engineering productivity during project development. This translates into significant reductions in initial software development and follow-up maintenance costs for the user.



NOTE: The Intellec® Development System pictured here is not included with the PL/M 86/88 Software package but merely depicts a language in its operating environment.

The following are trademarks of Intel Corporation and its affiliates and may be used only to identify Intel products: BXP, CREDIT, i, iCE, iCS, im, Insite, Intel, INTEL, Intelevision, Inteliink, Intellec, iMMX, iOSP, iPDS, iRMX, iSBC, iSBX, Library Manager, MCS, MULTIMODULE, Megachassis, Micromainframe, MULTIBUS, Multichannel, Plug-A-Bubble, PROMPT, Promware, RUPI, RMX/80, System 2000, UPI, and the combination iCS, iRMX, iSBC, iSBX, iCE, I2ICE, MCS, or UPI and numerical suffix. Intel Corporation Assumes No Responsibility for the use of Any Circuitry Other Than Circuitry Embodied in an Intel product. No Other Patent Licenses are implied. ©INTEL CORPORATION, 1983

MAY 1983

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FEATURES

Major features of the Intel PL/M 86 compiler and programming language include:

Block Structure

PL/M source code is developed in a series of modules, procedures, and blocks. Encouraging program modularity in this manner makes programs more readable, and easier to maintain and debug. The language becomes more flexible, by clearly defining the scope of user variables (local to a private procedure).

The use of procedures to break down a large problem is paramount to productive software development. The PL/M 86 implementation of a block structure allows the use of REENTRANT (recursive) procedures, which are especially useful in system design.

Language Compatibility

PL/M 86 object modules are compatible with object modules generated by all other iAPX 86 translators. This means that PL/M programs may be linked to programs written in any other iAPX 86 language.

Object modules are compatible with In-Circuit Emulators; DEBUG compiler control provides the In-Circuit Emulators with symbolic debugging capabilities.

PL/M 86 Language is upward compatible with PL/M 80, so that application programs may be easily ported to run on the iAPX 86.

Supports Seven Data Types

PL/M makes use of seven data types for various applications. These data types range from one to four bytes, and facilitate various arithmetic, logic, and addressing functions:

—Byte: 8-bit unsigned number
—Word: 16-bit unsigned number
—DWORD: 32-bit unsigned number
—Integer: 16-bit signed number
—Read: 32-bit floating point number
—Pointer: 16-bit or 32-bit memory address indicator
—Selector: 16-bit base portion of a pointer

Another powerful facility allows the use of BASED variables that map more than one variable to the same memory location. This is especially useful for passing parameters, relative and absolute addressing, and memory allocation.

Two Data Structuring Facilities

In addition to the five data types and based variables, PL/M supports two data structuring facilities. These help the user to organize data into logical groups.

— Array: Indexed list of same type data elements
— Structure: Named collection of same or different type data elements
— Combinations of Each: Arrays of structures or structures of arrays

8087 Numerics Support

PL/M programs that use 32-bit REAL data may be executed using the Numeric Data Processor for improved performance. All floating-point operations supported by PL/M may be executed on the iAPX 86/20 or 88/20 NDP, or the 8087 Emulator (a software module) provided with the package. Determination of use of the chip or Emulator takes place at linktime, allowing compilations to be run-time independent.

Built-In String Handling Facilities

The PL/M 86 language contains built-in functions for string manipulation. These byte and word functions perform the following operations on character strings: MOVE, COMPARE, TRANSLATE, SEARCH, SKIP, and SET.

Interrupt Handling

PL/M has the facility for handling interrupts. A procedure may be defined with the INTERRUPT attribute, and the compiler will automatically initialize an interrupt vector at the appropriate memory location. The compiler will also generate code to save and restore the processor status, for execution of the user-defined interrupt handler routine. The procedure SET$INTERRUPT, the function retuning an INTERRUPT$PTR, and the PL/M statement CAUSE$INTERRUPT all add flexibility to user programs involving interrupt and handling.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Compiler Controls

Including several that have been mentioned, the PL/M 86 compiler offers more than 25 controls that facilitate such features as:

— Conditional compilation
— Including additional PL/M source files from disk
— Corresponding assembly language code in the listing file
— Setting overflow conditions for run-time handling

Segmentation Control

The PL/M 86 compiler takes full advantage of program addressing with the SMALL, COMPACT, MEDIUM, and LARGE segmentation controls. Programs with less than 64KB total code space can exploit the most efficient memory addressing schemes, which lowers total memory requirements. Larger programs can exploit the flexibility of extended one-megabyte addressing.

Code Optimization

The PL/M 86 compiler offers four levels of optimization for significantly reducing overall program size.

— Combination or "folding" of constant expressions; and short-circuit evaluation of Boolean expressions
— "Strength reductions" (such as a shift left rather than multiply by 2); and elimination of common sub-expressions within the same block
— Machine code optimizations; elimination of superfluous branches; re-use of duplicate code; removal of unreachable code
— Byte comparisons (rather than 20-bit address calculations) for pointer variables; optimization of based-variable operations

Error Checking

The PL/M 86 compiler has a very powerful feature to speed up compilations. If a syntax or program error is detected, the compiler will skip the code generation and optimization passes. This usually yields a 2X performance increase for compilation of programs with errors.

A fully detailed set of programming and compilation errors is provided by the compiler.

```
M DO; /* Beginning of module */
      SORTPROC: PROCEDURE (PTR, COUNT, RECSIZE, KEYINDEX) PUBLIC;
                DECLARE PTR POINTER, (COUNT, RECSIZE, KEYINDEX) INTEGER;
      /* Parameters
         PTR is pointer to first record
         COUNT is number of records to be sorted
         RECSIZE is number of bytes in each record---max is 128
         KEYINDEX is byte position within each record of a BYTE scalar
                  to be used as sort key */
         DECLARE RECORD BASED PTR (1) BYTE
                 CURRENT (128) BYTE
                 (I, J) INTEGER;
SORT     DO J = 1 TO COUNT-1;
            CALL MOVB(@RECORD(J*RECSIZE), @CURRENT, RECSIZE);
            I = J;
FIND        DO WHILE I > 0;
               IF RECORD((I - 1)*RECSIZE + KEYINDEX) <
                  CURRENT(KEYINDEX)
               THEN DO;
                  CALL MOVB(@RECORD((I - 1)*RECSIZE),
                            @RECORD(I*RECSIZE),
                            RECSIZE);
                  I = I - 1;
               END FIND;
            CALL MOVB(@CURRENT, @RECORD(I*RECSIZE), RECSIZE);
         END SORT;
      END SORTPROC;
END M;    /* End of module */
```

PUBLIC and EXTERNAL attributes promote program modularity

Based Variables allow manipulation of external data by passing the base of the data structure (a pointer). This minimizes the STACK space used for parameter passing and the execution time to perform many STACK operations

The .AT. operator returns the *address* of a variable, instead of its contents. This is very useful in passing pointers for based variables

One of several PL/M built-in procedures for string manipulation

Figure 3. Sample PL/M 86 Program.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PL/M 86/88/186 SOFTWARE PACKAGE

BENEFITS

PL/M 86 is designed to be an efficient, cost-effective solution to the special requirements of iAPX 86 Microsystem Software Development, as illustrated by the following benefits of PL/M use:

Cost-Effective Alternative to Assembly Language

PL/M 86 programs are code efficient. PL/M 86 combines all of the benefits of a high-level language (ease of use, high productivity) with the ability to access the iAPX 86 architecture. Consequently, for the development of systems software, PL/M 86 is the cost-effective alternative to assembly language programming.

Low Learning Effort

PL/M is easy to learn and to use, even for the novice programmer.

Earlier Project Completion

Critical projects are completed much earlier than otherwise possible because PL/M 86, a structured high-level language, increases programmer productivity.

Lower Development Cost

Increases in programmer productivity translate immediately into lower software development costs because fewer programming resources are required for a given programmed function.

Increased Reliability

PL/M 86 is designed to aid in the development of reliable software (PL/M 86 programs are simple statements of the program algorithm). This substantially reduces the risk of costly correction of errors in systems that have already reached full production status, as the more simply stated the program is, the more likely it is to perform its intended function.

Easier Enhancements and Maintenance

Programs written in PL/M tend to be self-documenting, thus easier to read and understand. This means it is easier to enhance and maintain PL/M programs as the system capabilities expand and future products are developed.

SPECIFICATIONS

Operating Environment

REQUIRED HARDWARE:

Intel Microcomputer Development Systems (Series III/Series IV)

Documentation Package

PL/M-86 User's Guide for 8086-based Development Systems (121636)

SUPPORT:

Hotline Telephone Support, Software Performance Reporting (SPR), Software Updates, Technical Reports, Monthly Newsletter available.

ORDERING INFORMATION

Part Number	Description
MDS-313*	PL/M 86 Software Package

Requires Software License

*MDS is an ordering code only and is not used as a product name or trademark. MDS® is a registered trademark of Mohawk Data Sciences Corporation

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

iC-86
C COMPILER FOR THE 8086

- **Implements full C Language**
- **Produces high density code rivaling assembler**
- **Supports Intel Object Module Format (OMF)**
- **Runs under the Intel UDI on Intel Development Systems and iRMX™ 86**
- **Available for the VAX/VMS* Operating System**
- **Supports both small and large models of computation**
- **Supports PSCOPE-86 and I²ICE™**
- **Supports IEEE Floating Point Math with 8087 coprocessor**
- **Supports Bit Fields**
- **Supports full standard I/O Library (STDIO)**
- **Written in C**

The C Programming Language was originally designed in 1972 and has become increasingly popular as a systems development language. C is not a "very high level" language and is not tied to any specific application area. Although it is used for writing operating systems, it has been used equally well to write numerical, text-processing and data base programs. C combines the flexibility and programming speed of a higher level language with the efficiency and control of assembly language.

Intel iC-86 brings the full power of the C programming language to 8086 and 8088 based microprocessor systems.

Intel iC-86 supports the full C language as described in the Kernighan and Ritchie book, "The C Programming Lanugage," (Prentice-Hall, 1978). Also included are the latest enhancements to the C language: structure assignments, functions taking structure arguments and returning structures, and the "void" and "enum" data types.

C is rapidly becoming the standard microprocessor system implementation language because it provides:

1. the ability to manipulate the fundamental objects of the machine (including machine addresses) as easily as assembly language.
2. the power and speed of a structured language supporting a large number of data types, storage classes, expressions and statements,
3. processor independence (most programs developed for other processors can be easily transported to the 8086), and
4. code that rivals assembly language in efficiency

INTEL iC-86 COMPILER DESCRIPTION

The iC-86 compiler operates in four phases: preprocessor, parser, code generator, and optimizer. The preprocessor phase interprets directives in C source code, including conditional compilations (# define). The parser phase converts the C program into an intermediate free form and does all syntactic and semantic error checking. The code generator phase converts the parser's output into an efficient intermediate binary code, performs constant folding, and features an extremely efficient register allocator, ensuring high quality code. The optimizer phase converts the output of the code generator into

Intel Corporation Assumes No Responsibility for the Use of Any Circuitry Other Than Circuitry Embodied in an Intel Product No Other Circuit Patent Licenses are Implied Information Contained Herein Supercedes Previously Published Specifications of These Devices from Intel *VAX is a trademark of Digital Equipment Corporation
©INTEL CORPORATION, 1983

JUNE 1984

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

relocatable Intel Object Module Format (OMF) code, without creating an intermediate assembly file. Optionally, the iC-86 compiler can produce a symbolic assembly like file. The iC-86 optimizer eliminates common code, eliminates redundant loads and stores, and resolves span dependencies (shortens branches) within a program.

The iC-86 runtime library consists of a number of functions which the C programmer can call. The runtime system includes the standard I/O library (STDIO), conversion routines, routines for manipulating strings, special routines to perform functions not available on the 8086 (32-bit arithmetic and emulated floating point), and (where appropriate) routines for interfacing with the operating system.

iC-86 uses Intel's linker and locator and generates debug records for symbols and lines on request, permitting access to Intel's PSCOPE AND I²ICE™ to aid in program testing.

FEATURES

Support for Small and Large Models

Intel iC-86 supports both the SMALL and LARGE modes of segmentation. A SMALL model program can have up to 64K bytes of code and 64K bytes of data, with all pointers occupying two bytes. Because two byte pointers permit the generation of highly compact and efficient code, this model is recommended for programs that can meet the size restrictions. The LARGE segmentation model is used by programs that require access to the full addressing space of the 8086/8088 processors. In this model, each source file generates a distinct pair of code and data segments of up to 64K bytes in length. All pointers are four bytes long.

Preprocessor Directives

#define—defines a macro

#include—includes code outside of the program source file

#if—conditionally includes or excludes code

Other preprocessor directives include #undef, #ifdef, #ifndef, #else, #endif, and #line.

Statements

The C language supports a variety of statements:

Conditionals: IF, IF-ELSE

Loops: WHILE, DO-WHILE, FOR

Selection of cases: SWITCH, CASE, DEFAULT

Exit from a function: RETURN

Loop control: CONTINUE, BREAK

Branching: GOTO

Expressions and Operators

The C language includes a rich set of expressions and operators.

Primary expression: invoke functions, select elements from arrays, and extract fields from structures or unions

Arithmetic operators: add, subtract, multiply, divide, modulus

Relational operators: greater than, greater than or equal, less than, less than or equal, not equal

Unary operators: indirect through a pointer, compute an address, logical negation, ones complement, provide the size in bytes of an operand.

Logical operators: AND, OR

Bitwise operators: AND, exclusive OR, inclusive OR, bitwise complement

Data Types and Storage Classes

Data in C is described by its type and storage class. The type determines its representation and use, and the storage class determines its lifetime, scope, and storage allocation. The following data types are fully supported by iC-86.

char
an 8 bit signed integer

int
a 16 bit signed integer

short
same as int (on the 8086)

long
a 32 bit signed integer

unsigned
a modifier for integer data types (char, int, short, and long) which doubles the positive range of values

float
a 32 bit floating point number which utilizes the 8087 or a software floating point library

double
a 64 bit floating point number

AFN-00144C

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

void
a special type that cannot be used as an operand in expressions; normally used for functions called only for effect (to prevent their use in contexts where a value is required).

enum
an enumerated data type

These fundamental data types may be used to create other data types including: arrays, functions, structures, pointers, and unions.

The storage classes availabe in iC-86 include:

register
suggests that a variable be kept in a machine register, often enhancing code density and speed

extern
a variable defined outside of the function where it is declared; retaining its value throughout the entire program and accessible to other modules

auto
a local variable, created when a block of code is entered and discarded when the block is existed

static
a local variable that retains its value until the termination of the entire program

typedef
defines a new data type name from existing data types

BENEFITS

Faster Compilation

Intel iC-86 compiles C programs substantially faster than standard C compilers because it produces Intel OMF code directly, eliminating the traditional intermediate process of generating an assembly file.

Portability of Code

Because Intel iC-86 supports the STDIO and produces Intel OMF code, programs developed on a variety of machines can easily be transported to the 8086.

Rapid Program Development

Intel iC-86 provides the programmer with detailed error messages and access to PSCOPE-86 and I²ICE™ to speed program development.

Full Manipulation of the 8086

Intel iC-86 enables the programmer to utilize features of the C language to control bit fields, pointers, addresses and register allocation, taking full advantage of the fundamental concepts of the 8086.

SPECIFICATIONS

Operating Environment

The iC-86 compiler runs host resident on both the Intel Series III Microcomputer Development System under ISIS-II and on the System 86/330 under the iRMX™ 86 operating system. iC-86 can also run as a cross compiler on a VAX 11/780 computer under the VMS operating system 128 KBytes of User Memory is required on all versions. Specify desired version when ordering.

Required Hardware

Development System Version

—Intellec® Microcomputer Development System; Series III or Series IV

—Dual Diskette Drives, Single or Double Density

—System Console; CRT or Hardcopy Interactive Device

iRMX 86 version:

—Any iAPX 86/88, iSBC® 86/88, iTPS 86/XXX, or SYS 86/3XX based system capable of running the iRMX 86 Operating System

VAX version:

—Digital Equipment Corporation VAX 11/780 or compatible computer

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Optional Hardware

ISIS-II version:

—ICE-86, I²ICE-86

iRMX 86 version:

—Numeric Data Processors for support of the REALMATH standard

VAX version:

—None

Required Software

ISIS-II version:

—ISIS-II Diskette Operating System

—Series III or Series IV Operating System

iRMX 86 version:

—iRMX 86 Realtime Multiprogramming Operating System

—iRMX 860 Utilities Package

VAX version:

—VMS Operating System

Optional Software

Development System Version:

—None

iRMX 86 version:

—None

VAX version:

—MDS*-384 Kit-Mainframe Link for distributed development, or iMDX-394 Asynchronous Communications Link.

—VAX iAPX 86/88/186 MACRO Assembler and utilities package (iMDX-341VX)

Documentation Package

The C Programming Language by Kernighan and Ritchie (1978 Prentice-Hall)

iC-86 User Manual

Shipping Media

Development System Version:

—Two single and one double density ISIS-II format 8" diskettes, one 5 1/4" Series IV Format

iRMX 86 version:

—Double Density iRMX 86 format 8" diskette

—Double Density iRMX 86 format 5¼" diskette

VAX version:

—1600 bpi, 9 track Magnetic tape

ORDERING INFORMATION

Order Code	Description
iMDX-317	iC-86 Compiler for ISIS-II
iRMX-866	iC-86 Compiler for iRMX 86
iMDX-347	iC-86 Cross Compiler for VAX/VMS

Intel Software License required

SUPPORT

Intel offers several levels of support for this product which are explained in detail in the price list. Please consult the price list for a description of the support options available.

*MDS is an ordering code only and is not used as a product name or trademark. MDS is a registered trademark of Mohawk Data Sciences Corporation.

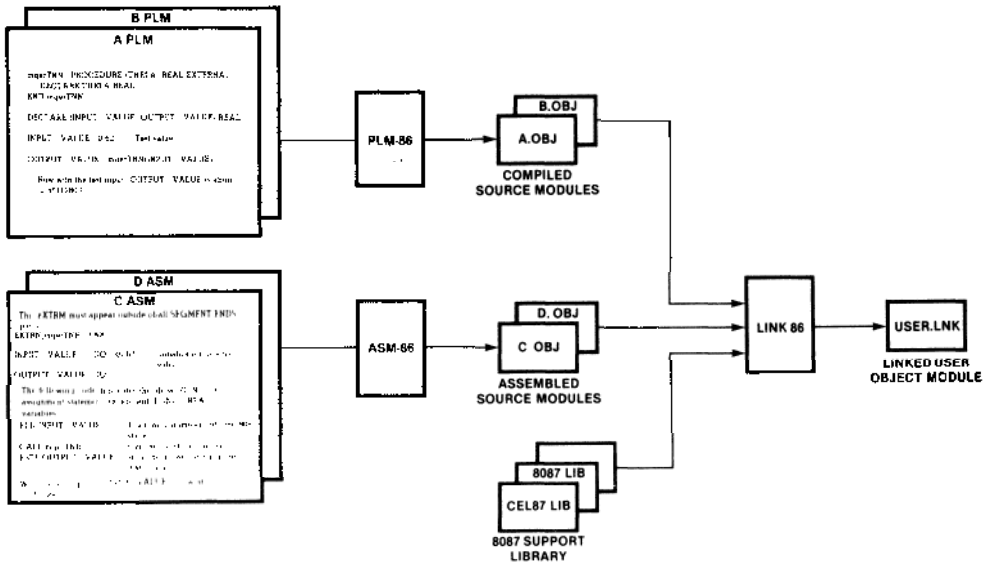
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8087 SUPPORT LIBRARY

- **Library to support floating point arithmetic in PL/M-86 and ASM-86**
- **Common elementary function library provides trigonometric, logarithmic and other useful functions**
- **Decimal conversion module supports binary-decimal conversions**
- **Error-handler module simplifies floating point error recovery**
- **Full 8087 Software Emulator for software debugging without the 8087 component**
- **Accurate, verified and efficient implementation of algorithms for functions**
- **Supports proposed IEEE Floating Point Standard for high accuracy and software portability**

The 8087 Support Library provides PL/M-86 and ASM-86 users with the equivalent numeric data processing capability of Fortran-86. With the Library, it is easy for PL/M-86 and ASM-86 programs to do floating point arithmetic. Programs can link in modules to do trigonometric, logarithmic and other numeric functions, and the user is guaranteed accurate, reliable results for all appropriate inputs. The 8087 Support Library implements Intel's REALMATH standard and also supports the proposed IEEE Floating Point Standard. Consequently, by using this Library, the PL/M-86 user not only saves software development time, but is guaranteed that the numeric software meets industry standards and is portable—his software investment is maintained.

The 8087 Support Library consists of the common elementary function library, the decimal conversion module, the error handler module, the full 8087 Software emulator and interface libraries to the 8087 and to the 8087 emulator.



: INTEL CORPORATION 1983

MAY 1983
ORDER NUMBER:121653-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

CEL87.LIB
THE COMMON ELEMENTARY FUNCTION LIBRARY

CEL87.LIB contains commonly used floating point functions. It is used along with the 8087 numeric coprocessor or the 8087 emulator and it provides a complete package of elementary functions, giving valid results for all appropriate inputs. This library provides PL/M-86 and ASM-86 users all the math functions supported intrinsically by the Fortran-86. Following is a summary of CEL87 functions, grouped by functionality.

Rounding and Truncation Functions:

mqerIEX, mqerIE2, and mqerIE4 round a real number to the nearest integer; to the even integer if there is a tie. The answer returned is real, a 16-bit integer or a 32-bit integer respectively.
mqerIAX; mqerIA2, mqerIA4 round a real number to the nearest integer, to the integer away from zero if there is a tie; the answer returned is real, a 16-bit integer or a 32-bit integer, respectively.
mqerICX, mqerIC2, mqerIC4 truncate the fractional part of a real input; the answer is real, a 16-bit integer or a 32-bit integer, respectively.

Logarithmic and Exponential Functions:

mqerLGD computes decimal (base 10) logarithms.
mqerLGE computes natural (base e) logarithms.
mqerEXP computes exponentials to the base e.
mqerY2X computes exponentials to any base.
mqerYI2 raises an input real to a 16-bit integer power.
mqerYI4 is as mqerYI2, except to a 32-bit integer power.
mqerYIS is as mqerYI2, but it accommodates PL/M-86 users.

Trigonometric and Hyperbolic Functions:

mqerSIN, mqerCOS, mqerTAN compute sine, cosine, and tangent.
mqerASN, mqerACS, mqerATN compute the corresponding inverse functions.
mqerSNH, mqerCSH, mqerTNH compute the corresponding hyperbolic functions.
mqerAT2 is a special version of the arc tangent function that accepts rectangular coordinate inputs.

Other Functions:

mqerDIM is FORTRAN's positive difference function.
mqerMAX returns the maximum of two real inputs.
mqerMIN returns the minimum of two real inputs.
mqerSGH combines the sign of one input with the magnitude of the other input.
mqerMOD computes a modulus, retaining the sign of the dividend.
mqerRMD computes a modulus, giving the value closest to zero.

DCON87.LIB
THE DECIMAL CONVERSION LIBRARY

DCON87.LIB is a library of procedures which convert binary representations of floating point numbers and ASCII-encoded string of digits.

The binary-to-decimal procedure mqcBIN DECLOW accepts a binary number in any of the formats used for the representation of floating point numbers in the 8087. Because there are so many output formats for floating point numbers, mqcBIN_DECLOW does not attempt to provide a finished, formatted text string. Instead, it provides the "building blocks" for you to use to construct the output string which meets your exact format specification.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

The decimal-to-binary procedure mqcDEC_BIN accepts a text string which consists of a decimal number with optional sign, decimal point, and/or power-of-ten exponent. It translates the string into the caller's choice of binary formats.

Decimal-to-binary procedure mqcDECLOW_BIN is provided for callers who have already broken the decimal number into its constituent parts.

The procedures mqcLONG_TEMP, mqcSHORT_TEMP, mqcTEMP_LONG, and mqcTEMP_SHORT convert floating point numbers between the longest binary format, TEMP_REAL, and the shorter formats.

EH87.LIB
THE ERROR HANDLER MODULE

EH87.LIB is a library of five utility procedures which a user can utilize for writing trap handlers. Trap handlers are called when an unmasked 8087 error occurs.

The 8087 error reporting mechanism can be used not only to report error conditions, but also to let software implement IEEE standard options not directly supported by the chip. The three such extensions to the 8087 are: normalizing mode, non-trapping not-a-number (NaN), and non-ordered comparison. The utility procedures support these extra features.

DECODE is called near the beginning of the trap handler. It preserves the complete state of the 8087, and also identifies what function called the trap handler, and returns available arguments and/or results. DECODE eliminates much of the effort needed to determine what error caused the trap handler to be called.

NORMAL provides the "normalizing mode" capability for handling the "D" exception. By calling NORMAL in your trap handler, you eliminate the need to write code in your application program which tests for non-normal inputs.

SIEVE provides two capabilities for handling the "I" exception. It implements non-trapping NaN's and non-ordered comparisons. These two IEEE standard features are useful for diagnostic work.

ENCODE is called near the end of the trap handler. It restores the state of the 8087 saved by DECODE, and performs a choice of concluding actions, by either retrying the offending function or returning a specified result.

FILTER calls each of the above four procedures. If your error handler does nothing more than detect fatal errors and implement the features supported by SIEVE and NORMAL, then your interface to EH87.LIB can be accomplished with a single call to FILTER.

E8087
THE FULL 8087 EMULATOR

E8087 is an object module that functionally emulates the 8087 coprocessor chip. It is ideal for use during prototyping and debugging floating point programs. However, the target system should use the 8087 component because it executes 1000 times faster and uses significantly less memory.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

E8087.LIB, 8087.LIB, 87NULL.LIB INTERFACE LIBRARIES

E8087. LIB, 8087.LIB and 87NULL. LIB libraries configure a user's application program for his run-time environment: running with the emulator, with the 8087 component or without floating point arithmetic, respectively.

SPECIFICATIONS

TARGET ENVIRONMENT

8086/8088 Based Microcomputer System

DEVELOPMENT ENVIRONMENT

Required Hardware

All Intel Microcomputer Development Systems (Series II, Series III/Series IV)

Required Software

For Series II:
8086/8088 Software Development Package

Documentation Package

Numeric Support Library Manual

*Recommended

ORDERING INFORMATION

Part Number	Description
MDS*-319	8087 Support Library

Requires Software License

SUPPORT

Intel offers several levels of support for this product which are explained in detail in the price list. Please consult the price list for a description of the support options available.

*MDS is an ordering code only and is not used as a product name or trademark. MDS is a registered trademark of Mohawk Data Sciences Corporation.

AFN-02063B

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8087
SOFTWARE SUPPORT PACKAGE

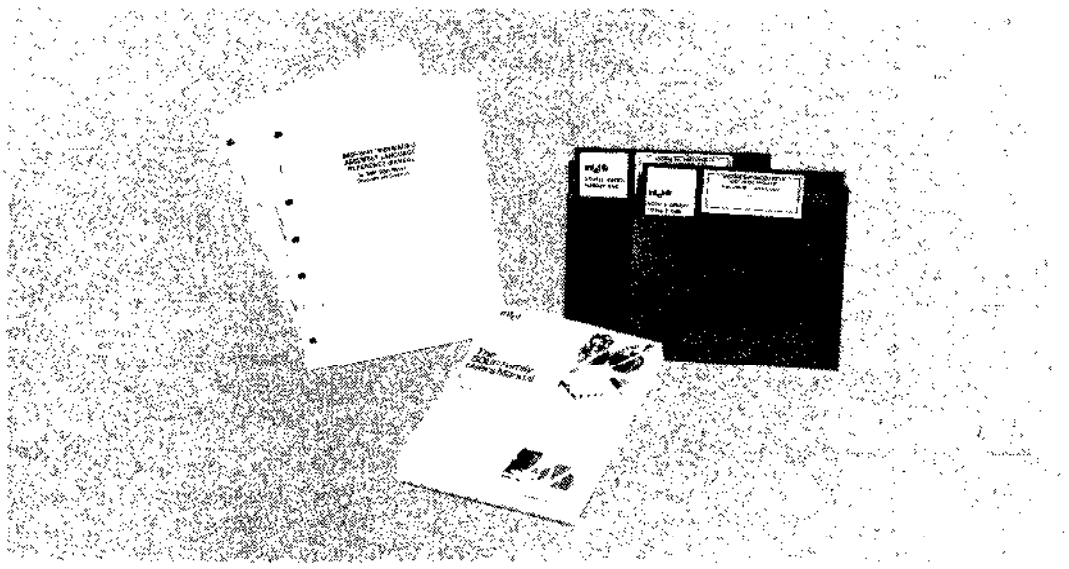
- **Program Generation for the 8087 Numeric Data Processor on 8080/8085 Based Intel Microcomputer Development Systems**
- **Consists of: 8086/8087/8088 Macro Assembler, 8087 Software Emulator**
- **Macro Assembler Generates Code for 8087 Processor or Emulator, While Also Supporting the 8086/8088 Instruction Set**
- **8087 Emulator Duplicates Each 8087 Floating-Point Instruction in Software, for Evaluation of Prototyping, or for Use in an End Product**
- **Macro Assembler and 8087 Emulator are Fully Compatible with Other 8086/8088 Development Software**
- **Implementation of the IEEE Proposed Floating-Point Standard (the Intel® Realmath Standard)**

The 8087 Software Support Package is an optional extention of Intel's 8086/8088 Software Development Package.

The 8087 Software Support Package consists of the 8086/8087/8088 Macro Assembler, and the Full 8087 Emulator. The assembler is a functional superset of the 8086/8088 Macro Assembler, and includes instructions for over sixty new floating-point operations, plus new data types supported by the 8087.

The 8087 Emulator is an 8086/8088 object module that simulates the environment of the 8087, and executes each floating-point operation using software algorithms. This emulator functionally duplicates the operation of the 8087 Numeric Data Processor.

Also included in this package are interface libraries to link with 8086/8087/8088 object modules, which are used for specifying whether the 8087 Processor or the 8087 Emulator is to be used. This enables the run-time environment to be invisible to the programmer at assembly time.



The following are trademarks of Intel Corporation and may be used only to identify Intel products: BXP, CREDIT, Intellec, Multibus, I, iSBC, Multimodule, ICE, iSBX, PROMPT, iRMX, iCS, Library Manager, Promware, Insite, MCS, RMX, Intel, Megachassis, UPI, Intelevision, Micromap, μScope and the combination of ICE, iCS, iSBC, iSBX, MCS, or RMX and a numerical suffix
© INTEL CORPORATION, 1983

SEPTEMBER 1984
ORDER NUMBER:4042150-002

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FUNCTIONAL DESCRIPTION

8086/8087/8088 Macro Assembler

The 8086/8087/8088 Macro Assembler translates symbolic macro assembly language instructions into appropriate machine instructions. It is an extended version of the 8086/8088 Macro Assembler, and therefore supports all of the same features and functions, such as limited type checking, conditional assembly, data structures, macros, etc. The extensions are the new instructions and data types to support floating-point operations. Realmath floating-point instructions (see Table 1) generate code capable of being converted to either 8087 instructions or interrupts for the 8087 Emulator. The Processor/Emulator selection is made via interface libraries at LINK-time. In addition to the new floating-point instructions, the macro assembler also introduces two new 8087 data types: QWORD (8 bytes) and TBYTE (ten bytes). These support the highest precision of data processed by the 8087.

Full 8087 Emulator

The Full 8087 Emulator is a 16-kilobyte object module that is linked to the application program for floating-point operations. Its functionality is identical to the 8087 chip, and is ideal for prototyping and debugging floating-point applications. The Emulator is an alternative to the use of the 8087 chip, although the latter executes floating-point applications up to 100 times faster than an 8086 with the 8087 Emulator. Furthermore, since the 8087 is a "co-processor," use of the chip will allow many operations to be performed in parallel with the 8086.

Table 1. 8087 Instructions

Arithmetic Instructions

Addition	
FADD	Add real
FADDP	Add real and pop
FIADD	Integer add
Subtraction	
FSUB	Subtract real
FSUBP	Subtract real and pop
FISUB	Integer subtract
FSUBR	Subtract real reversed
FSUBRP	Subtract real reversed and pop
FISUBR	Integer subtract reversed
Multiplication	
FMUL	Multiply real
FMULP	Multiply real and pop
FIMUL	Integer multiply
Division	
FDIV	Divide real
FDIVP	Divide real and pop
FIDIV	Integer divide
FDIVR	Divide real reversed
FDIVRP	Divide real reversed and pop
FIDIVR	Integer divide reversed
Other Operations	
FSQRT	Square root
FSCALE	Scale
FPREM	Partial remainder
FRNDINT	Round to integer
FXTRACT	Extract exponent and significand
FABS	Absolute value
FCHS	Change sign

Processor Control Instructions

FINIT/FNINIT	Initialize processor
FDISI/FNDISI	Disable interrupts
FENI/FNENI	Enable interrupts
FLDCW	Load control word
FSTCW/FNSTCW	Store control word
FSTSW/FNSTSW	Store status word
FCLEX/FNCLEX	Clear exceptions
FSTENV/FNSTENV	Store environment
FLDENV	Load environment
FSAVE/FNSAVE	Save state
FRSTOR	Restore state
FINCSTP	Increment stack pointer
FDECSTP	Decrement stack pointer
FFREE	Free register
FNOP	No operation
FWAIT	CPU wait

Comparison Instructions

FCOM	Compare real
FCOMP	Compare real and pop
FCOMPP	Compare real and pop twice
FICOM	Integer compare
FICOMP	Integer compare and pop
FTST	Test
FXAM	Examine

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 1. 8087 Instructions (cont'd)

Transcendental Instructions

FPTAN	Partial tangent
FPATAN	Partial arctangent
F2XM1	$2^x - 1$
FYL2X	$Y \bullet \log_2 X$
FYL2XP1	$Y \bullet \log_2(X+1)$

Constant Instructions

FLDZ	Load +0.0
FLD1	Load +1.0
FLDPI	Load π
FLDL2T	Load $\log_2 10$
FLDL2E	Load $\log_2 e$
FLDLG2	Load $\log_{10} 2$
FLDLN2	Load $\log_e 2$

Data Transfer Instructions

Real Transfers	
FLD	Load real
FST	Store real
FSTP	Store real and pop
FXCH	Exchange registers
Integer Transfers	
FILD	Integer load
FIST	Integer store
FISTP	Integer store and pop
Packed Decimal Transfers	
FBLD	Packed decimal (BCD) load
FBSTP	Packed decimal (BCD) store and pop

SPECIFICATIONS

Operating Environment

REQUIRED HARDWARE

Intel Microcomputer Development Systems
—Series II
—Personal Development System
—Series IV

REQUIRED SOFTWARE

8086/8088 Software Development Package

Documentation Package

8086/8087/8088 Macro Assembly Language Reference Manual for 8080/8085-Based Development Systems

8086/8087/8088 Macro Assembler Operating Instructions for 8080/8085-Based Development Systems

The 8086 Family Users Manual Supplement for the 8087 Numeric Data Processor

ORDERING INFORMATION

Part Number	Description
MDS*-387	8087 Software Support Package

Requires Software License

SUPPORT

Intel offers several levels of support for this product which are explained in detail in the price list. Please consult the price list for a description of the support options available.

*MDS is an ordering code only and is not used as a product name or trademark. MDS is a registered trademark of Mohawk Data Sciences Corporation.

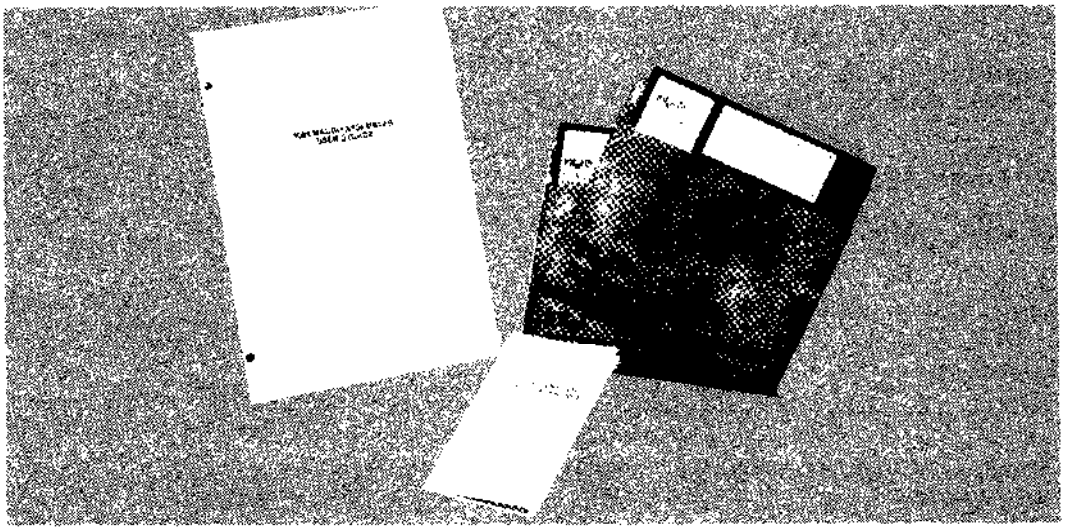
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8089 IOP
SOFTWARE SUPPORT PACKAGE
#407200

- **Program Generation for the 8089 I/O Processor on the Intellec® Microcomputer Development System**
- **Contains 8089 Macro Assembler, plus Relocation and Linkage Utilities**
- **Relocatable Object Module Compatible with All iAPX 86 and iAPX 88 Object Modules**
- **Fully Supports Symbolic Debugging with the RBF-89 Software Debugger**
- **Supports 8089-Based Addressing Modes with a Structure Facility that Enables Easy Access to Based Data**
- **Powerful Macro Capabilities**
- **Provides Timing Information in Assembly Listing**
- **Fully Detailed Set of Error Messages**

The IOP Software Support Package extends Intellec Microcomputer Development System support to the 8089 I/O Processor. The macro assembler translates symbolic 8089 macro assembly language instructions into relocatable machine code. The relocation and linkage utilities provide compatibility with iAPX 86, iAPX 88, and 8089 modules, and make structured, modular programming easier.

The macro assembler also provides symbolic debugging capability when used with the RBF-89 software debugger. 8089 program modularity is supported with inter-segment jumps and calls. The macro assembler also provides instruction cycle counts in the listing file, for giving the programmer execution timing information. The programs in the 8089 Software Support Package run on any Intellec Series II or Model 800 with 64K bytes of memory.



The following are trademarks of Intel Corporation and may be used only to identify Intel products: BXP, CREDIT, Intellec, Multibus, i, iSBC, Multimodule, ICE, iSBX, PROMPT, ICS, iRMX, Library Manager, Promware, Insite, MCS, RMX, Intel, Megachassis, UPI, Intelevision, Micromap, μScope and the combination of ICE, iSBC, iSBX, MCS, or RMX and a numerical suffix

MAY 1983

© INTEL CORPORATION 1983

ORDER NUMBER:210653-002

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 1. Sample Program Listing

```
8089 MACRO ASSEMBLER

[SIS-II 8089 MACRO ASSEMBLER V1.0 ASSEMBLY OF MODULE TASK
OBJECT MODULE PLACED IN  :F1:TA89.OBJ
ASSEMBLER INVOKED BY:  as89  :f1:task.a89 gen macro debug page(,,60) print(:f1:task.lst)


LOC    OBJECT CODE             TIMING    INC AMC  LINE SOURCE

                                            1  ;*****************************************
                                            2  ;*                                       *
                                            3  ;*              8089 TASK PROGRAM        *
                                            4  ;*                                       *
                                            5  ;*****************************************
                                            6
                                            7 name TASK
0000                                        8 TASK segment
                                            9
                                           10          In the first part of this simple program data is moved from
                                           11 ;        8086 system RAM to memory local to the 8089 IOP.  In the second
                                           12 ;        part, the data is moved from the local memory to a data port
                                           13 ;        also in the 8089 I/O space
                                           14
  1888                                     15 data@port@8251  equ     0c888h          ;8251 DP on 8089 local bus
  888A                                     16 command@port@8251 equ   0c88Ah          ;8251 CP on 8089 local bus
  8200                                     17 buffer@8089     equ     8200h          ;RAM buffer in 8089 I/O space
                                           18
                                           19 extrn   buffer@8086                    ;RAM buffer in 8086 system memory
                                           20 extrn   y                              ;location of the buffer count
                                           21 ;
                                           22 %*define (macro_1)
                                           23 (       mov     gb,buffer@8089         ; Move buffer address into GB
                                           24          lpd     gc,y                   ; Load pointer to count into GC
                                           25          movb    bc,[gc]                ; Move byte count into BC
                                           26 )
                                           27 %*define (macro_2(param_1, param_2)) local loop
                                           28 (        inc     %param_1               ; Increment pointer into source
                                           29          dec     %param_2               ; Decrement byte count
                                           30          jnz     %param_2,%loop         ; Loop back if byte count > 0
                                           31 )
0000   1180 00806806                       32 ONE:    lpd     ga,buffer@8086         ; Load register GA with address
                                           33                                        ; of 8086 buffer
                                           34 %macro_1
0006   3138 0002                          35          mov     gb,buffer@8089         ; Move buffer address into GB
0008   5189 0000C000                      36          lpd     gc,y                   ; Load pointer to count into GC
0010   6002                              37          movb    bc,[gc]                ; Move byte count into BC
                                           38
0012   0E50 00C0                          39 loop@81: movb    [gb],[ga]              ; Move byte from 8086 to 8089 buffer
0016   0E30                              40          inc     ga                     ; Increment pointer into 8086 buffer
                                           41 %macro_2(gb,gc)
0018   2E3B                              42          inc     %PARAM_1
                                           43 ;    gb                             ; Increment pointer into source
001A   4E3C                              44          dec     %PARAM_2
                                           45 ;    gc                             ; Decrement byte count
001C   404B F3                           46          jnz     %PARAM_2
                                           47 ;    gc,%LOOP
                                           48          LOOPBB          ; Loop back if byte count > 0
                                           49
001F   1138 00C8                         50 TWO:    mov     ga,data@port@8251      ; load GA with address of 8251 DP
0023   1130 01C0                         51          mov     gb,command@port@8251   ; load GC with address of 8251 CP
                                           52 %macro_1
0027   3138 0002                          53          mov     gb,buffer@8089         ; Move buffer address into GB
0029   5180 00000000                      54          lpd     gc,y                   ; Load pointer to count into GC
0031   6002                              55          movb    bc,[gc]                ; Move byte count into BC
                                           56
0033   A00A FD                           57 loop@1: jnot    [ga],0,loop@1          ; loop until 8251 transmit ready
0036   0E91 00CD                         58          movb    [ga],[gb]              ; move message into buffer
                                           59 %macro_2(gb,gc)
0034   2E30                              60          inc     %PARAM_1
                                           61 ;    gb                             ; Increment pointer into source
003C   4E3C                              62          dec     %PARAM_2
                                           63 ;    gc                             ; Decrement byte count
003E   4048 F2                           64          jnz     %PARAM_3
                                           65 ;    gc,%LOOP
                                           66          LOOPBB          ; Loop back if byte count > 0
                                           67
0041   2E40                              68          hlt
0043                                       69 TASK    ends
                                           70 END

ASSEMBLY COMPLETE, NO ERRORS FOUND
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FUNCTIONAL DESCRIPTION

The IOP Software Support Package contains:

ASM89 —The 8089 Macro Assembler.

LINK86 — Resolves control transfer references between 8089 object modules, and data references in 8086, 8088, and 8089 modules.

LOC86 —Assigns absolute memory addresses to 8089 object modules.

OH86 —Converts absolute object modules to hexadecimal format.

UPM —The Universal PROM Mapper, which supports PROM programming in all iAPX 86/11 and iAPX 88/11 applications.

ASM89 translates symbolic 8089 macro assembly language instructions into the appropriate machine codes. The ability to refer to both program and data addresses with symbolic names makes it easier to develop and modify programs, and avoids the errors of hand translation.

The powerful macro facility allows frequently used code sequences to be referred to by a single name, so that any changes to that sequence need to be made in only one place in the program. Common code sequences that differ only slightly can also be referred to with a macro call, and the differences can be substituted with macro parameters.

ASM89 provides symbolic debugging information in the object file. The RBF-89 debugger makes use of this information, so the programmer can symbolically debug 8089 programs. ASM89 also provides cycle counts for each instruction in the assembly listing file (see Table 1). These cycle counts help the programmer determine how long a particular routine or code sequence will take to execute on the 8089.

ASM89 provides relocatable object module compatibility with the 8086 and 8088 microprocessors. This object module compatibility, along with the 8086/8088 relocation and linkage utilities, facilitates the designing of iAPX 86/11 and iAPX 88/11 systems.

ASM89 fully supports the based addressing modes of the 8089. A structure facility allows the user to define a template that enables accessing of based data symbolically.

SPECIFICATIONS

Operating Environment

Intel Microcomputer Development Systems (Model 800, Series II, Series III, Series IV)

Support

Hotline Telephone Support, Software Performance Report (SPR), Software Updates, Technical Reports, and Monthly Technical Newsletters are available.

Documentation Package

8089 Macro Assembler User's Guide (9800938)

8089 Macro Assembler Pocket Reference (9800936)

MCS-86 Software Development Utilities Operating Instructions for ISIS-II Users (9800639)

Universal PROM Programmer User's Manual (9800819)

Shipping Media

—Single and Double Density Diskettes

ORDERING INFORMATION

Part Number	Description
MDS*-312	8089 IOP Software Support Package

Requires Software License

*MDS is an ordering code only and is not used as a product name or trademark. MDS® is a registered trademark of Mohawk Data Sciences Corporation

AFN-00840BC

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

iAPX 286 SOFTWARE DEVELOPMENT PACKAGE

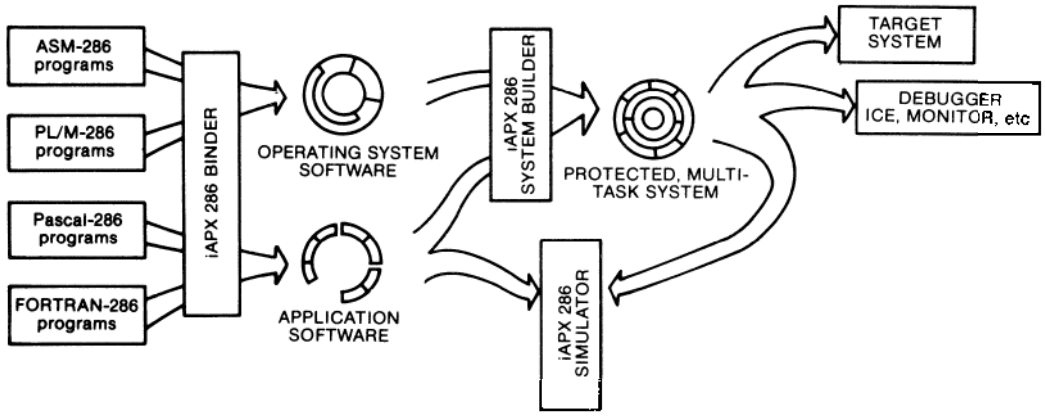
- **Complete System Development Capability for High-Performance iAPX 286 Applications.**
- **Allows creation of Multi-User, Virtual Memory, and Memory-Protected Systems.**
- **Macro Assembler for Machine-Level Programming.**
- **System Utilities for Program Linkage and System Building.**
- **Software Simulator for Execution and Symbolic Debugging on Intel Development System.**
- **Package Supports Program Development with PL/M-286, Pascal-286, and FORTRAN 286.**
- **Extends Existing Intellec® Development Systems to Provide Broad Support for the iAPX 286 Microprocessor.**

The iAPX 286 is a 16-bit microprocessor system with 32-bit virtual addressing, integrated memory protection, and instruction pipelining for high performance. The iAPX 286 Software Development Package is a cohesive set of software design aids for programming the iAPX 286 microprocessor system. The package enables system programmers to design protected, multi-user and multi-tasking operating system software, and enables application programmers to develop tasks to run on a protected operating system.

The iAPX 286 Software Development package contains a macro assembler, a program binder (for linking separately compiled modules together), a system builder (for configuring protected multiple-task systems), and a software simulator (for execution and symbolic debugging).

The memory protection features of the iAPX 286 architecture are invisible to application programmers, who use language translators and the program binder. System programmers may use special memory protection features in ASM-286 or PL/M 286, and use the system builder for initializing and managing protection features. The Simulator duplicates the operation of the 80286 CPU, as well as the floating point operations of the 80287.

All the utilities in the Software Development Package run on the Intel Microcomputer Development Systems (Series III/Series IV)



The iAPX 286 Software Development Package keeps the protection mechanism invisible to the application programmer, yet easy to configure for the system programmer.

Intel Corporation Assumes No Responsibility for the Use of Any Circuitry Other Than Circuitry Embodied in an Intel Product. No Other Circuit Patent Licenses are Implied. Information Contained Herein Supercedes Previously Published Specifications of These Devices from Intel.
©INTEL CORPORATION, 1983

JUNE 1984
ORDER NUMBER: 210565-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

iAPX 286 MACRO ASSEMBLER

- **Instruction Set and Assembler Mnemonics Are Upward Compatible with ASM-86/88.**
- **Powerful and Flexible Text Macro Facility.**
- **Type-Checking at Assembly Time Helps Reduce Errors at Run-Time.**
- **Structures and RECORDS Provide Powerful Data Representation.**
- **"High-Level" Assembler Mnemonics Simplify the Language.**
- **Supports Full Instruction Set of the iAPX 286/20, Including Memory Protection and Numerics.**

ASM-286 is the "high-level" macro assembler for the iAPX 286 assembly language. ASM-286 translates symbolic assembly language mnemonics into relocatable object code. The assembler mnemonics are a superset of ASM-86/88 mnemonics; new ones have also been added to support the new iAPX 286 instructions. The segmentation directives have been greatly simplified.

The iAPX 286 assembly language includes approximately 150 instruction mnemonics. From these few mnemonics the assembler can generate over 4,000 distinct machine instructions. Therefore, the software development task is simplified, as the programmer need know only 150 mnemonics to generate all possible machine instructions. ASM-286 will generate the shortest machine instruction possible (given explicit information as to the characteristics of any forward referenced symbols).

The powerful macro facility in ASM-286 saves development and maintenance time by coding common program sequences only once. A macro substitution is made each time the sequence is to be used. This facility also allows for conditional assembly of certain program sequences.

ASM-286 offers many features normally found only in high-level languages. The assembly language is strongly typed, which means it performs extensive checks on the usage of variables and labels. This means that many programming errors will be detected when the program is assembled, long before it is being debugged.

ASM-286 object modules conform to a thorough, well-defined format used by all 286 high-level languages and utilities. This makes it easy to call (and be called from) HLL object modules.

Key Benefit:

For programmers who wish to use assembly language, ASM-286 provides many powerful "high-level" capabilities that simplify program development and maintenance.

AFN-00376B

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

iAPX 286 BINDER

- **Links Separately Compiled Program Modules Into an Executable Task.**
- **Makes the iAPX 286 Protection Mechanism Invisible to Application Programmers.**
- **Works with PL/M-286, Pascal-286, FORTRAN-286 and ASM-286 Object Modules.**
- **Performs Incremental Linking with Output of Binder and Builder.**
- **Resolves PUBLIC/EXTERNAL Code and Data References, and Performs Intermodule Type-Checking.**
- **Provides Print File Showing Segment Map, Errors and Warnings.**
- **Assigns Virtual Addresses to Tasks in the 2^{32} Address Space.**
- **Generates Linkable or Loadable Module for Debugging.**

BND-286 is a utility that combines iAPX 286 object modules into executable tasks. In creating a task, the Binder resolves Public and External symbol references, combines segments, and performs address fix-ups on symbolic code and data.

The Binder takes object modules written in ASM-286, PL/M-286, Pascal-286 or FORTRAN-286, and generates a loadable module (for execution or debugging), or a linkable module (to be re-input to the Binder later; this is called incremental binding). The binder accepts library modules as well, linking only those modules required to resolve external references. BND-286 generates a print file displaying a segment map, and error messages.

The Binder will be used by system programmers and application programmers. Since application programmers need to develop software independent of any system architecture, the 286 memory protection mechanism is "hidden" from users of the Binder. This allows application tasks to be fully debugged before becoming part of a protected system. (A protected system may be debugged, as well.) System protection features are specified later in the development cycle, using the 286 System Builder. It is possible to link operating system services required by a task using either the Binder or the Builder. This flexibility adds to the ease of use of the 286 utilities.

Key Benefit:

The Binder is the only utility an application programmer needs to develop and debug an individual task. Users of the Binder need not be concerned with the architecture of the target machine, making application program development for the 286 very simple.

iAPX 286 MAPPER

- **Flexible Utility to Display Object File Information.**
- **MAP-286 Selectively Purges Symbols from a Load Module.**
- **Provides Inter-Module Cross-Referencing for Modules Written in All Languages.**
- **Mapper Allows Users to Display:**

Protection Information:	Debug Information:
SEGMENT TABLES	MODULE NAMES
GATE TABLES	PROGRAM SYMBOLS
PUBLIC ADDRESSES	LINE NUMBERS

Key Benefit:

A cross-reference map showing references *between* modules simplifies debugging; the map also lists and controls all symbolic information in one easy-to-read place.

AFN-00378B

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

iAPX 286 LIBRARIAN

- **Fast, Easy Management of iAPX 286 Object Module Libraries.**
- **Only Required Modules Are Linked, When Using the Binder or Builder.**
- **Librarian Allows Users to:**
  - **Create Libraries**
  - **Add Modules**
  - **Replace Modules**
  - **Delete Modules**
  - **Copy Modules from Another Library**
  - **Save Library Module to Object File**
  - **Create Backup**
  - **Display Module Information (creation date, publics, segments)**

Key Benefit:

Program libraries improve management of program modules, and reduce software administrative overhead.

iAPX 286 SYSTEM BUILDER

- **Supports Complete Creation of Protected, Multi-task Systems.**
- **Resolves PUBLIC/EXTERNAL Definitions (between protection levels).**
- **Supports Memory Protection by Building System Tables, Initializing Tasks, and Assigning Protection Rights to Segments.**
- **Creates a Memory Image of a 286 System for Cold-start Execution.**
- **Target System may be Boot-loadable, Programmed into ROM, or Loaded From Mass-store.**
- **Generates Print File with Command Listing and System Map.**

BLD-286 is the utility that lets system programmers configure multi-tasking, protected systems from an operating system and discrete tasks. The Builder generates a cold-start execution module, suitable for ROM-based or disk-based systems.

The Builder accepts input modules from iAPX 286 translators or the iAPX 286 Binder. It also accepts a "Build File" containing definitions and initial values for the 286 protection mechanism—descriptor tables, gates, segments, and tasks. BLD-286 generates a Loadable or bootloadable output module, as well as a print file with a detailed map of the memory-protected system.

Using the Builder command Language, system programmers may perform the following functions:

— Assign physical addresses to segments; also set segment access rights and limits.
— Create Call, Trap, and Interrupt "Gates" (entry-points) for inter-level program transfers.
— Make gates available to tasks; this is an easier way to define program interfaces than using interface libraries.
— Create Global (GDT), Interrupt (IDT), and any Local (LDT) Descriptor Tables.
— Create Task State Segments and Task Gates for multi-task applications.
— Resolve inter-module and inter-level references, and perform type-checking.
— Automatically select required modules from libraries.
— Configure the memory image into partitions in the address space.
— Selectively generate an object file and various sections of the print file.

Key Benefit:

Allows a system programmer to define the configuration of a protected system in *one* place, with one easy-to-use Utility. This specification may then be adopted by all project members, using either the Builder *or just the Binder*. The flexibility simplifies program development for all users.

AFN-00378B

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

iAPX 286 SIMULATOR

- **Supports Symbolic Debugging of Complete, Protected 286 Systems.**
- **Allows 286 Program Execution and Debugging in Absence of iAPX 286 Hardware Execution Vehicle.**
- **Functionally Duplicates the Operation of the iAPX 286 Microprocessor, Including Memory Protection.**
- **Executes Full Instruction Set, Including 80287 Numerics.**
- **Symbolic Access to Program Variables as well as Descriptor Tables.**
- **Two Execution Timers for Program Benchmarking and Interrupt Simulation.**
- **UDI File System Support for User Program.**

SIM-286 is an 8086-resident program designed to support development of iAPX 286 O.S. kernels, systems, and applications. All of these may be developed and debugged without the use of a 286 hardware execution vehicle.

The Simulator consists of a human interface layer, and software executors for the 80286 CPU and 80287 Numeric Data Processor. The human interface receives commands with symbolic names, and passes control to the executor as though it were a 286-resident monitor.

SIM-286 lets designers manipulate a 286 program using the symbolic names given for code and data. It also lets users symbolically examine and modify the protection features (such as system tables, access rights, etc.), if it is desired.

SIM-286 contains two instruction timers. One may be set and incremented during execution; this allows program sequences to be benchmarked in clock cycles and microseconds. The second, an interval timer, may be set to generate interrupts every η clock cycles, to simulate event-driven processing. These timers are extremely useful for developing system kernels.

For programs that make operating system calls for file I/O, SIM-286 provides access to these services through the Universal Development Interface.

Key Benefit:

Symbolic system debugging (for protected 286 software) may be performed in the absence of a 286-based target.

SPECIFICATIONS

OPERATING ENVIRONMENT

Intel Microcomputer Development Systems
(Series III/Series IV)

DOCUMENTATION

ASM 286 Language Reference Manual
ASM 286 Macro Assembler Operating Instructions
iAPX 286 Utilities User's Guide
iAPX 286 System Builder User's Guide
iAPX 286 Simulator User's Guide
Pocket Reference for all the above:
- ASM 286
- Utilities
- SIM 286

SUPPORT:

Hotline Telephone Support, Software Performance Report (SPR), Software Updates, Technical Reports, and Monthly Technical Newsletters are available.

ORDERING INFORMATION

Product Code	Description
iMDX-321	iAPX 286 Software Development Package

AFN-00378B

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PL/M 286 SOFTWARE PACKAGE

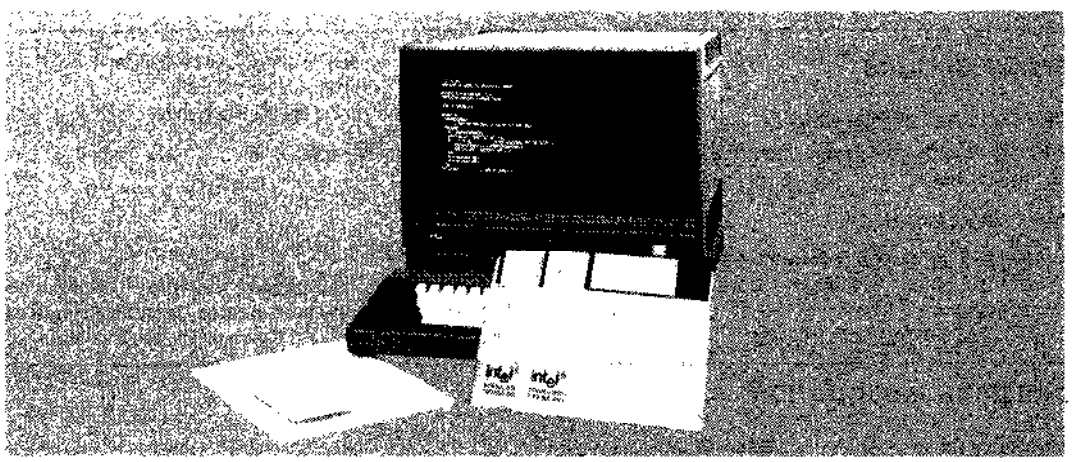
- **Systems programming language for the protected virtual address mode iAPX 286**
- **Upward compatible with PL/M 86 and PL/M 80 assuring software portability**
- **Enhanced to support design of protected, multi-user, multi-tasking, virtual memory operating system software**
- **Advanced, structured system implementation language for algorithm development**
- **Produces relocatable object code which is linkable to object modules generated by all other iAPX 286 language translators**
- **Multiple levels of optimization**
- **Resident on Intel microcomputer development systems (Series III, IV)**

PL/M 286 is a powerful, structured, high-level system implementation language for the development of system software for the protected virtual address mode iAPX 286. PL/M 286 has been enhanced to utilize iAPX 286 features—memory management and protection—for the implementation of multi-user, multi-tasking virtual memory operating systems.

PL/M 286 is upward compatible with PL/M 86 and PL/M 80. Existing systems software can be re-compiled with PL/M 286 to execute in protected virtual address mode on the iAPX 286.

PL/M 286 is the high-level alternative to assembly language programming on the iAPX 286. For the majority of iAPX 286 system programs, PL/M 286 provides the features needed to access and to control efficiently the underlying iAPX 286 hardware and consequently it is the cost-effective approach to develop reliable, maintainable system software.

The PL/M 286 compiler has been designed to efficiently support all phases of software development. Features such as a built-in syntax checker, multiple levels of optimization, virtual symbol table and four models of program size and memory usage for efficient code generation provide the total program development support needed.



MAY 1983
ORDER NUMBER:210536-002

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FEATURES

Major features of the Intel PL/M 286 compiler and programming language include:

Structured Programming

PL/M source code is developed in a series of modules, procedures, and blocks. Encouraging program modularity in this manner makes programs more readable, and easier to maintain and debug. The language becomes more flexible by clearly defining the scope of user variables (local to a private procedure, for example).

The use of modules and procedures to break down a large problem leads to productive software development. The PL/M 286 implementation of block structure allows the use of REENTRANT procedures, which are especially useful in system design.

Language Compatibility

PL/M 286 object modules are compatible with object modules generated by all other 286 translators. This means that PL/M programs may be linked to programs written in any other 286 language.

Object modules are compatible with In-Circuit Emulators; DEBUG compiler control provides the In-Circuit Emulators with full symbolic debugging capabilities.

PL/M 286 language is upward compatible with PL/M 86 and PL/M 80 so that application programs may be easily ported to run on the protected mode iAPX 286.

Supports Seven Data Types

PL/M makes use of seven data types for various applications. These data types range from one to four bytes and facilitate various arithmetic, logic, and addressing functions:

Byte: 8-bit unsigned number
—Word: 16-bit unsigned number
—Dword: 32-bit unsigned number
—Integer: 16-bit signed number
—Real: 32-bit floating-point number
—Pointer: 16-bit or 32-bit memory address indicator
—Selector: 16-bit pointer base.

Another powerful facility allows the use of BASED variables which permit run-time mapping of variables to memory locations. This is especially useful for passing parameters, relative and absolute addressing, and dynamic memory allocation.

Two Data Structuring Facilities

In addition to the seven data types and based variables, PL/M supports two powerful data structuring facilities. These help the user to organize data into logical groups.

—Array: Indexed list of same type data elements
—Structure: Named collection of same or different type data elements
—Combinations of both: Arrays of structures or structures of arrays.

Numerics Support

PL/M programs that use 32-bit REAL data are executed using the 80287 Numeric Data Processor for high performance. All floating-point operations supported by PL/M are executed on the 80287 according to the IEEE floating-point standard. PL/M 286 programs can use built-in functions and predefined procedures—INIT$REAL$MATH$UNIT, SET$REAL$MODE, GET$REAL$ERROR, SAVE$REAL$STATUS, RESTORE$REAL$STATUS—to control the operation of the 80287 within the scope of the language.

Built-In String Handling Facilities

The PL/M 286 language contains built-in functions for string manipulation. These byte and word functions perform the following operations on character strings: MOVE, COMPARE, TRANSLATE, SEARCH, SKIP, and SET.

Built-In Port I/O

PL/M 286 directly supports input and output from the iAPX 286 ports for single BYTE and WORD transfers. For BLOCK transfers, PL/M 286 programs can make calls to predefined procedures.

Interrupt Handling

PL/M 286 has the facility for generating and handling interrupts on the iAPX 286. A procedure may be defined as an interrupt handler through use of the INTERRUPT attribute. The compiler will then generate code to save and restore the processor status on each execution of the user-defined

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

interrupt handler routine. The PL/M statement CAUSE$INTERRUPT allows the user to trigger a software interrupt from within the program.

Protection Model

PL/M 286 supports the implementation of protected operating system software by providing built-in procedures and variables to access the protection mechanism of the iAPX 286. Predefined variables—TASK$REGISTER, LOCAL$TABLE, MACHINE$STATUS, etc.—allow direct access and modification of the protection system. Untyped procedures and functions—SAVE$GLOBAL$TABLE, RESTORE$GLOBAL$TABLE, SAVE$INTERRUPT$TABLE, RESTORE$INTERRUPT$TABLE, CLEAR$TASK$SWITCHED$FLAG, GET$ACCESS$RIGHTS, GET$SEGMENT$LIMIT, SEGMENT$READABLE, SEGMENT$WRITABLE, ADJUST$RPL—provide all the facilities needed to implement efficient operating system software.

Compiler Controls

The PL/M 286 compiler offers controls that facilitate such features as:

—Optimization
—Conditional compilation
—The inclusion of additional PL/M source files from disk
—Cross-reference of symbols
Optional assembly language code in the listing file
—The setting of overflow conditions for run-time handling.

Addressing Control

The PL/M 286 compiler uses the SMALL, COMPACT, MEDIUM, and LARGE controls to generate optimum addressing instructions for programs. Programs of any size can be easily modularized into "subsystems" to exploit the most efficient memory addressing schemes. This lowers total memory requirements and improves run-time execution of programs.

Code Optimization

The PL/M 286 compiler offers four levels of optimization for significantly reducing overall program size.

—Combination or "folding" of constant expressions; and short-circuit evaluation of Boolean expressions
—"Strength reductions": a shift left rather than multiply by 2; and elimination of common subexpressions within the same block
—Machine code optimizations; elimination of superfluous branches; reuse of duplicate code; removal of unreachable code
—Optimization of based-variable operations and cross-statement load/store.

Error Checking

The PL/M 286 compiler has a very powerful feature to speed up compilations. If a syntax or program error is detected, the compiler will skip the code generation and optimization passes. This usually yields a 2X performance increase for compilation of programs with errors.

A fully detailed and helpful set of programming and compilation error messages is provided by the compiler and user's guide.

BENEFITS

PL/M 286 is designed to be an efficient, cost-effective solution to the special requirements of protected mode iAPX 286 Microsystem Software Development, as illustrated by the following benefits of PL/M use:

Low Learning Effort

PL/M 286 is easy to learn and use, even for the novice programmer.

Earlier Project Completion

Critical projects are completed much earlier than otherwise possible because PL/M 286, a structured high-level language, increases programmer productivity.

Lower Development Cost

Increases in programmer productivity translate immediately into lower software development costs because less programming resources are required for a given programmed function

Increased Reliability

PL/M 286 is designed to aid in the development of reliable software (PL/M 286 programs are simple statements of the program algorithm). This substantially reduces the risk of costly correction of errors in

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

systems that have already reached full production status, as the more simply stated the program is, the more likely it is to perform its intended function.

Easier Enhancements and Maintenance

Programs written in PL/M tend to be self-documenting, thus easier to read and understand. This means it is easier to enhance and maintain PL/M programs as the system capabilities expand and future products are developed.

Cost-Effective Alternative to Assembly Language

PL/M 286 programs are code efficient. PL/M 286 combines all of the benefits of a high-level language (ease of use, high productivity) with the ability to access the iAPX 286 architecture. This includes language features for control of the iAPX 286 protection mechanism. Consequently, for the development of systems software, PL/M 286 is the cost-effective alternative to assembly language programming.

SPECIFICATIONS

Operating Environment

Intel Microcomputer Development System (Series III/Series IV)

Documentation Package

PL/M 286 User's Guide

ORDERING INFORMATION

Part Number	Description
iMDX 323	PL/M 286 Software Package

Requires Software License

SUPPORT:

Hotline Telephone Support, Software Performance Report (SPR), Software Updates, Technical Reports, and Monthly Technical Newsletters are available.

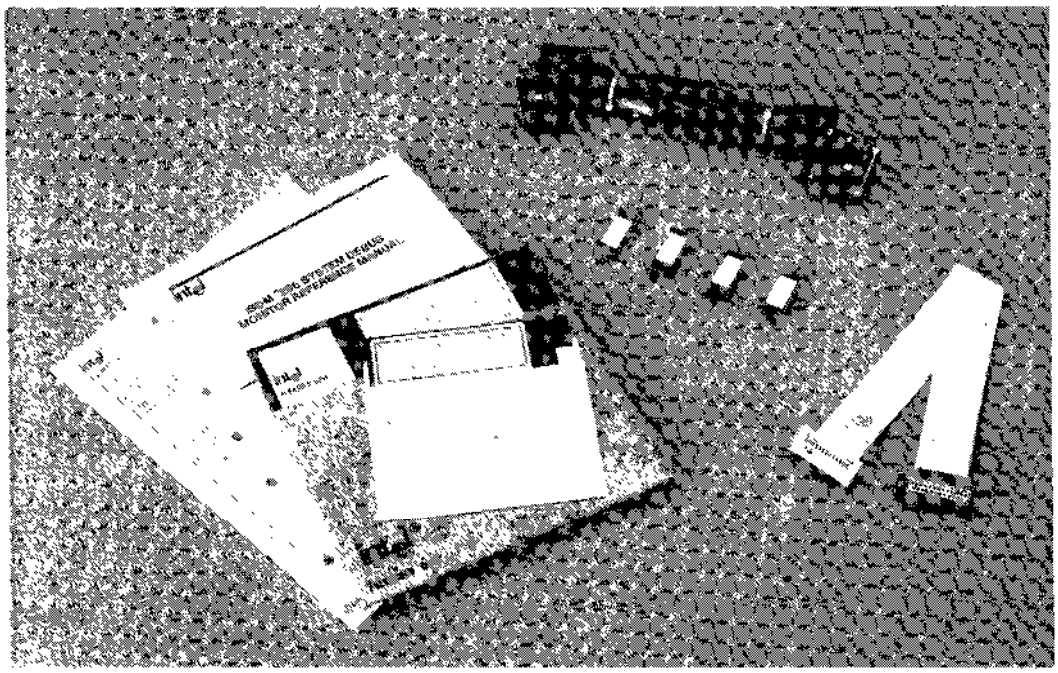
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PRELIMINARY

iSDM™ 286
iAPX 286 SYSTEM DEBUG MONITOR

- **Development support for iSBC® 286- and iAPX 286-based applications**
- **Real Address Mode (RAM) and Protected Virtual Address Mode (PVAM) support**
- **Support of MULTIBUS® I and MULTIBUS® II environments**
- **Powerful debugging commands, including single step CPU operation**
- **For MULTIBUS® II, software configuration of system boards at start-up and automatic configuration of memory boards**
- **Universal Development Interface (UDI) support via development system connection**
- **Command execution, including program load capability from Intellec® Series III or Series IV Development Systems**
- **Supports 80287 Numeric Processor Extension (NPX) for high-speed math applications**

The Intel iSDM™ 286 System Debug Monitor package contains the necessary software, cables, EPROMs, and documentation required to interface an iSBC® 286 board or iAPX 286 application to an Intellec® Series III or Series IV through a high-speed link. The System Debug Monitor supports an OEM's choice of MULTIBUS® I or MULTIBUS II environments, and the iRMX™ 86 Real-Time Multitasking Operating System or a custom operating system. The monitor contains debugging tools that examine CPU registers, memory content, CPU descriptor tables, and other crucial environmental details. The Monitor also allows programs to access files on the development system via the internal UDI support and the serial communication link.



The following are trademarks of Intel Corporation and may be used only to describe Intel products: Intel, ICE, iMMX, iRMX, iSBC, iSBX, iSXM, MULTIBUS, MULTICHANNEL and MULTIMODULE. Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied. Information contained herein supercedes previously published specifications on these devices from Intel.

© INTEL CORPORATION, 1984

SEPTEMBER, 1984
ORDER NUMBER: 230882-002

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FUNCTIONAL DESCRIPTION

Overview

The iSDM 286 System Debug Monitor provides programmers of iAPX 286-based applications with the debugging tools needed to test new applications ranging from single-user systems to complex operating systems executing in either a MULTIBUS I or MULTIBUS II environment. Programmers are given direct access to both the Real Address (RAM) and Protected Virtual Address (PVAM) modes of the CPU via a simple terminal interface or via an Intellec Series III or Series IV Development System.

Powerful Debugging Commands

The iSDM 286 Monitor contains a powerful set of user functions, including commands to:

- Examine and modify CPU registers
- Examine, modify, and move memory locations
- Symbolic reference to variable names
- Find and compare memory contents
- Set program breakpoints
- Bootstrap load application software from iRMX 86 file compatible peripherals (requires the iRMX 86 Operating System for Bootstrap Loader)
- Single-step CPU operation
- Switch from Real Address Mode to Protected Virtual Address Mode

Formatted Displays

The iSDM 286 Monitor formats all iAPX 286 predefined data structures into clearly understandable displays. This display gives programmers a formatted view of such CPU structures as LDTs, GDTs, IDTs, Segment Selectors, and Task State Segments—not just a series of unconnected digits.

Universal Development Interface (UDI)

Via the Universal Development Interface (UDI), the iSDM 286 Monitor can support the execution of iRMX 86, Series III, Series IV, or any other UDI-based applications. The Monitor emulates many of the UDI calls (RAM or PVAM), and passes all requests for a file system to the host development system. UDI applications, such as compilers and other programs available from Independent Software Vendors, can be tested in the target iAPX 286 environment immediately.

MULTIBUS® II Software Configuration of System Boards

The MULTIBUS II Interconnect Space Registers allow the software to configure boards, eliminating much of the need for jumpers and wire wraps. The iSDM 286 Monitor can initialize these registers at configuration time using user-defined variables. The Monitor can also automatically configure memory boards, defining the addresses for each board sequentially in relation to the board's physical placement in the card cage. This feature allows for the swapping, adding, and deleting of memory boards on a dynamic basis.

Command Execution

Commands to the iSDM 286 Monitor are entered interactively via a standalone terminal, an Intellec® Series III or a Series IV Development System. The target application hardware is connected to the terminal or development system via a serial link. Figure 1 shows a typical MULTIBUS I environment and Figure 2 shows a typical MULTIBUS II environment. All control operations and UDI file manipulations occur over the serial link through the cables supplied. More than one channel can be configured for the communication since the Monitor scans all configured channels to determine which channel is in use.

Numeric Data Processor Support

In addition to executing 80287 Numeric Processor Extension (NPX) applications with full NPX performance, programmers may examine and modify NPX registers using decimal and real number format. Any location in memory known to contain numeric values in standard real format (IEEE-P754) may be examined or modified using normal decimal notation. In this manner, programmers may feel confident that correct and meaningful numbers are available to applications without having to encode and decode complex real, integer, and BCD hexadecimal formats.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

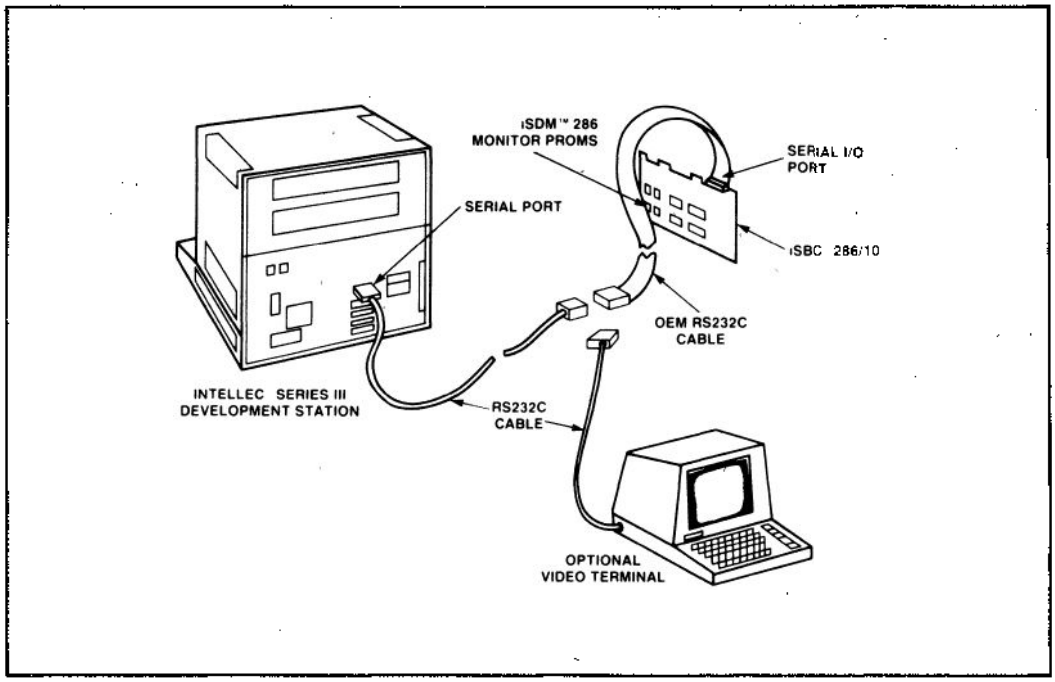


Figure 1. Typical MULTIBUS® I Environment

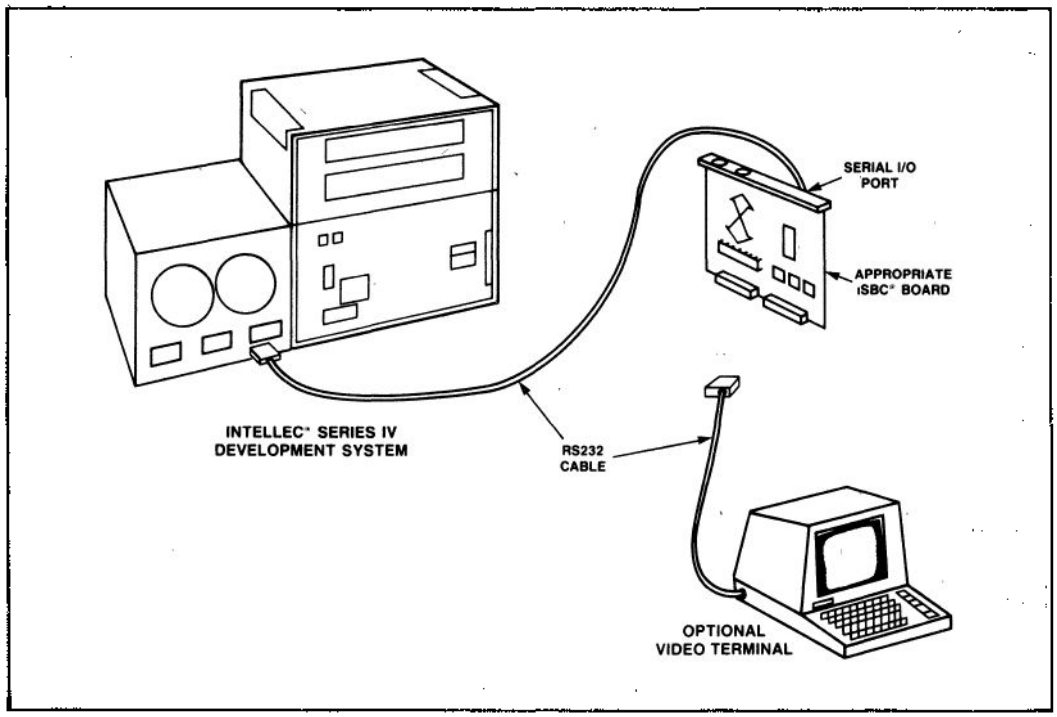


Figure 2. Typical MULTIBUS® II Environment

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SPECIFICATIONS

Development System Environment

Intellec Series III or Series IV Development System with 128K of memory and 1 disk drive.

Target System Environment

Any iAPX 286 system with at least 4K of read-write memory starting at location 0H and 32K of read-only memory starting at location 0FF8000H.

Serial communication with a stand-alone terminal or development system requires either a 8274 USART and 8253 or 8254 PIT, or an 82530 SCC.

Monitor EPROMs are supplied for locations 0FF8000H through 0FFFFFFH.

ORDERING INFORMATION

The iSDM 286 System Debug Monitor package includes cables, EPROMs, software, and a reference manual. The software is provided on a double-density, single-sided ISIS-formatted 8" diskette for Series III Development System use and on a double-density, double-sided iRMX-formatted 5¼" diskette for Series IV Development System use.

The OEM license option listed here allows users to incorporate iSDM 286 into their applications. Each use requires payment of an Incorporation Fee.

ORDER CODE: iSDM 286 RO.

The iSDM 286 RO product also includes 90 days of support services that includes the Software Problem Report service.

Another licensing option includes prepayment of all future incorporation fees.

As with all Intel software, purchase of any of these options requires the execution of a standard Intel Master Software license. The specific rights granted to users depends on the specific option and the license signed.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

80287 SUPPORT LIBRARY

- **Library to support floating point arithmetic in Pascal-286, PL/M-286 and ASM-286**
- **Common elementary function library provides trigonometric, logarithmic and other useful functions**
- **Decimal conversion module supports binary-decimal conversions**
- **Error-handler module simplifies floating point error recovery**
- **Supports proposed IEEE Floating Point Standard for high accuracy and software portability**

The 80287 Support Library provides Pascal-286, PL/M-286 and ASM-286 users with numeric data processing capability. With the Library, it is easy for programs to do floating point arithmetic. Programs can bind in library modules to do trigonometric, logarithmic and other numeric functions, and the user is guaranteed accurate, reliable results for all appropriate inputs. Figure 1 below illustrates how the 80287 Support Library can be bound with PL/M-286 and ASM-286 user code to do this. The 80287 Support Library supports the proposed IEEE Floating Point Standard. Consequently, by using this Library, the user not only saves software development time, but is guaranteed that the numeric software meets industry standards and is portable--the software investment is maintained.

The 80287 Support Library consists of the common elementary function library (CEL287.LIB), the decimal conversion library (DC287.LIB), the error handler module (EH287.LIB) and interface libraries (80287.LIB, NUL287.LIB).

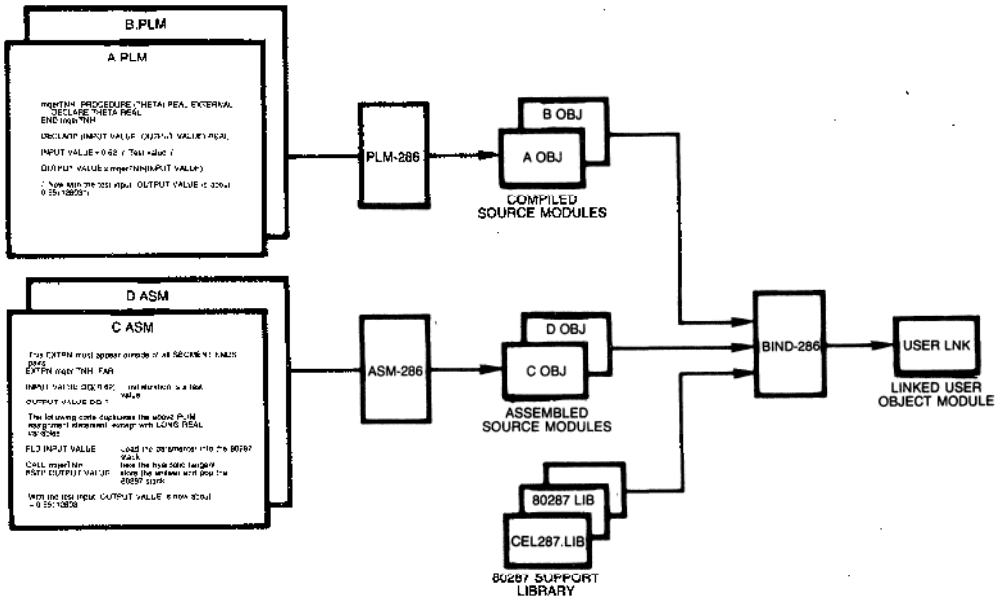


Figure 1. Use of 80287 Support Library with PL/M-286 and ASM-286.

Intel Corporation Assumes No Responsibility for the Use of Any Circuitry Other Than Circuitry Embodied in an Intel Product. No Other Circuit Patent Licenses are implied
Information Contained Herein Supercedes Previously Published Specifications of These Devices from Intel
©INTEL CORPORATION, 1984

MARCH 1984
ORDER NUMBER: 231041-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

CEL287.LIB
THE COMMON ELEMENTARY FUNCTION LIBRARY

FUNCTIONS

CEL287.LIB contains commonly used floating point functions. It is used along with the 80287 numeric coprocessor. It provides a complete package of elementary functions, giving valid results for all appropriate inputs. Following is a summary of CEL287 functions, grouped by functionality.

Rounding and Truncation Functions:

mqerIEX, mqerIE2, and mqerIE4. Round a real number to the nearest integer; to the even integer if there is a tie. The answer returned is real, a 16-bit integer or a 32-bit integer respectively.

mqerIAX, mqerIA2, mqerIA4. Round a real number to the nearest integer, to the integer away from zero if there is a tie; the answer returned is real, a 16-bit integer or a 32-bit integer, respectively.

mqerICX, mqerIC2, mqerIC4. Truncate the fractional part of a real input; the answer is real, a 16-bit integer or 32-bit integer, respectively.

Logarithmic and Exponential Functions:

mqerLGD computes decimal (base 10) logarithms.
mqerLGE computes natural (base e) logarithms.
mqerEXP computes exponentials to the base e.
mqerY2X computes exponentials to any base.
mqerY12 raises an input real to a 16-bit integer power.
mqerY14 is as mqerY12, except to a 32-bit integer power.
mqerYIS is as mqerY12, but it accommodates PL/M-286 users.

Trigonometric and Hyperbolic Functions:

mqerSIN, mqerCOS, mqerTAN compute sine, cosine, and tangent.

mqerASN, mqerACS, mqerATN compute the corresponding inverse functions.

mqerSNH, mqerCSH, mqerTNH compute the corresponding hyperbolic functions.

mqerAT2 is a special version of the arc tangent function that accepts rectangular coordinate inputs.

Other Functions:

mqerDIM is FORTRAN's positive difference function.
mqerMAX returns the maximum of two real inputs.
mqerMIN returns the minimum of two real inputs.
mqerSGH combines the sign of one input with the magnitude of the other input.
mqerMOD computes a modulus, retaining the sign of the dividend.
mqerRMD computes a modulus, giving the value closest to zero.

DC287.LIB
THE DECIMAL CONVERSION LIBRARY

DC287.LIB is a library of procedures which convert binary representations of floating point numbers and ASCII-encoded string of digits.

The binary-to-decimal procedure mqcBIN_DECLOW accepts a binary number in any of the formats used for the representation of floating point numbers in the 80287. Because there are so many output formats for floating point numbers, mqcBIN_DECLOW does not attempt to provide a finished, formatted text string. Instead, it provides the "building blocks" for you to use to construct the output string which meets your exact format specification.

The decimal-to-binary procedure mqcDEC_BIN accepts a text string which consists of a decimal number with optional sign, decimal point, and/or power-of-ten exponent. It translates the string into the caller's choice of binary formats.

Decimal-to-binary procedure mqcDECLOW_BIN is provided for callers who have already broken the decimal number into its constituent parts.

The procedures mqcLONG_TEMP, mqcSHORT_TEMP, mqcTEMP_LONG, and mqcTEMP_SHORT convert floating point numbers between the longest binary format, TEMP_REAL, and the shorter formats.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

EH287.LIB
THE ERROR HANDLER MODULE

EH287.LIB is a library of five utility procedures for writing trap handlers. Trap handlers are called when an unmasked 80287 error occurs.

The 80287 error reporting mechanism can be used not only to report error conditions, but also to let software implement IEEE standard options not directly supported by the chip. The three such extensions to the 80287 are: normalizing mode, non-trapping not-a-number (NaN), and non-ordered comparison. The utility procedures support these extra features.

DECODE is called near the beginning of the trap handler. It preserves the complete state of the 80287, and also identifies what function called the trap handler, and returns available arguments and/or results. DECODE eliminates much of the effort needed to determine what error caused the trap handler to be called.

NORMAL provides the "normalizing mode" capability for handling the "D" exception. By calling NORMAL in your trap handler, you eliminate the need to write code in your application program which tests for non-normal inputs.

SIEVE provides two capabilities for handling the "I" exception. It implements non-trapping NaN's and non-ordered comparisons. These two IEEE standard features are useful for diagnostic work.

ENCODE is called near the end of the trap handler. It restores the state of the 80287 saved by DECODE, and performs a choice of concluding actions, by either retrying the offending function or returning a specified result.

FILTER calls each of the above four procedures. If your error handler does nothing more than detect fatal errors and implement the features supported by SIEVE and NORMAL, then your interface to EH287.LIB can be accomplished with a single call to FILTER.

80287.LIB, NUL287.LIB
INTERFACE LIBRARIES

80287.LIB and NUL287.LIB libraries configure a user's application program for his run-time environment; running with the 80287 component or without floating point arithmetic, respectively.

SPECIFICATIONS

Operating Environment

Intel Microcomputer Development Systems (Series III, Series IV)

Documentation Package

80287 Support Library Reference Manual

Related Software

A 80287 software emulator is available as part of the 8086 software toolbox (iMDX364)

ORDERING INFORMATION

Part Number	Description
iMDX329	80287 Support Library

Requires Software License

SUPPORT

Intel offers several levels of support for this product which are explained in detail in the price list. Please consult the price list for a description of the support options available.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PASCAL-286 SOFTWARE PACKAGE

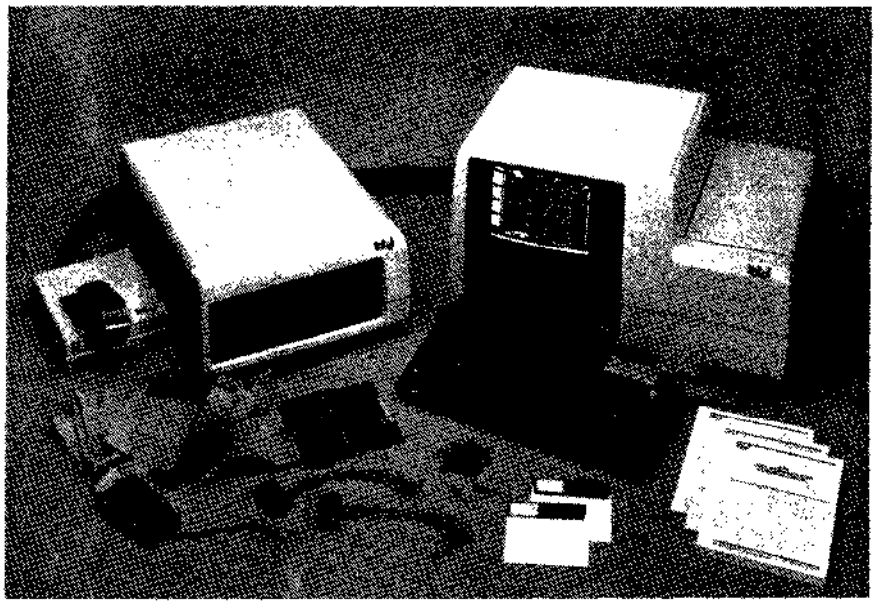
- **High-level programming language for the protected virtual mode iAPX 286**
- **Implements ISO standard Pascal. Many useful extensions may be enabled via a compiler switch**
- **Upward compatible with Pascal-86 for software portability**
- **Produces relocatable object code which is linkable to object modules generated by other iAPX 286 translators**
- **Supports full symbolic debugging with iAPX 286 software and ICE™ debuggers**
- **Fully supports the 80287 numeric processor using the IEEE floating point standard**

Pascal-286 is a powerful, structured, applications programming language for the protected virtual address mode of the iAPX 286. Pascal-286 is upward compatible with Pascal-86 so that 8086 Pascal source code can be ported to the iAPX 286 in protected mode.

Pascal-286 implements strict ISO standard Pascal, but with many useful extensions. These include separate compilation of modules, interrupt handling, port I/O, and 80287 numerics support. A control is provided in the compiler to flag all non-ISO features used.

Pascal-286 produces relocatable object code which can be linked with object code produced by other iAPX 286 translators such as ASM-286 and PL/M-286. Thus, a combination of translators can be used to provide great programming flexibility.

Type and symbol information needed by software and in-circuit debuggers is added to the object code by the Pascal-286 compiler. This information can be stripped off by the compiler or linker for the final production version.



Intel Corporation Assumes No Responsibility for the use of Any Circuitry Other Than Circuitry Embodied in an Intel Product. No Other Patent Licenses are Implied. ©INTEL CORPORATION, 1982. Note: The development system pictured here is not included in the Pascal-286 software package, but merely depicts the language in its operating environment

©Intel Corporation, 1983

NOVEMBER 1983
ORDER NUMBER: 230863-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FEATURES

Conforms to ISO Standard Pascal

Pascal has gained wide acceptance as a portable language for microcomputer applications. However, portability can result only if standards are adhered to. Pascal-286 is a strict implementation of ISO standard Pascal. Extensions are provided to make the language more powerful for microprocessor applications. All extensions are clearly highlighted in the documentation. In addition, the compiler provides a control to flag any non ISO feature used. Pascal-286 will evolve to track future enhancements to standard Pascal.

Upward Compatible with Pascal-86

The Pascal-286 compiler produces object code for the protected virtual address mode of the iAPX 286 language. However, no 286 architecture specific features have been added to the Pascal-286 language. This makes Pascal-286 source code upward compatible with Pascal-86, which allows for porting of 8086 software to the protected 286 with relative ease.

Compatible With Other iAPX 286 Translators

All Intel iAPX 286 translators output object code in a standardized format. This allows 286 programs to be written in a mixture of languages. Systems routines which need access to architectural features can be coded in PL/M-286 or ASM-286. Pascal-286 may be better suited for the applications routines. The systems and application routines can then be combined using the 286 linker (BIND-286).

Standardized Run Time Support

Programs compiled with Pascal-286 can be moved from the development host environment to the target environment with ease. This is the result of standardizing run-time operating system interfaces required by the compiled program into a well defined and well documented set of routines. After programs are developed on a development host, they can then be executed in the target using the same set of system interfaces.

Extensions for Microprocessor Programming

Pascal-286 provides extensions that make it powerful for microprocessor applications. Built-in procedures allow I/O directly from the ports of the iAPX 286. This speeds up I/O as it is done by direct communication with the microprocessor. Interrupt processing is also supported by built in procedures. Examples are: ENABLEINTERRUPTS, DISABLEINTERRUPTS, CAUSEINTERRUPT. Many built in procedures and variables are provided for communicating with the 80287 for numeric computations.

Compiler Controls

The Pascal-286 compiler provides many controls which can be used at invocation time to enhance programming flexibility. Examples are: CODE/NOCODE, DEBUG/NODEBUG, INCLUDE (file), LIST/NOLIST, OPTIMIZE (n), EXTENSIONS/NOEXTENSIONS. All controls have default values that are active unless the opposite is specified during invocation. Thus, for most compiles, no controls need be specified.

Support for IEEE Standard Numerics

Pascal-286 provides full support for the 80287 numerics co-processor. All floating point operations are done according to the IEEE floating point standard. The benefits are predictable, accurate and consistent results. Built-in procedures to support the 80287 include GET8087ERRORS and MASK 8087ERRORS. A full set of 80287 library routines are supplied with the compiler.

Optimizations

The Pascal-286 compiler produces highly optimized code, both in size and execution time. This is achieved by:

—Use of powerful iAPX 286 instructions, in particular, for string handling, 80287 numerics and subroutine linkage

—Short circuit evaluation of boolean expressions, constant folding and strength reduction of multiplications and additions

—Elimination of superfluous branches, optimization of span dependent jumps

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SPECIFICATIONS

Operating Environment

Intel 8086 based microcomputer
Development systems (Series III, Series IV)

Documentation Package

Pascal-286 User's Guide
Pascal-286 Pocket Reference

ORDERING INFORMATION

Part Number	Description
iMDX-324	Pascal-286 Software Package

Requires Software License

Support

Hotline service, SPR (Software Performance Reports),
Updates and technical newsletters are available.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*/VMS* RESIDENT SOFTWARE DEVELOPMENT PACKAGES FOR iAPX 286

- **Hosted on DEC VAX* Minicomputer Under the VMS* Operating System**
- **Allows Development of System and Application Software for the Protected Virtual Address Mode of the iAPX-286**
- **Packages include PL/M-286, BUILD-286, BIND-286, LIB-286 and MAP-286**
- **Compatible with Corresponding Intel Development System Resident Products**

These packages provide the capability of developing software on a VAX*/VMS* host for the iAPX-286 in protected virtual address mode. With these packages a user can assemble and compile 286 programs, configure system and application software and create and manage 286 object libraries. Figure 1 illustrates the process of 286 software development on VAX*/VMS* hosts.

Two packages are available:

1. A PL/M-286 package which contains the PL/M-286 compiler and run time support libraries.
2. An ASM-286 package which contains the iAPX-286 Assembler (ASM-286) and programming utilities. These utilities include the iAPX-286 System Builder (BLD-286), the System Binder (BND-286), a Library Utility (LIB-286) and an Object Map Utility (MAP-286).

These packages are compatible with corresponding products which are hosted on Intel development systems. Correspondence can be established via version numbers. For example, BND-286 V2.0 offers the same set of features on VAX/VMS and Intel development systems.

Owing to this compatibility, iAPX-286 software developed on VAX/VMS can be linked to iAPX-286 software from development systems. Moreover, iAPX-286 programs developed on the VAX can then be downloaded to development systems and debugged using 286 debuggers like the I^2ICE^{TM}-286 system.

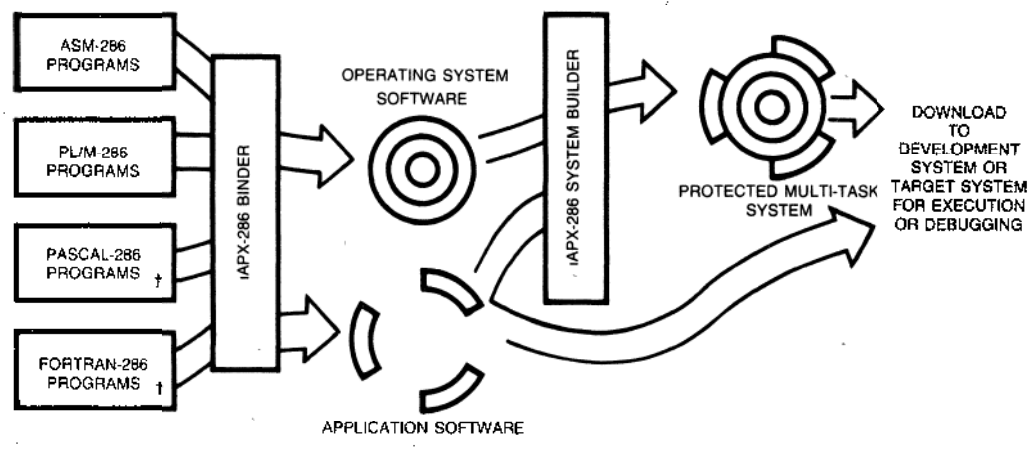


Figure 1: 286 Software Development on VAX*/VMS*

*VAX, VMS are trademarks of Digital Equipment Corporation †Currently Available on Intel Development Systems Only

Intel Corporation Assumes No Responsibility for the Use of Any Circuitry Other Than Circuitry Embodied in an Intel Product No Other Circuit Patent Licenses are Implied Information Contained Herein Supercedes Previously Published Specifications on These Devices from Intel
©INTEL CORPORATION, 1984

MARCH 1984
ORDER NUMBER: 231038-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*/VMS* RESIDENT PL/M-286

- **Systems Programming Language for the protected virtual address mode iAPX-286**
- **Produces relocatable object code linkable to object modules generated by other Intel 286 language translators**
- **Enhanced to support design of protected, multi-user, multi-tasking, virtual memory operating system software**
- **Upward compatible with PL/M-86 and PL/M-80 to allow software portability**
- **Provides multiple levels of optimization to produce efficient code**
- **Compatible with development system resident PL/M-286**

PL/M-286 is a powerful, structured, high-level system implementation language for the development of system software for the protected virtual address mode iAPX-286. PL/M-286 has been enchanced to utilize iAPX-286 features--memory management and protection--for the implementation of multi-user, multi-tasking virtual memory operating systems.

PL/M-286 is upward compatible with PL/M-86 and PL/M-80. Existing systems software can be re-compiled with PL/M-286 to execute in protected virtual address mode on the iAPX-286.

PL/M-286 is the high-level alternative to assembly language programming on the iAPX-286. For the majority of iAPX-286 system programs, PL/M-286 provides the features needed to access and to control efficiently the underlying iAPX-286 hardware, and consequently it is the cost-effective approach to develop reliable, maintainable system software.

The PL/M-286 compiler has been designed to efficiently support all phases of software development. Features such as built-in syntax checker, multiple levels of optimization, virtual symbol table and four models of program size and memory usage for efficient code generation provide the total program development support needed. The compiler also provides complete symbolic debug capability to the various 286 debuggers and emulators

VAX/VMS resident PL/M-286 is completely feature compatible with development system resident PL/M-286 with the same version number.

231038-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*/VMS* RESIDENT iAPX-286 MACRO ASSEMBLER

- **Supports full Instruction Set of the iAPX-286 including memory protection and numerics (with 80287)**
- **Structures and RECORDS provide powerful data representation**
- **Type checking at assembly time helps reduce errors at run-time**
- **Powerful and flexible Text Macro facility**
- **Upward compatible with ASM-86/88/186**
- **Compatible with development system resident iAPX-286 Macro Assembler**

ASM-286 is the "high-level" macro assembler for the iAPX-286 assembly language. ASM-286 translates symbolic assembly language mnemonics into relocatable object code. The assembler mnemonics are a superset of ASM-86/88 mnemonics; new ones have also been added to support the new iAPX-286 instructions. The segmentation directives have been greatly simplified.

The iAPX-286 assembly language includes approximately 150 instruction mnemonics. From these few mnemonics the assembler can generate over 4,000 distinct machine instructions. Therefore, the software development task is simplified, as the programmer need know only 150 mnemonics to generate all possible machine instructions. ASM-286 generates the shortest machine instruction possible (given explicit information as to the characteristics of any forward referenced symbols).

The powerful macro facility in ASM-286 saves development and maintenance time by coding common program sequences only once. A macro substitution is made each time the sequence is to be used. This facility also allows for conditional assembly of certain program sequences.

ASM-286 offers many features normally found only in high-level languages. The assembly language is strongly typed, which means it performs extensive checks on the usage of variables and labels. This means that many programming errors will be detected when the program is assembled, long before it is being debugged.

ASM-286 object modules conform to a thorough, well-defined format used by 286 high-level languages and utilities. This makes it easy to call (and be called from) HLL object modules.

ASM-286 also provides support for the 80287 numerics co-processor. The complete instruction set of the 80287 is available through high-level mnemonics.

VAX/VMS resident ASM-286 is completely feature compatible with development system resident ASM-286 with the same version number.

231038-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*/VMS* RESIDENT iAPX-286 SYSTEM BUILDER

- **A tool for configuring multi-tasking protected, virtual memory systems software for the iAPX-286**
- **Links separately compiled modules. Resolves EXTERNAL/PUBLIC definitions**
- **Creates a memory image of a 286 system for cold start execution**
- **Target system may be bootloadable, programmed into ROM or loaded from mass storage**
- **Generates print file with command listing and system map**
- **Compatible with development system resident iAPX-286 System Builder**

BLD-286 is the iAPX-286 System Builder. It allows systems programmers to configure multi-tasking and memory protected iAPX-286 software. The configuration is specified by the user in a "Build file" using a symbolic meta-language. BLD-286 thus provides the programmer a high-level symbolic interface to the multi-tasking and memory protection features of the iAPX-286 architecture.

BLD-286 accepts as inputs object modules from the iAPX-286 translators, the iAPX-286 Binder and itself (for incremental building). Using the programmer's specifications in the Build File, it produces a bootloadable or loadable module as well as a print file with a map of the configured module.

Using the builders command language, system programmers may perform the following functions:

— Assign physical addresses to segments; also set segment access rights and limits.
— Create Call, Trap, and Interrupt "Gates" (entry-points) for inter-level program transfers.
— Make gates available to tasks; this is an easier way to define program interfaces than using interface libraries.
— Create Global (GDT), Interrupt (IDT), and any Local (LDT) Descriptor Tables.
— Create Task State Segments and Task Gates for multi-tasking applications.
— Resolve inter-module and inter-level references, and perform type-checking.
— Automatically select required modules from libraries.
— Configure the memory image into partitions in the address space.
— Selectively generate an object file and various sections of the print file.

VAX/VMS BLD-286 is completely feature compatible with development system resident BLD-286 with the same version number.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*/VMS* RESIDENT iAPX-286 BINDER

- **Links separately compiled program modules into an executable task**
- **Resolves PUBLIC/EXTERNAL code and data references, and performs intermodule type-checking**
- **Makes the iAPX-286 protection mechanism invisible to application programmers**
- **Provides print file showing segment map, errors and warnings**
- **Assigns virtual addresses to tasks**
- **Generates linkable or loadable module for debugging**
- **Performs incremental linking with output of Binder and Builder**
- **Compatible with development system resident iAPX-286 Binder**

BND-286 is a utility that combines iAPX-286 object modules into executable tasks. In creating a task, the Binder resolves Public and External symbol references, combines segments, and performs address fix-ups on symbolic code and data.

The Binder takes object modules, produced by the 286 translators, and generates a loadable module (for execution or debugging), or a linkable module (to be re-input to the Binder later; this is called incremental binding). The binder accepts library modules as well, linking only those modules required to resolve external references. BND-286 generates a print file displaying a segment map, and error messages.

The Binder is useful for system as well as application programmers. Since application programmers need to develop software independent of any system architecture, the 286 memory protection mechanism is "hidden" from users of the Binder. This allows application tasks to be fully debugged before becoming part of a protected system. (A protected system may be debugged, as well.) System protection features are specified later in the development cycle, using the 286 System Builder. It is possible to link operating system services required by a task using either the Binder or the Builder. This flexibility adds to the ease of use of the 286 utilities.

VAX/VMS resident BND-286 is completely feature compatible with development system resident BND-286 with the same version number.

VAX*/VMS* RESIDENT iAPX-286 LIBRARIAN

- **Allows creation and management of iAPX-286 object libraries**
- **Only required modules linked in when using Binder or Builder**
- **Library functions include Create, Delete, Add, Replace, Copy, Save, Backup and Display**
- **Compatible with development system resident iAPX-286 Librarian**

LIB-286 is the iAPX-286 Librarian. It can be used to create and manage iAPX-286 Object Libraries. By placing often used object modules into libraries, the administrative overhead of managing software modules can be reduced.

VAX/VMS based LIB-286 is completely feature compatible with development system resident LIB-286 with the same version number.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*/VMS* RESIDENT iAPX-286 MAPPER

- **Flexible Utility to display object file information in symbolic form**
- **Compatible with development system resident iAPX-286 Mapper**

MAP-286 is a cross reference utility for iAPX-286 object modules. It provides a symbolic listing of the EXTERNAL and PUBLIC symbols in the specified object modules.

VAX/VMS resident MAP-286 is completely feature compatible with development system resident MAP-286 with the same version number.

SPECIFICATIONS

Operating Environment

DEC VAX* 11/780 or compatible model running VMS* operating system V3.4 (or upward compatible versions)

Documentation

Installation guide and user's manuals for the software are supplied with the products.

SUPPORT

Hotline Telephone Support, Software, Performance Report (SPR) Software Updates, Technical Reports and Monthly Newsletters are available.

ORDERING INFORMATION

Product Code	Description
iMDX-371VX	ASM-286, BLD-286, BND-286, LIB-286, MAP-286
iMDX-373VX	PL/M-286

*VAX,/VMS are trademarks of Digital Equipment Corporation

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*/VMS* RESIDENT iAPX-86/88/186 SOFTWARE DEVELOPMENT PACKAGES

- **Executes on DEC VAX* Minicomputer under VMS* Operating System to translate PL/M-86, Pascal-86 and ASM-86 Programs for iAPX-86, 88 and 186 Microprocessors.**
- **Packages Include Pascal-86; PL/M-86; ASM-86; Link and Relocation Utilities; OH-86 Absolute Object Module to Hexadecimal Format Converter; and Library Manager Program.**
- **Output linkable with Code Generated on Intellec® Development Systems.**

The VAX/VMS Resident Software Development Packages contain software development tools for the iAPX-86, 88, and 186 microprocessors. The package lets the user develop, compile, maintain libraries, and link and locate programs on a VAX running the VMS operating system. The translator output is object module compatible with programs translated by the corresponding version of the translator on an Intellec Development System.

Three packages are available:

1. An ASM-86 Assembler Package which includes the Assembler, the Link Utility, the Locate Utility, the absolute object to hexadecimal format conversion utility and the Library Manager Program.
2. A PL/M-86 Compiler Package which contains the PL/M-86 Compiler and Runtime Support Libraries.
3. A Pascal-86 Compiler Package which contains the Pascal-86 Compiler and Runtime Support Libraries.

The VAX/VMS resident development packages and the Intellec Development System development packages are built from the same technology base. Therefore, the VAX/VMS resident development packages and the Intellec Development System development packages are very similar.

Version numbers can be used to identify features correspondence. The VAX/VMS resident development packages will have the same features as the Intellec Development System product with the same version number.

Support for the iAPX-186 processor will be provided as an update to the iAPX-86, 88 software.

The object modules produced by the translators contain symbol and type information for programming debugging using ICE™ translators and/or the PSCOPE debugger. For final production version, the compiler can remove this extra information and code.

*VAX, DEC, and VMS are trademarks of Digital Equipment Corporation.

Intel Corporation Assumes No Responsibility for the Use of Any Circuitry Other Than Circuitry Embodied in an Intel Product. No Other Circuit Patent Licenses are Implied. Information Contained Herein Supercedes Previously Published Specifications of These Devices from Intel.
©INTEL CORPORATION, 1983

JUNE 1984
ORDER NUMBER: 210643-002

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*-PL/M-86/88/186 SOFTWARE PACKAGE

- **Executes on VAX* Minicomputer Under the VMS* Operating System**
- **Supports 16-Bit Signed Integer and 32-Bit Floating Point Arithmetic in Accordance with IEEE Proposed Standard**
- **Easy-To-Learn Block-Structured Language Encourages Program Modularity**
- **Produces Relocatable Object Code Which is Linkable to All Other Intel 8086 Object Modules, Generated on Either a VAX* or Intellec® Development Systems**
- **Code Optimization Assures Efficient Code Generation and Minimum Application Memory Utilization**
- **Built-In Syntax Checker Doubles Performance for Compiling Programs Containing Errors**
- **Source Input/Object Output Compatible with PL/M-86 Hosted on an Intellec® Development System**
- **ICE™, PSCOPE Symbolic Debugging Fully Supported**

Like its counterpart for MCS®-80/85 program development, and Intellec® hosted iAPX-86 program development, VAX-PL/M-86 is an advanced, structured high-level programming language. The VAX-PL/M-86 compiler was created specifically for performing software development for the Intel iAPX-86, 88, and 186 Microprocessors.

PL/M is a powerful, structured, high-level system implementation language in which program statements can naturally express the program algorithm. This frees the programmer to concentrate on the logic of the program without concern for burdensome details of machine or assembly language programming (such as register allocation, meanings of assembler mnemonics, etc.).

The VAX-PL/M-86 compiler efficiently converts free-form PL/M language statements into equivalent iAPX-86/88/186 machine instructions. Substantially fewer PL/M statements are necessary for a given application than if it were programmed at the assembly language or machine code level.

The use of PL/M high-level language for system programming, instead of assembly language, results in a high degree of engineering productivity during project development. This translates into significant reductions in initial software development and follow-on maintenance costs for the user.

*VAX, DEC, and VMS are trademarks of Digital Equipment Corporation

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*-PASCAL-86/88 SOFTWARE PACKAGE

- **Executes on VAX* Minicomputer Under the VMS* Operating System**
- **Produces Relocatable Object Code Which Is Linkable to All Other Intel 8086 Object Modules, Generated on Either a VAX* or Intellec® Development Systems**
- **ICE™, PSCOPE Symbolic Debugging Fully Supported**
- **Implements REALMATH for Consistent and Reliable Results**
- **Supports iAPX-86/20, 88/20 Numeric Data Processors**
- **Strict Implementation of ISO Standard Pascal**
- **Useful Extensions Essential for Microcomputer Applications**
- **Separate Compilation with Type-Checking Enforced Between Pascal Modules**
- **Compiler Option to Support Full Run-Time Range-Checking**
- **Source Input/Object Output Compatible with Pascal-86 Hosted on a Intellec Development System**

VAX PASCAL 86 conforms to and implements the ISO Pascal standard. The language is enhanced to support microcomputer applications with special features, such as separate compilation, interrupt handling and direct port I/O. Other extensions include additional data types not required by the standard and miscellaneous enhancements such as an allowed underscore in names, an OTHERWISE clause in CASE construction and so forth. To assist the development of portable software, the compiler can be directed to flag all non-standard features

The VAX-PASCAL-86 compiler runs on the Digital Equipment Corporation VAX under the VMS Operating System. A well-defined I/O interface is provided for run-time support. This allows a user-written operating system to support application programs on the target system as an alternate to the development system environment. Program modules compiled under PASCAL-86 are compatible and linkable with modules written in PL/M-86, and ASM-86. With a complete family of compatible programming languages for the iAPX-86, 88, and 186 one can implement each module in the language most appropriate to the task at hand

*VAX, DEC, and VMS are trademarks of Digital Equipment Corporation

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*-iAPX-86/88/186 MACRO ASSEMBLER

- **Executes on VAX* Minicomputer Under The VMS* Operating System**
- **Produces Relocatable Object Code Which Is Linkable to All Other Intel iAPX-86/88/186 Object Modules, Generated on Either a VAX* or Intellec® Development Systems**
- **Powerful and Flexible Text Macro Facility with Three Macro Listing Options to Aid Debugging**
- **Highly Mnemonic and Compact Language, Most Mnemonics Represent Several Distinct Machine Instructions**
- **"Strongly Typed" Assembler Helps Detect Errors at Assembly Time**
- **High-Level Data Structuring Facilities Such as "STRUCTURES" and "RECORDS"**
- **Over 120 Detailed and Fully Documented Error Messages**
- **Produces Relocatable and Linkable Object Code**
- **Source Input/Object Output Compatible with ASM-86 hosted on an Intellec Development System**

VAX-ASM-86 is the "high-level" macro assembler for the iAPX-86/88/186 assembly language. VAX-ASM-86 translates symbolic iAPX-86/88/186 assembly language mnemonics into iAPX-86/88/186 relocatable object code

VAX-ASM-86 should be used where maximum code efficiency and hardware control is needed. The iAPX-86/88/186 assembly language includes approximately 100 instruction mnemonics. From these few mnemonics the assembler can generate over 3,800 distinct machine instructions. Therefore, the software development task is simplified, as the programmer need know only 100 mnemonics to generate all possible iAPX-86/88/186 machine instructions. VAX-ASM-86 will generate the shortest machine instruction possible given no forward referencing or given explicit information as to the characteristics of forward referenced symbols

VAX-ASM-86 offers many features normally found only in high-level languages. The iAPX-86/88/186 assembly language is strongly typed. The assembler performs extensive checks on the usage of variable and labels. The assembler uses the attributes which are derived explicity when a variable or label is first defined, then makes sure that each use of the symbol in later instructions conforms to the usage defined for that symbol. This means that many programming errors will be detected when the program is assembled, long before it is being debugged on hardware

*VAX, DEC, and VMS are trademarks of Digital Equipment Corporation

AFN-00680C

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*-LIB-86

- **Executes on VAX* Minicomputer Under the VMS* Operating System**
- **VAX*-LIB-86 is a Library Manager Program which Allows You to:**
  - **Create Specifically Formatted Files to Contain Libraries of Object Modules**
  - **Maintain These Libraries by Adding or Deleting Modules**
  - **Print a Listing of the Modules and Public Symbols in a Library File**
- **Libraries Can be Used as Input to VAX*-LINK-86 Which Will Automatically Link Modules from the Library that Satisfy External References in the Modules Being Linked**
- **Abbreviated Control Syntax**

Libraries aid in the job of building programs. The library manager program VAX-LIB-86 creates and maintains files containing object modules. The operation of VAX-LIB-86 is controlled by commands to indicate which operation VAX-LIB-86 is to perform. The commands are:

CREATE:	creates an empty library file
ADD:	adds object modules to a library file
DELETE:	deletes modules from a library file
LIST:	lists the module directory of library files
EXIT:	terminates the LIB-86 program and returns control to VMS

When using object libraries, the linker will call only those object modules that are required to satisfy external references, thus saving memory space.

VAX-OH-86

- **Executes on VAX* Minicomputer Under the VMS* Operating System**
- **Converts an iAPX 86/88/186 Absolute Object Module to Symbolic Hexadecimal Format**
- **Facilitates Preparing a file for Loading by Symbolic Hexadecimal Loader (e.g. iSBC™ Monitor SDK-86 Loader), or Universal PROM Mapper**
- **Converts an Absolute Module to a More Readable Format that can be Displayed on a CRT or Printed for Debugging**

The VAX-OH-86 utility converts an 86/88 absolute object module to the hexadecimal format. This conversion may be necessary for later loading by a hexadecimal loader such as the iSBC 86/12 monitor or the Universal PROM Mapper. The conversion may also be made to put the module in a more readable format that can be displayed or printed.

The module to be converted must be in absolute form; the output from VAX-LOC-86 is in absolute format.

*VAX, VMS are trademarks of Digital Equipment Corporation.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*-LINK-86

- **Executes on VAX* Minicomputer Under the VMS* Operating System**
- **Automatic Combination of Separately Compiled or Assembled 86/88/186 Programs Into a Relocatable Module, Generated on Either a VAX or an Intellec® Development System**
- **Automatic Selection of Required Modules from Specified Libraries to Satisfy Symbolic References**
- **Extensive Debug Symbol Manipulation, allowing Line Numbers, Local Symbols, and Public Symbols to be Purged and Listed Selectively**
- **Automatic Generation of a Summary Map Giving Results of the LINK-86 Process**
- **Abbreviated Control Syntax**
- **Relocatable modules may be Merged into a Single Module Suitable for Inclusion in a Library**
- **Supports "Incremental" Linking**
- **Supports Type Checking of Public and External Symbols**

VAX-LINK-86 combines object modules specified in the VAX-LINK-86 input list into a single output module VAX-LINK-86 combines segments from the input modules according to the order in which the modules are listed

VAX-LINK-86 will accept libraries and object modules built from VAX-PL/M-86, VAX-PASCAL-86, VAX-ASM-86, or any other Intel translator generating 8086 Relocatable Object Modules, such as the Series III resident translators.

Support for incremental linking is provided since an output module produced by VAX-LINK-86 can be an input to another link At each stage in the incremental linking process, unneeded public symbols may be purged

VAX-LINK-86 supports type checking of PUBLIC and EXTERNAL symbols reporting a warning if their types are not consistent

VAX-LINK-86 will link any valid set of input modules without any controls However, controls are available to control the output of diagnostic information in the VAX-LINK-86 process and to control the content of the output module

VAX-LINK-86 allows the user to create a large program as the combination of several smaller, separately compiled modules After development and debugging of these component modules the user can link them together, locate them using VAX-LOC-86 and enter final testing with much of the work accomplished.

*VAX, DEC and VMS are trademarks of Digital Equipment Corporation

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

VAX*-LOC-86

- **Executes on the VAX* Minicomputer Under the VMS* Operating System**
- **Automatic Generation of a Summary Map Giving Starting Address, Segment Addresses and Length, and Debug Symbols and their Addresses**
- **Extensive Capability to Manipulate the Order and Placement of Segments in 8086/8088 Memory**
- **Abbreviated Control Syntax**
- **Automatic and Independent Relocation of Independent Relocation of Segments. Segments May be Relocated to Best Match Users Memory Configuration**
- **Extensive Debug Symbol Manipulation, Allowing Line Numbers, Local Symbols, and Public Symbols to be Purged and Listed Selectively**

Relocatability allows the programmer to code programs or sections of programs without having to know the final arrangement of the object code in memory.

VAX-LOC-86 converts relative addresses in an input module in iAPX-86/88/186 object module format to absolute addresses. VAX-LOC-86 orders the segments in the input module and assigns absolute addresses to the segments. The sequence in which the segments in the input module are assigned absolute addresses is determined by their order in the input module and the controls supplied with the command.

VAX-LOC-86 will relocate any valid input module without any controls. However, controls are available to control the output of diagnostic information in the VAX-LOC-86 process, to control the content of the output module, or both.

The program you are developing will almost certainly use some mix of random access memory (RAM), read-only memory (ROM), and/or programmable read-only memory (PROM). Therefore, the location of your program affects both cost and performance in your application. The relocation feature allows you to develop your program and then simply relocate the object code to suit your application.

SPECIFICATIONS

Operating Environment

Required Hardware

VAX* 11/780, 11/782, 11/750, or 11/730
9 Track Magnetic Tape Drive, 1600 BPI

Required Software

VMS Operating System V3.0 or Later. All of the development packages are delivered as unlinked VAX object code which can be linked to VMS as designed for the system where the development package is to be used. VMS command files to perform the link are provided.

*VAX, DEC, and VMS are trademarks of Digital Equipment Corporation

Documentation Package

iAPX-86, 88 Development Software Installation Manual and User's Guide for VAX/VMS, Order number 121950-001

Shipping Media

9 Track Magnetic Tape 1600 bpi

ORDERING INFORMATION

Part Number	Description
iMDX-341VX	VAX-ASM-86, VAX-LINK-86, VAX-LOC-86, VAX-LIB-86, VAX-OH-86, Package
iMDX-343VX	VAX-PLM-86 Package
IMDX-344VX	VAX-PASCAL-86 Package

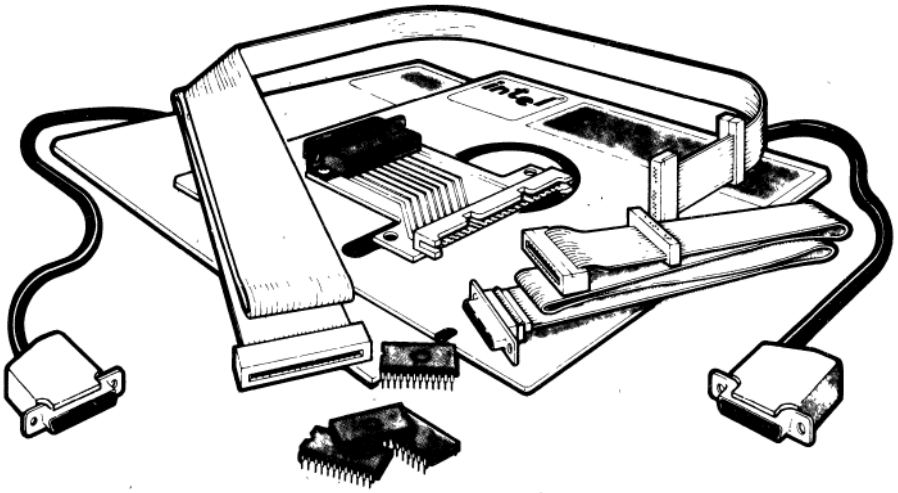
REQUIRES SOFTWARE LICENSE

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

iSDM™ 86
SYSTEM DEBUG MONITOR

- **Supports target system debugging for iSBC® /iAPX 86, 88, 186 and 188-based applications**
- **Provides interactive debugging commands including single-step code execution and symbolic displays of results**
- **Supports 8087 Numeric Processor Extension (NPX) for high-speed math applications**
- **Allows building of custom commands through the Command Extension Interface (CEI)**
- **Supports application access to ISIS-II files**
- **Provides program load capability from an Intellec® Development System**
- **Contains configuration facilities which allow an applications bootstrap from iRMX™ 86 and 88 file compatible peripherals**
- **Modular to allow use from an Intellec Development System or from a stand-alone terminal**

The Intel iSDM™ 86 System Debug Monitor package contains the necessary hardware, software, cables, EPROMs and documentation required to interface, through a serial or parallel connection, an iSBC® 86/05, 86/12A, 86/14, 86/30, 88/25, 88/40, 88/45, 186/03, 186/51, 188/48, or iAPX 86, 88, 186 or 188 target system to an MDS 800, Series II, Series III, or Series IV Intellec® Microcomputer Development System for execution and interactive debugging of applications software on the target system. The Monitor can: load programs into the target system; execute the programs instruction by instruction or at full speed; set breakpoints; and examine/modify CPU registers, memory content, and other crucial environmental details. Additional custom commands can be built using the Command Extension Interface (CEI). The Monitor supports the OEM's choice of the iRMX™ 86 Operating System, the iRMX 88 Real-Time Multi-tasking Executive or a custom system for the target application system. OEM's may utilize any iRMX 86, 88 supported target system peripheral for a bootstrap of the application system or have full access to the ISIS-II files of the Intellect System.



The following are trademarks of Intel Corporation and may be used only to describe Intel products: Intel ICE, iMMX, iRMX, iSBC, iSBX, iSXM, MULTIBUS, Multichannel and MULTIMODULE. Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied. Information contained herein supercedes previously published specifications on these devices from Intel.

INTEL CORPORATION, 1983

October, 1983
ORDER NUMBER: 230882-002

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FUNCTIONAL DESCRIPTION

Overview

The iSDM 86 Monitor extends the software development capabilities of the Intellec system so the user can effectively develop applications to ensure timely product availability.

The iSDM 86 package consists of four parts:

- The loader program
- The iSDM 86 Monitor
- The Command Extension Interface (CEI)
- The ISIS-II Interface

The user can use the ISDM 86 package to load programs into the target system from the development system, execute programs in an instruction-by-instruction manner, and add custom commands through the command extension interface. The user also has the option of using just the iSDM 86 Monitor and the CEI in a stand-alone application, without the use of an Intellec development system.

Powerful Debugging Commands

The iSDM 86 Monitor contains a powerful set of commands to support the debugging process. Some of the features included are: bootstrap of application software; selective execution of program modules based on breakpoints or single stepping requests; examination, modification and movement of memory contents; examination and modification of CPU registers, including NPX registers. All results are displayed in clearly understandable formats. Refer to Table 1 for a more detailed list of the iSDM 86 monitor commands.

Numeric Data Processor Support

Arithmetic applications utilizing the 8087 Numeric Processor Extension (NPX) are fully supported by the iSDM 86 Monitor. In addition to executing applications with the full NPX performance, users may examine and modify the NPX's registers using decimal and real number format.

This feature allows the user to feel confident that correct and meaningful numbers are entered for the application without having to encode and decode complex real, integer, and BCD hexadecimal formats.

Command Extension Interface (CEI)

The Command Extension Interface (CEI) allows the addition of custom commands to the iSDM 86 Monitor commands. The CEI consists of various procedures that can be used to generate custom commands. Up to three custom commands (or sets of commands) can be added

Table 1. Monitor Commands

Command	Function
B	**Bootstrap** application program from target systems peripheral device
C	**Compare** two memory blocks
D	**Display** contents of memory block
E*	**Exit** from loader program to ISIS-II Interface
F	**Find** specified constant in a memory block
G	**Execute** application program
I	**Input** and display data obtained from input port
L*	**Load** absolute Intellec® object file into target system memory
M	**Move** contents of memory block to another location
N	**Display and execute** single instruction
O	**Output** data to output port
P	**Print** values of literals
R*	**Load and execute** absolute Intellec® object file in target system memory
S	**Display and (optionally) modify** contents of memory
T*	**Transfer** block of memory to an Intellec® file
U,V,W	**User** defined custom commands extensions
X	**Examine and (optionally) modify** CPU and NPX registers

* Commands require an attached Series II/Series III.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

to the monitor without programming new EPROMs or changing the monitor's source code.

ISIS-II Interface

The ISIS-II interface consists of libraries which contain interfaces to ISIS-II I/O calls. A program running on an iAPX 86, 88, 186 or 188-based system can use the ISIS-II interface and access the individual ISIS-II I/O calls. The interface allows the inclusion of these calls into the program; however, most of the calls require a Series II/Series III system. Table 2 contains a summary of the major I/O calls and parameters.

Program Load Capability

The iSDM 86 loader allows the loading of iAPX 86, 88, 186 or 188-based programs into the target system. It executes on a Intellec Microcomputer Development System and communicates with the target system through a serial or a parallel load interface. If a Series II/Series III/Series IV system containing an Intel I/O expansion board is being used, the board can be used as a fast parallel load interface, freeing up the UPP port for application use.

Configuration Facility

The monitor contains a full set of configuration facilities which allow it to be carefully tailored to the requirements of the target system. Pre-configured EPROM-resident monitors are supplied by Intel for the iSBC 86/05, 86/12A, 86/14, 86/30, 88/45, 186/03, 186/51, and 188/48 boards. The monitor must be configured by the user for the iSBC 88/25, 88/40 boards and for other iAPX 86, 88, 186, 188 applications. iRMX 86 and iRMX 88 system users may use the configuration facilities to include the iAPX 86, 88 Bootstrap Loader (V5.0 or newer) in the monitor.

Variety of Connections Available

The physical interface between the Intellec Microcomputer Development System and the target system can be established in one of three ways. The systems can be connected via a serial link, a parallel link or a fast parallel link. The fast parallel link requires the use of an iSBC 108(A), 116(A), 517 or 519 I/O expansion board in the Intellec system and is only available for connections with the Series II/Series III/Series IV systems. The cabling arrangement is different depending upon the development system being used. Figure 1 displays the cable connections needed between an Intellec Series III system and a target system for a serial interface.

The iSDM 86 Monitor does not require the use of a development system. The monitor can be used by simply attaching a stand-alone terminal to the target system. Figure 1 also displays the cable connections needed for this arrangement.

Table 2. Routines for ISIS-II Services Available to Target System Applications

Routine	Target System Function
ATTRIB	Changes to ISIS-II file **attribute**
CI	Returns a character **input** from the **console**
CO	Transfers a character for **console output**
CLOSE	**Closes** an opened ISIS-II file
DELETE	**Deletes** the specified ISIS-II file
DQ$CFG	Returns information about monitor's communication link and type
ERROR	Displays an **error** message on the Intellec® console
EXIT	**Exits** to the target system monitor
LOAD	**Loads** target system memory with ISIS-II object code file
OPEN	**Opens** an ISIS-II file for access
READ	**Reads** up to 4096 bytes from an ISIS-II file to memory
RENAME	**Renames** an ISIS-II disk file
SEEK	**Seeks** to the specified ISIS-II file location
WRITE	**Writes** up to 4096 bytes from memory to an ISIS-II file

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

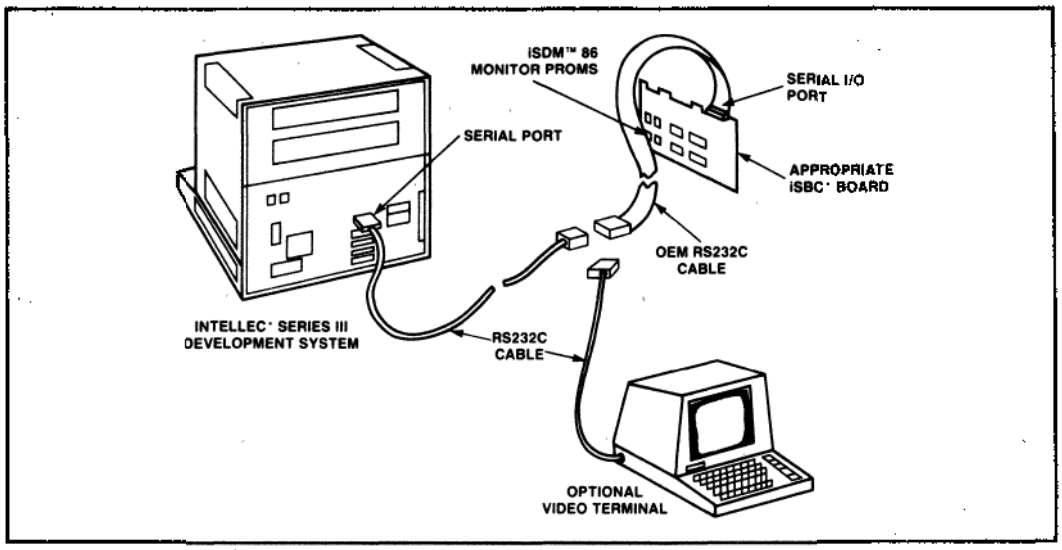


Figure 1. Typical iSDM™ 86 Serial Connection Environment

SPECIFICATIONS

Development System Environment

The Intellec Microcomputer Development System may be utilized for application program development and, if used, requires the following to support the iSDM 86 package:

- 48 Kbytes memory
- Double density or single density diskette subsystem
- ISIS-II Operating System and associated language translators

iAPX 86, 88, 186, 188 TARGET SYSTEM ENVIRONMENT

To support the iSDM 86 package, the target system must contain the following:

- 2K read-write memory beginning at location 0H
- 16K read-only memory beginning at location FC000H
- For Parallel link:
  — 8255A Programmable Peripheral Interface
- For Serial link:
  — 8251A USART or 8274 Multiprotocol Serial Controller, and 8253/4 or 80130 or iAPX 186/188 timer, or
  — 82530 Serial Communications Controller, including 82530 timer

Hardware

- Supported iSBC Microcomputers:

iSBC 86/05	Single Board Computer
iSBC 86/12A	Single Board Computer
iSBC 86/14	Single Board Computer
iSBC 86/30	Single Board Computer
iSBC 88/25	Single Board Computer
iSBC 88/40	Single Board Computer
iSBC 88/45	Single Board Computer
iSBC 186/03	Single Board Computer
iSBC 186/51	Single Board Computer
iSBC 188/48	Single Board Computer

- Supported iSBX MULTIMODULE™ Boards:

  iSBX 350 Parallel I/O MULTIMODULE Board
  iSBX 351 Serial I/O MULTIMODULE Board

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

iSDM™ 86 Package Contents

Cables:

1 — Parallel I/O Cable (upload/download)

2 — RS232 Cables

Adaptors:

1 — Parallel Status Adaptor

1 — Parallel Adaptor

I/O Drivers and Terminators:

4 — Pull-up Resistor Packs

4 — Pull-up/down Resistor Packs

4 — Line Driver Packs

Interface and Execution Software Diskettes:

1 — Single Density, ISIS Compatible

1 — Double Density, ISIS Compatible

System Monitor EPROMs:

Microcomputer	EPROM
iSBC® 86/05 iSBC® 86/12A iSBC® 86/14 iSBC® 86/30	Four 2732A EPROMs
iSBC® 88/45	Two 2764 EPROMs
iSBC® 186/03 iSBC® 186/51	Two 2764 EPROMs
iSBC® 188/48	Two 2764 EPROMs

Reference Manual (Supplied):

146165-001 — iSDM 86 System Debug Monitor Reference Manual

ORDERING INFORMATION

Part Number	Description
iSDM 86	Intellec to target system interface and target system monitor, suitable for use on iSBC 86, 88, 186, 188 computers, or other iAPX 86, 88, 186, 188 microcomputers. Package includes cables, EPROMs, software and operator manual. The iSDM 86 package includes SPR Service for 90 days after shipment. As with all Intel Software, purchase of any of these options requires execution of a standard Intel Master Software License.
iSDM 86 RO	Object Software
iSDM 86 BSR	Machine Readable Source

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

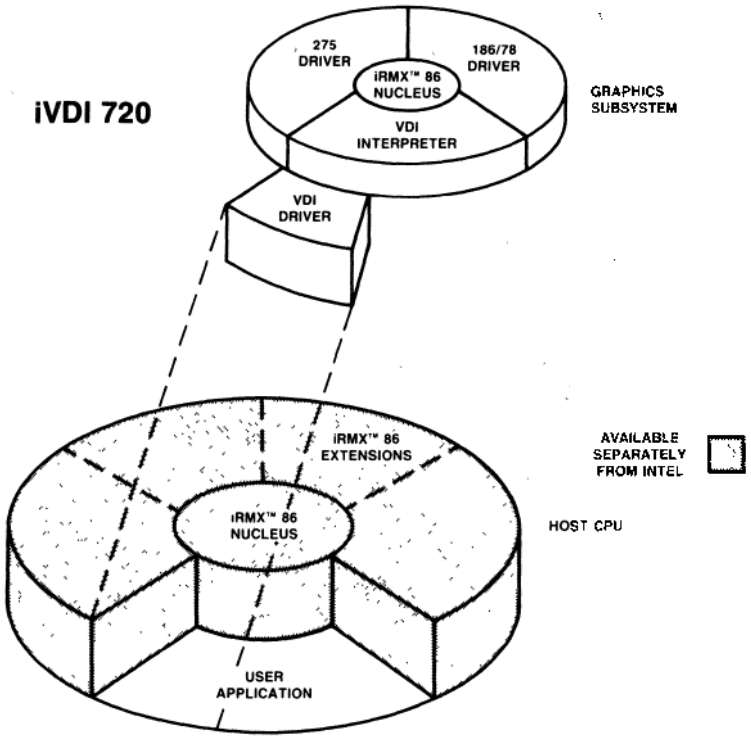
PRELIMINARY

iVDI 720
GRAPHICS VIRTUAL DEVICE Interpreter

- **Provides standardized decoding of high-level graphics commands**
- **Full iRMX™ 86 compatibility operating system (Rel. 6)**
- **Standardized input & output drivers**
- **Compact for EPROM installation**
- **Support for iSBX™ 275 and iSBC® 186/78 Graphics hardware modules**
- **Procedural interface from Pascal 86, PL/M 86 and Fortran 86**
- **Compatible with (proposed) ANSI X3H33 specification**
- **Virtual Device Metafile interpreter**

The Intel iVDI 720 Graphics Virtual Device Interpreter provides both a powerful library of high-level commands, and the drivers necessary to support the iSBX™ 275 or iSBC® 186/78 graphics modules in an iRMX™ 86 (Release 6) environment. It allows the OEM to quickly tailor an Intel system for application into the rapidly growing graphics marketplace, especially low-cost CAD/CAE, CAM, and process control. Individual single-board computer (SBC) modules may also be configured from Intel's broad product family.

For intra-systems graphics control, iVDI 720 is the most powerful and efficient product available that brings (proposed) ANSI X3H33 compatability to an iRMX 86 operating system environment.



The following are trademarks of Intel Corporation and may only be used to describe Intel Products: Intel, ICE, iRMX, iSBC, iSBX, iSXM, MULTIBUS, MULTICHANNEL, MULTIMODULE, and ICS. Intel Corporation assumes no responsibility for the use of any circuitry other than the circuitry embodied in an Intel Product. No other circuit patent licenses are implied

©INTEL CORPORATION, 1984

JUNE, 1984
ORDER NUMBER: 280002-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FUNCTIONAL DESCRIPTION

Graphics Standard Software

The iVDI 720 Graphics Virtual Device Interpreter implements the proposed ANSI standard on any Intel-based graphics system running under the iRMX 86 operating system, release 6. The proposed standard is a significant advancement in graphics software. It creates a predictable environment for the input and output of high-level commands between the user and system, or among the graphics peripherals attached to the system, such as a mouse, tablet, printer or plotter. The software supports two environments: stand-alone and distributed, depending on the hardware configuration.

All elements of iVDI 720 can run as tasks of the operating system or as part of the graphics application program, hence a stand-alone partitioning of graphics activities such as with the iSBX 275 MULTIMODULE™ attached to a general purpose CPU board like the iSBC 86/30. In a distributed environment, the device driver runs under the iRMX 86 operating system and the remaining application code and VDI interpreter are exercised by a separate processor dedicated to graphics activities. The iSBC 186/78 subsystem was designed especially for the distributed solution.

iSBC® 186/78 Graphics Subsystem Support

By virtue of its on-board, high integration microprocessor (the Intel 80186), the iSBC 186/78 subsystem is an excellent platform on which to perform graphics routines in a distributed environment. This is particularly important in multi-user systems where one iSBC 186/78 subsystem can be dedicated to each user. (see figure 1)

The compact coding of the iVDI 720 Graphics Virtual Device Interpreter lends itself to EPROM installation on the iSBC 186/78 subsystem. The host CPU board is thereby off-loaded from graphics activities so it can direct more global system level operations such as database management or network communications.

iSBX™ 275 Graphics MULTIMODULE™ Support

In single-user applications or where graphics activities are not the major focus of the system, the iSBX 275 MULTIMODULE shares the CPU on the host processor board through the iSBX expansion bus. The subsystem formed in this manner supports either monochrome or eight colors and is a very cost-effective solution. Like the iSBC 186/78 subsystem,

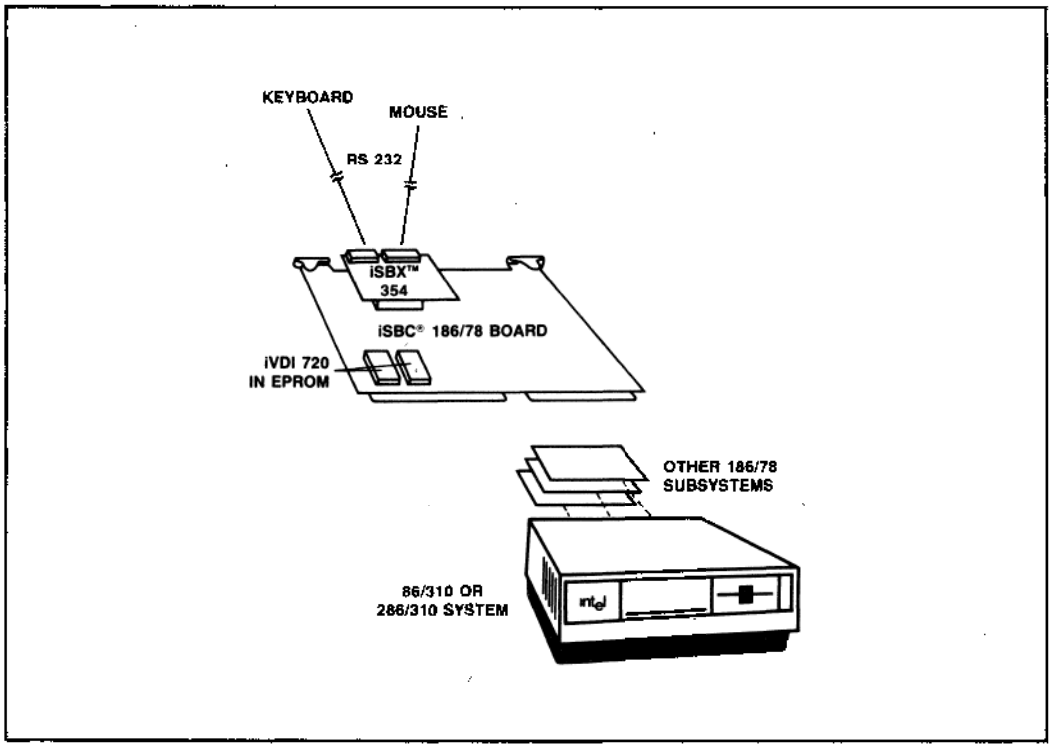


Figure 1. Multi-User Example

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

this expansion module is based on the Intel 82720 Graphics Display Controller (GDC) component. (see figure 2)

For example using an iSBC 86/30 CPU board, the iVDI 720 library can be installed in EPROM to simplify the application and provide higher performance execution.

82720 Component Designs

The Intel 82720 GDC is an intelligent graphics controller component designed to operate as the heart of a raster-scan computer graphics display system. The 82720 performs all the basic timing needed to generate the raster display and manage the display memory. In addition, it supports several high-level graphics figure drawing functions. The Intel 82720 is an alternative to the NEC 7220 component.

VDI COMMAND LIBRARY

In addition to providing driver support for Intel's growing family of graphics modules, the iVDI 720 Graphics Virtual Device Interpreter decodes a wealth of high-level commands to streamline the development of application code for a variety of graphics devices.

The proposed ANSI standard provides multiple encodings of high-level text and graphics commands and capabilities. The iVDI 720 software decodes a binary representation of these proposed commands, along with the Virtual Device Metafile (VDM) routines that allow consistant formatting and storage of VDI encoded images.

In addition to a full set of inquiry functions, many additional high-level commands are supported in the iVDI 720 software. (See Table 1)

These features are configurable as defined in the iVDI 720 Software Reference Manual. However, they are typically device dependent and therefore reflect the users application. Consequently, the reference manual should be consulted to assure compatability.

DEVELOPMENT ENVIRONMENT

Intel's family of development systems and their extensions are highly recommended for both the development of iVDI 720 and related application code. Languages that are supported include Fortran 86, Pascal 86 and PL/M 86. All iVDI 720 commands can be called from any of these programming languages through the PL/M 86 procedural interface that is integral to the iVDI 720 product.

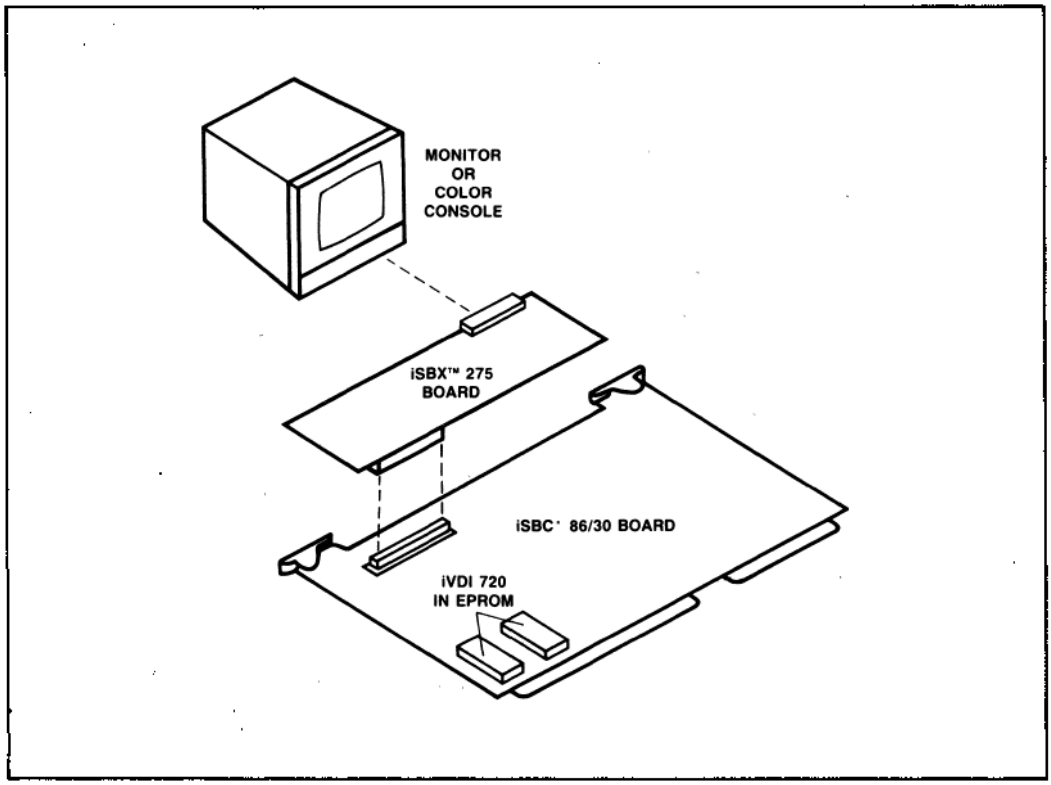


Figure 2. Single-User Example

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 1. iVDI 720 Command Library

Graphical Elements:

- Polyline
- Polygon
- Arc
- Text
- Cell Array
- Polymarker
- Circle
- Arc Close (Pie or Chord)
- Append Text

Attribute Elements:

- Aspect Source Flags
- Character Orientation
- Character Path
- Character Spacing
- Text Alignment
- Perimeter Type & Color
- Hatch Fill
- Pattern Fill
- Pattern Definition
- Text Precision
  - String
  - Character
  - Stroke
- Bundled & Individual Attributes
- Character Height
- Character Expansion Factor
- Interior Style
- Marker Type & Color
- Line Type & Color
- Set Color Table
- Pattern Size
- Pattern Reference Point
- Text Color

Control & Descriptor Elements:

- Begin & End Metafile
- Background Color
- Clip Rectange
- Clear Surface
- Set Device Viewpoint
- Scaling Mode
- Marker Size Mode
- Begin & End Picture
- VDC Extent
- Clip Indicator
- Defaults Replacement
- Color Direct Precision
- Color Specification Mode

Input Elements:

- Initialize Locator
- Sample Locator
- Request Locator
- Set Prompt State
- Release Input Device
- Initialize String
- Sample String
- Request Locator
- Set Echo State
- Set Input Device Mode

SPECIFICATIONS

ANSI X3H33 VDI Specification

The American National Standards Institute (ANSI) administers the standard specification. Requests for information should be directed to:

X3 Secretariat
Computer Business Equipment Manufacturers Association (CBEMA)
311 First Street, NW
Washington, D.C. 20001

Intel is heavily involved in the development of the ANSI X3H33 Virtual Device Interface standard. We will endeavor to bring to our user base the latest revisions through phased introductions and updates. Consequently, it is strongly advised that implementers of iVDI 720 also subscribe to the update service (VDI 720 WX, see below).

iVDI 720 Specifications

Code size — 80 Kbytes in distributed mode (using the iSBC 186/78 subsystem), including the iRMX 86 nucleus

Code size — 64 Kbytes in stand-alone mode (using the iSBX 275 MULTIMODULE)

Source-code language — PL/M 86

Related Literature

Reference material may be ordered from any Intel sales represenative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, Calif., 95051.

146717 — iVDI 720 Software Reference Manual

210506 — iSBX 275 Video Graphics Controller Data Sheet

231035 — iSBC 186/78 Video Graphics Subsystem Data Sheet

146666 — iSBC 186/78 Video Graphics Subsystem Hardware Reference Manual

210655 — 82720 GDC Component Data Sheet

9803126 — iRMX 86 Configuration Guide

Ordering Information

Intel makes available a variety of licensing programs to the iVDI 720 Graphics Virtual Device Interpreter which allow different plans for incorporation of the Intel software into the final product. The Intel Master software Agreement should be consulted to determine which plan is best suited for the particular application and production environment.

The iVDI 720 Graphics Virtual Device Interpreter comes in three formats as shown below, along with source listings and update services. The iRMX 86-Real-time Multitasking Operating System is available separately.

iVDI 720RO OEM license (8 inch single-sided/double densite ISIS and iRMX plus 5¼ inch double-sided/double density iRMX formats are supplied)

IVDI 720RF Incorporation fee payment

iVDI 720WX Object code update

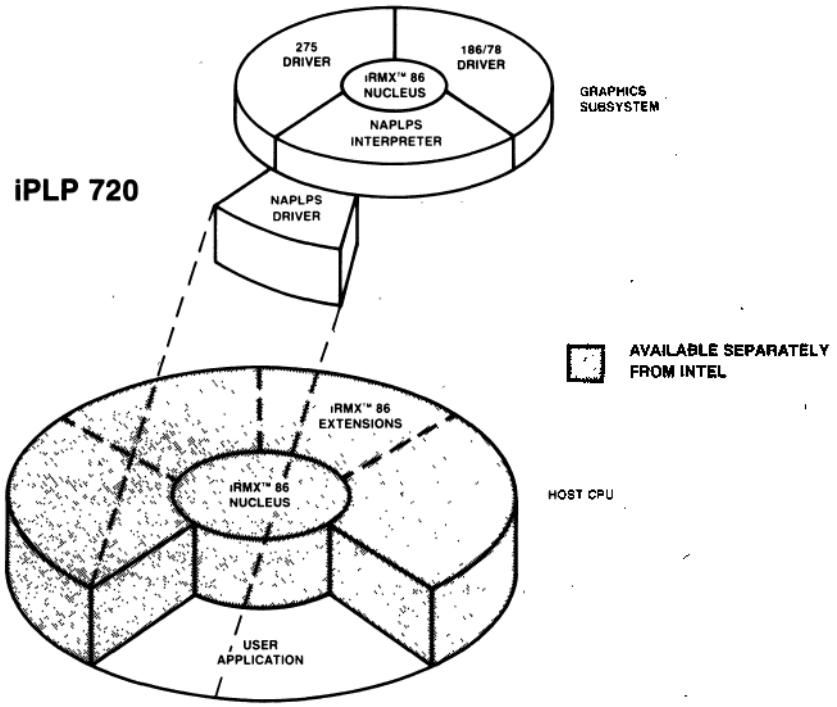
Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

iPLP 720
NAPLPS Interpreter

- **Provides decoding of NAPLPS (Videotex) commands**
- **Multiple font sizes and definitions**
- **Full iRMX™ 86 compatibility**
- **Compact for EPROM installation**
- **Compatible with ANSI BSR X3.110 -1983**
- **Driver support for Intel Graphics hardware modules**
- **Complete library of high-level frame management instructions**
- **Resolution Independant presentation**
- **Simplified communication of Graphic images between systems**

The Intel iPLP 720 NAPLPS (North American Presentation Level Protocol Syntax) Interpreter provides both a powerful library of high-level commands, and the drivers necessary to support the iSBX™ 275 or iSBC 186/78 Graphics Modules in an iRMX™ 86 Operating System (Release 5 or later) environment. It allows the OEM to quickly tailor an Intel system for application into the rapidly growing Videotex marketplace. Individual iSBC® modules may also be configured from Intel's broad product family, and iPLP 720 will also be convenient for the those implementing custom, component designs based on the Intel 82720 Graphics Display Controller (GDC).

Regardless of the hardware configuration, iPLP 720 is the most powerful and efficient product available that brings full ANSI X3L2 Videotex compatibility to an iRMX 86 environment.



The following are trademarks of Intel Corporation and may be used only to describe Intel products: Intel, ICE, iMMX, iRMX, iSBC, iSBX, iSXM, MULTIBUS, Multichannel and MULTIMODULE. Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied. Information contained herein supercedes previously published specifications on these devices from Intel

© INTEL CORPORATION, 1984

May, 1984
Order Number: 280001-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FUNCTIONAL DESCRIPTION

NAPLPS Interpreter

The iPLP 720 software implements the NAPLPS videotex standard on any 82720-based graphics application using the iRMX 86 Operating System, Release 5 or later. The NAPLPS Standard is a significant advancement in device-level graphics software. It creates a predictable environment for development of presentation frames along with a transmission scheme to dramatically improve performance when moving graphics images from one system to another. The software supports two environments: stand-alone and distributed, depending on the hardware configuration.

All elements of iPLP 720 can run as tasks of the operating system or as part of the graphics application program, hence a stand-alone partitioning of graphics activities such as with the iSBX 275 MULTIMODULE™. In a distributed environment, the device driver runs under the iRMX 86 Operating System and the remaining controller code and NAPLPS interpreter are exercised by a separate processor dedicated to graphics activities. The iSBC 186/78 subsystem was designed especially for the distributed solution.

This architecture also allows iPLP 720 to run under non-iRMX environments.

iSBC® 186/78 Video Graphics Subsystem Support

By virtue of its on-board, high integration microprocessor (the Intel 80186), the iSBC 186/78 subsystem is an excellent platform on which to perform graphics routines in a distributed environment. This is particularly important in multi-user systems where one iSBC 186/78 subsystem can be dedicated for each user. The host CPU board is thereby off-loaded to direct more global system level operations such as database management or network communications.

The combination of the iSBC 186/78 subsystem and iPLP 720 interpreter meets all requirements of ANSI's NAPLPS Standard Reference Model (SRM).

iSBX™ 275 Graphics Controller Support

In single-user applications or where graphics activities are not the major focus of the system, the iSBX 275 MULTIMODULE shares the CPU on the host processor board through the iSBX Expansion Bus. The subsystem formed in this manner supports either monochrome or eight colors and is a very cost-

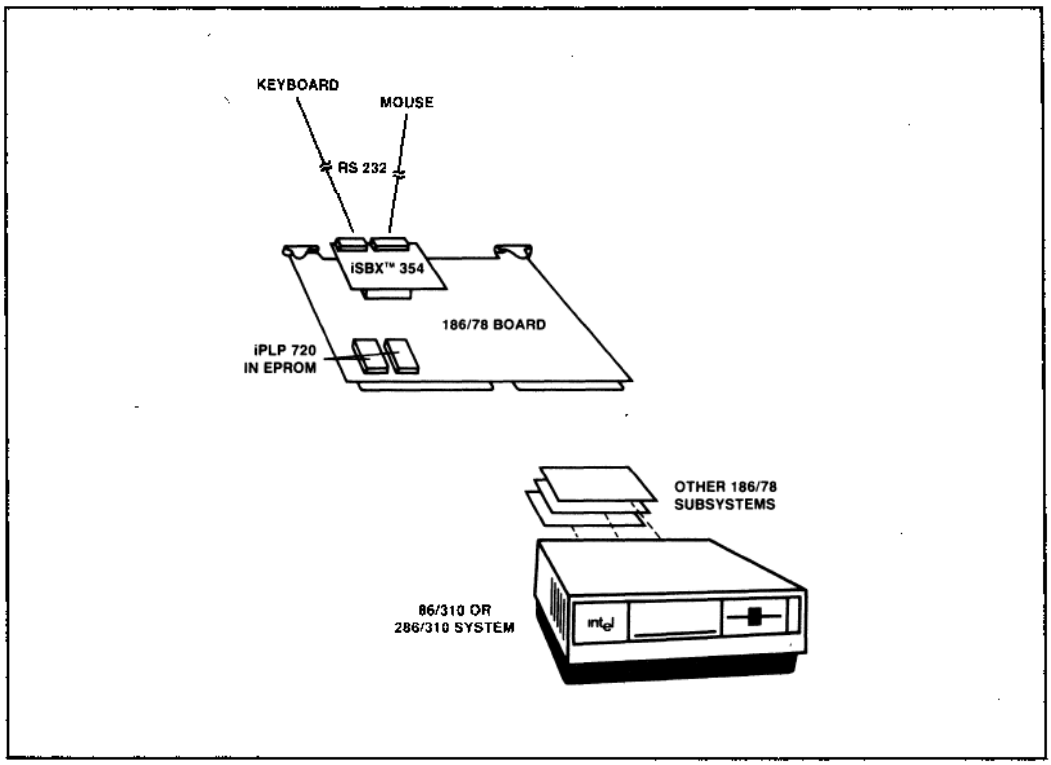


Figure 1. Multi-User Example

Order Number: 280001-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

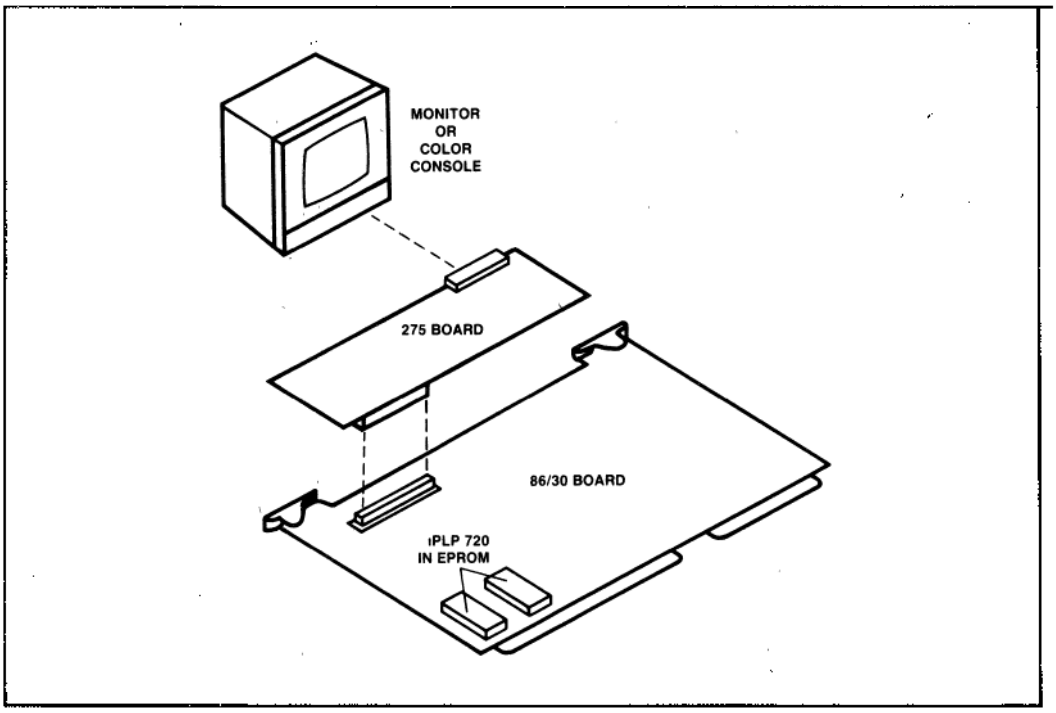


Figure 2. Single-User Example

effective solution. Like the iSBC 186/78 subsystem, this expansion module is based on the Intel 82720 component.

82720 Component Designs

The Intel 82720 GDC is an intelligent graphics controller component designed to operate as the heart of a raster-scan computer graphics display system. The 82720 performs all the basic timing needed to generate the raster display and manage the display memory. In addition, it supports several high-level graphics figure drawing functions. The Intel 82720 is an alternative to the NEC 7220 component. Custom hardware designs based on either component are supported by iPLP 720 running in an iRMX 86 environment.

NAPLPS COMMAND LIBRARY

In addition to providing driver support for Intel's growing family of graphics products, the iPLP 720 NAPLPS Interpreter decodes a wealth of high level commands to streamline the development of application code for videotex environments.

The NAPLPS standard provides character set encodings of high level text and graphics commands and capabilities. The iPLP 720 NAPLPS Interpreter implements every one of these character encodings.

The following table lists the major high level commands which are encoded in the NAPLPS standard:

Table 1. iPLP 720 Command Library

Geometric Drawing Primitives:	
Point	Incremental Point
Line	Incremental Line
Polygon	Incremental Polygon
Arc	Rectangle
Text:	
Character Rotation	Cursor Styles
Character Path Movement	Word or Character Wrap Around
Inter-character Spacing	Inter-row Spacing
Character Field Dimensions	Scrolling
Texture:	
Line Texture	
Highlighting	
Programmable Texture Patterns	
Texture Mask Size	
Miscellaneous Functions:	
Logical Pel Size	Set Color
Macro Definitions	Mosaic Sets
Blink Processes	Wait Intervals
Dynamically Redefinable Character Sets	

Order Number: 280001-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Other features are configurable as defined in the standard, however they are typically device dependent and therefore reflect the users application. Consequently, the standards document should be consulted to assure compatibility.

ANSI X3L2 NAPLPS Specification

The development of any standard is an evolutionary process. Consequently, the specification used as a reference model for the design of iPLP 720 was published in October 1982. Updates to iPLP 720 will render it fully compatible with the final specification. The American National Standards Institute (ANSI) administers the standard specification and makes it available to all interested parties. Requests for copies should be directed to:

X3 Secretariat
Computer Business Equipment Manufacturers Association (CBEMA)
311 First Street, NW
Washington, D.C. 20001

Reference document: BSR X3.110

iPLP 720 Specifications

Code size — 90 Kbytes (four 27256 EPROMs) in distributed mode;
— 72 Kbytes in stand-alone mode

Source-code language — PL/M 86

Related Literature

Reference material may be ordered from any Intel sales representative, distributor office, or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, Calif., 95051.

146144 — iPLP 720 Software Reference Manual

210506 — iSBX™ 275 Video Graphics Controller Data Sheet

231035 — iSBC® 186/78 Video Graphics Subsystem Data Sheet

146666-001 — 186/78 Reference Manual.

210655 — 82720 GDC Component Data Sheet

9803126 — iRMX™ 86 Configuration Guide

145412 — Intel's Guide to Understanding the ANSI Videotex/Teletex Standard

ORDERING INFORMATION

Intel makes available a variety of licenses to the iPLP 720 NAPLPS interpreter that allow different plans for incorporation of the Intel software into the final product. The Intel Master Software Agreement should be consulted to determine which is best suited for the particular application and production environment.

The iPLP 720 NAPLPS Interpreter comes in two formats as shown below, along with yearly update services. The iRMX 86 Real-time Multitasking Operating Systems are available separately.

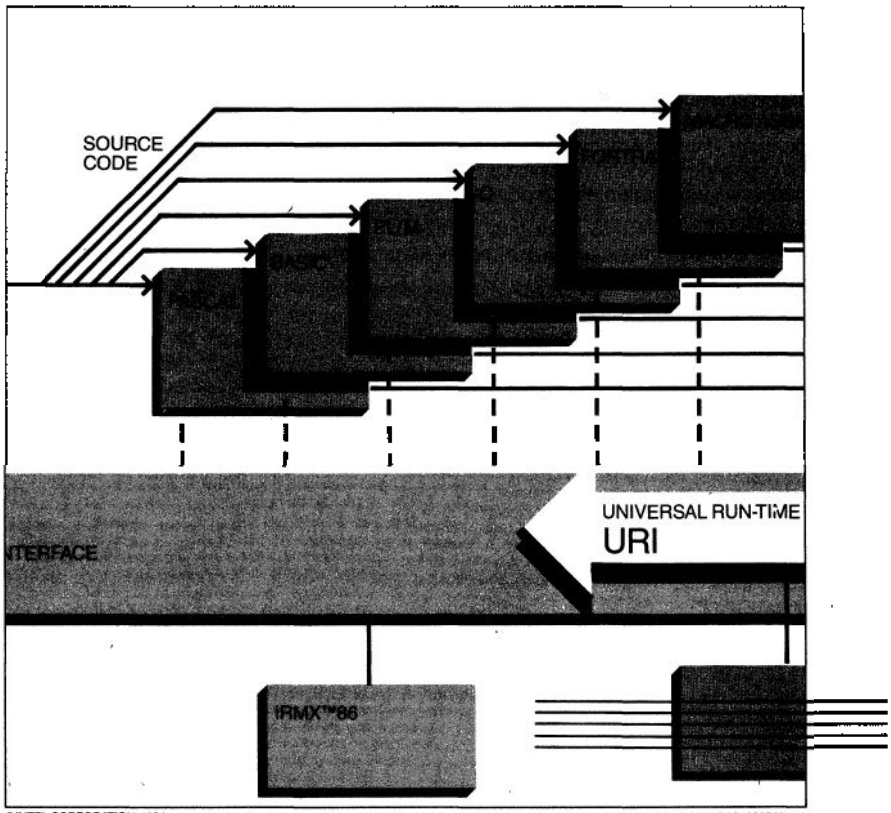
iPLP 720RO	OEM License, single density (both ISIS and iRMX formats are included)
iPLP 720RF	Incorporation fee payment
iPLP 720WX	Object code update

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

intel

iRMX™ LANGUAGE

- Industry-standard languages and utilities for developing applications on iRMX-based systems. Includes FORTRAN, Pascal, C, BASIC, PL/M, assembler, text editor
- Complete set of utilities to create and manage object modules
- Mix languages on single application system with UDI standard
- Intel 8087 and 80287 math coprocessor support
- 8086 and 80286 compatibility
- Worldwide post-sales service and support organization



©INTEL CORPORATION, 1984
*XENIX is a trademark of Microsoft Corporation
†UNIX is a trademark of Bell Laboratories

ORDER NUMBER 230749-003

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Full Language Support for iRMX™-Based Systems

Intel's iRMX™ 86 based systems are completely supported by a wide variety of popular languages and utilities with which to build fast, real-time, multi-tasking applications. Included are the latest versions of FORTRAN, Pascal, BASIC, C, PL/M and Assembler for Intel's iAPX 86 and iAPX 286 processors. Previously developed applications using any of these languages port easily to iRMX-based systems with minimal source code modifications.

In addition to the wealth of languages available, iRMX-based systems are complemented by utilities with which to create and manage object modules. This latitude in configurability allows programmers to team their efforts in order to achieve a shorter development time than would otherwise be possible.

Because the high-level languages are actually resident on the iRMX-based system, OEMs can pass application software directly on to end users. End users may then tailor the OEM's system to better meet application needs by writing programs using the same languages.

Language-Independent Application Development

Intel's Universal Development Interface (UDI) and Object Module Format (OMF) enable several users to write different modules of an application, in different languages, then link them together.

The OMF provides users with the ability to mix languages on a single application system, affording the luxury of choosing exactly the right language tools for specific pieces of the application, rather than compromising specialized tasks for the sake of one, project-wide language.

iRMX languages are fully compatible with the Intel Series III and Series IV Development Systems, should the user choose to develop applications on a specialized development system. Applications are easily moved to the final target system for test, debug and minor redevelopment.

Fast, Lean Programs for Rapid Processing

The iRMX language products enable programmers to write the smallest, fastest programs available in high-level languages, due to the compiler's superior ability to optimize code.

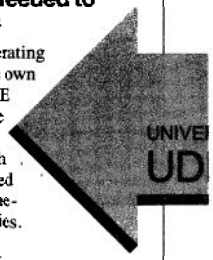
It is also possible to make iRMX operating system calls directly from FORTRAN, PASCAL and PL/M. This means that application developers can take full advantage of the iRMX multi-tasking capability, whereby multiple applications execute concurrently on the operating system. Multi-tasking, a requirement of most real-time systems, is sometimes as necessary in application software development as in an operating system environment.

Standardized REALMATH Support

All the iRMX languages (except BASIC and C) support the REALMATH floating point standard. This ensures universal consistency in numeric computation results and enables the user to take advantage of the Intel iAPX 86/20 and iAPX 88/20 Numeric Data Processor or iSBC® 337 MULTIMODULE™ boards, which boost performance two to four times over that possible on a mini-computer.

All the Utilities Needed to Link Languages

Utilities for iRMX operating systems include Intel's own EDIT, LINK, LOCATE and LIBRARIAN. The iRMX EDIT program meets the needs of both novice and sophisticated users with powerful line-oriented editing facilities.

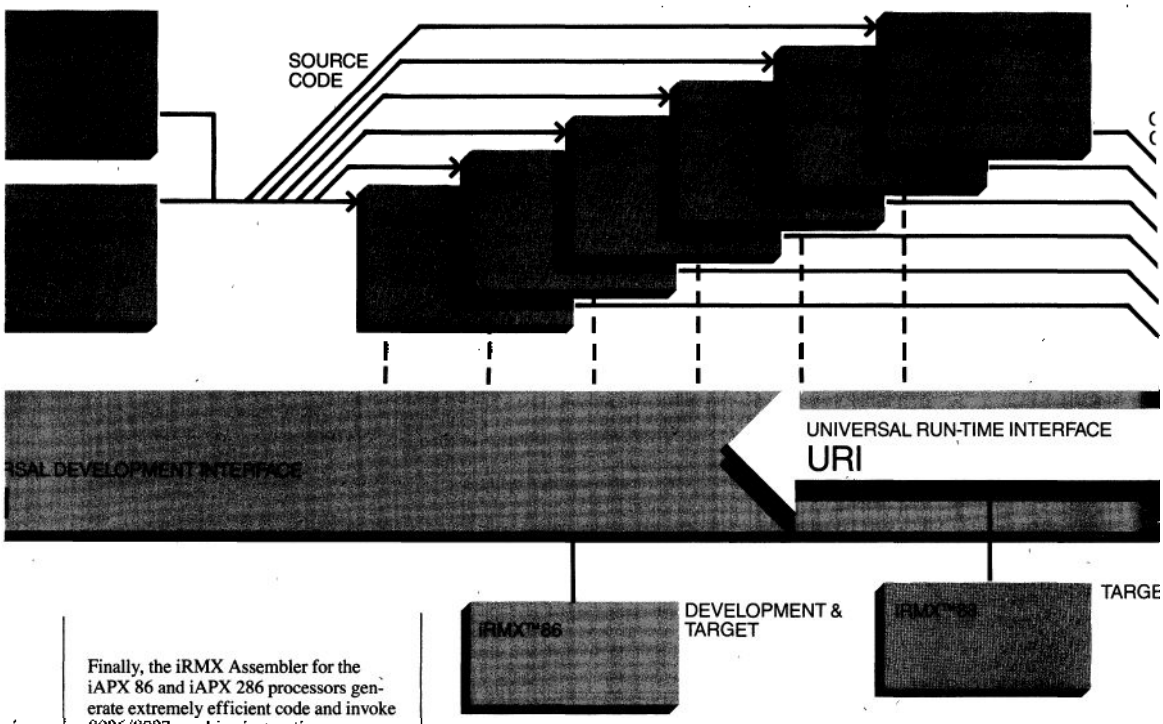


Using the iRMX LINK program, users may link individually compiled object modules to form a single, relocatable object module. This provides the ability to merge work from several programmers into one cohesive application system.

The iRMX LOCATE utility maps relocatable object code into the processor memory segments, allowing user definition of module/memory type allocation. For example, often-used portions of an application may be mapped to (P)ROM.

The LIBRARIAN object code library manager affords easy creation, collection and maintenance of related object code to reduce the overhead of separately maintained modules.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com



Finally, the iRMX Assembler for the iAPX 86 and iAPX 286 processors generate extremely efficient code and invoke 8086/8087 machine instructions.

iRMX™ 86 Pascal

iRMX Pascal meets the proposed ISO language standard and implements several microcomputer extensions. A compile-time option checks conformance to the standard, making it easy to write uniform code. Industry-standard specifications contribute to portability of application programs and provide greater reliability.

iRMX 86 Pascal supports extensions, such as an interrupt-handler and direct port I/O extension, that allow programs to be written specifically for microcomputers. Separate module compilation allows linkage of Pascal modules with modules written in other high-level languages.

For more information on iRMX 86 Pascal see the Pascal 86 Software Package data sheet.

iRMX™ 86 FORTRAN

The iRMX 86 FORTRAN compiler provides total compatibility with FORTRAN 66 language standards, plus most new features provided by the FORTRAN 77 language standard. iRMX 86 FORTRAN includes extensions specifically for microcomputer application development. Programming is simplified by relocatable object libraries, which provide run-time support for execution time activities.

iRMX 86 FORTRAN supports the 8087 math coprocessor for the most powerful

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

microcomputer solution available in number-intensive applications. For more information on iRMX 86 FORTRAN see the FORTRAN 86 Software Package data sheet.

iRMX™ 86 PL/M

PL/M offers full access to micro-computer architecture while simultaneously offering all the benefits of a high-level language. Invented by Intel in 1976, PL/M 80 was the first microcomputer-specific, block-structured, high-level language available. Since then, thousands of users have generated code for millions of microcomputer-based systems using PL/M 80 and PL/M 86.

Software written for 8-bit processors (PL/M 80) are easily ported to the more powerful 16-bit (PL/M 86) environment. The same portability will be available for future VLSI.

For more information about iRMX 86 PL/M see the PL/M 86/88 Software Package data sheet.

iRMX™ 86 BASIC

Intel's offering of Microsoft BASIC is a standardized version of the most popular high-level language in the world. Existing BASIC programs are easily ported to iRMX-based systems. BASIC is an excellent pass-through language by which an OEM can offer customers the ability to write and modify their own applications.

iRMX™ 86 C Compiler

The popular new programming language, C (Mark William's Company version), is fully supported on iRMX-based systems. iRMX 86 C offers both small and large segmentation models, enabling applications to be written efficiently. The iRMX 86 C compiler combines assembly language efficiency with high-level language convenience; it can manipulate on a machine-address level while maintaining the power and speed of a structured language.

The iRMX 86 C compiler affords easy portability of existing C programs to iRMX-based systems. For more information on the iRMX C compiler see the iRMX 86 C Software Package data sheet.

iRMX™ 86 Text Editor

The iRMX 86 Text Editor is screen-oriented, menu-driven and easy to learn. Guided by the menu of commands always before him, the user can edit text and programs easily and efficiently.

iRMX 86 Text Editor allows the simultaneous edit of two files. This allows easy transferral of text between files and use of existing material in the creation of new files. Creating macros, strings of frequently-used commands, is also very simple. The editor "remembers" the selected commands and allows the user to re-use them repeatedly.

Worldwide Service and Support

All iRMX systems are completely supported by Intel's worldwide staff of trained hardware and software engineers. iRMX Language customers receive a warranty that includes Hotline Support, Software Updates, and Subscription Service.

Complete documentation is provided for all operating system and application software languages, as well as for system hardware components. An Intel system is not a collection of hardware and software pieces as much as a cohesive whole that is supported and serviced as such.

Intel Has Total Solutions for Real-Time Systems

iRMX 86 is the fastest, most powerful operating system available for multi-tasking, multi-user, real-time applications. Complemented by a wide range of industry-standard languages and utilities, the iRMX-based systems are highly flexible and configurable.

Application development for iRMX-based systems is possible at the board or the system level. OEMs can integrate functionality at the most profitable level of product design, using one system for both development and target use. Intel's choice of industry standard high-level languages enables the end user to extend OEM-provided functionality even further, if desired.

Who is better qualified to write and supply software for Intel VLSI than Intel? Today you have the ability to tap into hundreds of available application software packages, languages and utilities, peripherals and controllers and MULTIBUS® boards.

Tomorrow, and ten years down the road, you will be able to tap into the latest, high-performance VLSI — without losing today's software investment.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Specifications

Required Hardware	Required Software	
• Any iAPX 86/286 based or iSBC 86/286 based system including Intel's System 86/300 and 286/300 family. In addition, object code from the compilers will run on iAPX 88 based systems. • 140KB of memory • Two iRMX 86 compatible floppy disks or one hard disk • One 8″ double density or 5.25″ double-density floppy disk drive for distribution of software • System console device	The iRMX 86 Operating System Release 6 or later including the nucleus, basic I/O system, extended I/O system and human interface Purchase of any RMX language requires signing of Intel's OEM License Agreement (OLA).	

Ordering Information

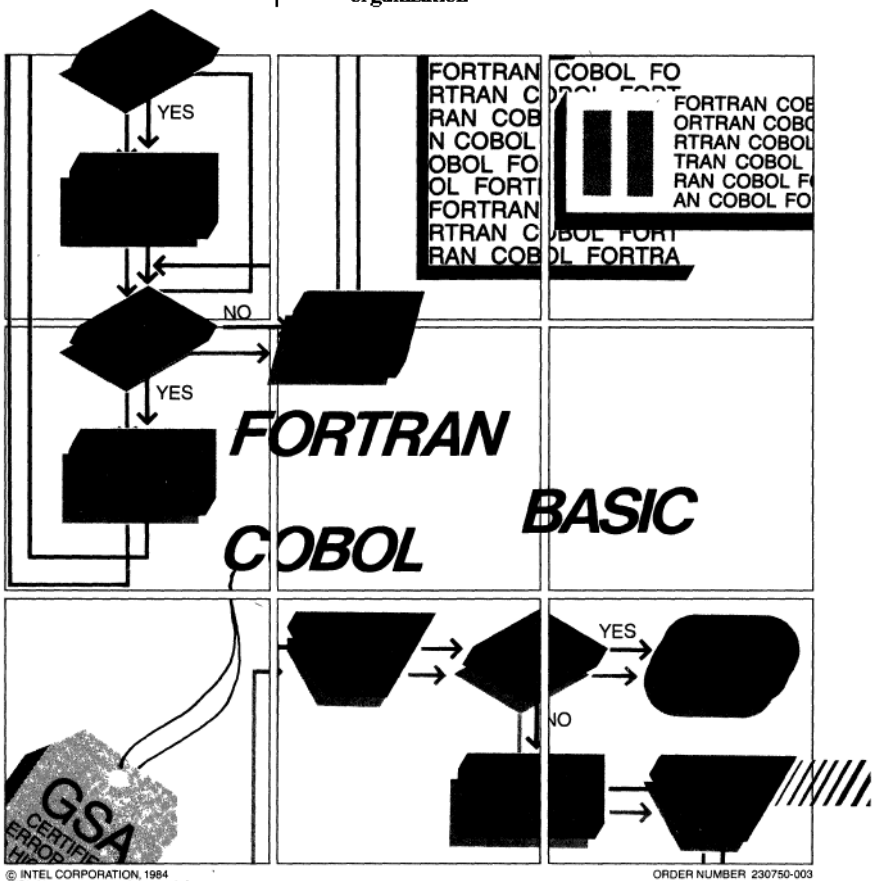
Language	Order Code	Product Contents	Warranty
ASM 86, Utilities	RMX 860	Two 8″ disk and two 5.25″ diskettes Edit Reference Manual iAPX 86/88 Family Utilities User's Guide Macro Assembler Operating Instructions ASM 86 Language Reference Manual 8087 Support Library Reference Manual	**90 days:** Software Updates, Subscription Service, Hotline Support
Pascal	RMX 861	Two 8″ diskettes and two 5.25″ diskettes Pascal 86 User's Guide	**90 days:** Software Updates, Subscription Service, Hotline Support
FORTRAN	RMX 862	Two 8″ diskettes and two 5.25″ diskettes FORTRAN 86 User's Guide	**90 days:** Software Updates, Subscription Service, Hotline Support
PL/M	RMX 863	One 8″ diskette and one 5.25″ diskette PL/M 86 User's Guide	**90 days:** Software Updates, Subscription Service, Hotline Support
TX Editor	RMX 864	One 8″ diskette and one 5.25″ diskette TX Screen Echter User's Guide	**90 days:** Software Updates, Subscription Service
BASIC	RMX 865	One 8″ diskette and one 5.25″ diskette BASIC Reference Manual BASIC 86 User's Guide One 8″ diskette and one 5.25″ diskette	**90 days:** Software Updates, Subscription Service
C	RMX 866	One 8″ diskette and one 5.25″ diskette C *Programming Language* by Kernighan and Ritchie (Prentice-Hall) C 86 Compiler User's Guide	**90 days:** Software Updates, Subscription Service, Hotline Support

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

intel

XENIX* LANGUAGES

- COBOL, BASIC and FORTRAN support for Xenix-based systems
- Conformation to international standards: ANSI 77 subset FORTRAN, ANSI X3.23 1974 COBOL to Federal High Level and ANSI X3.60—1978 subset BASIC
- Powerful microcomputer extensions to ANSI standards
- Easy porting of mainframe and minicomputer applications to micro environment
- Intel 80287 math coprocessor support
- Worldwide service and support organization



© INTEL CORPORATION, 1984
*XENIX is a trademark of Microsoft Corporation

ORDER NUMBER 230750-003

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

High-level Language Support for XENIX-Based Systems

Intel's XENIX operating system, available for component, board, or system-level integration, is a multi-user operating system well suited for both technical and commercial interactive applications. Typical applications include small business systems, software development/engineering workstations, distributed data processing and graphics.

For OEM and end-user application development on XENIX, Intel has provided three industry-standard, high-level languages — FORTRAN, COBOL and BASIC — with which to build microcomputer-based solutions for systems products or component and board-level applications. XENIX BASIC, FORTRAN and COBOL accommodate easy porting of existing mainframe and mini-based applications to the micro environment.

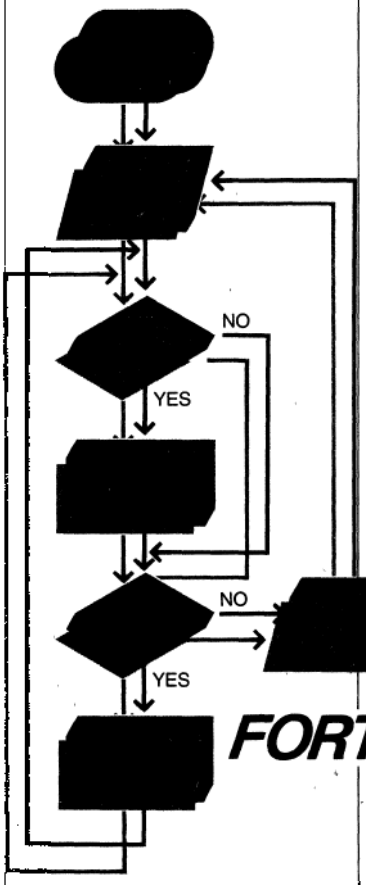
XENIX FORTRAN for Scientific and Technical Applications

FORTRAN is the most popular programming language for scientific and numerical applications. There are thousands of existing FORTRAN programs and subroutines written in mainframe and minicomputer environments, most of which can be ported to a micro environment via Intel's offering of Microsoft FORTRAN.

Compliance with the X3.9 1978 ANSI standard for FORTRAN at the subset level ensures portability with minimal source code modifications. By moving to a microcomputer-based system, you lose none of your mainframe and mini-developed software investment.

Speed and Accuracy Where They're Needed

Scientific, math-oriented applications usually require fast, highly accurate processing. XENIX FORTRAN delivers accuracy with double-precision arithmetic





which handles numbers containing 15 significant digits.

High speed results from XENIX FORTRAN support of the Intel 80287 floating point coprocessor, as well as from an extensive subroutine library, which includes subroutines for 16- and 32-bit integer arithmetic and 32- and 64-bit floating-point arithmetic. Because of XENIX FORTRAN's 80287 math coprocessor support, some programs written in XENIX FORTRAN will execute from two to four times faster than their minicomputer counterparts.

Calls to "C" and MS MACRO Assembler are possible, making it easy to interface non-standard peripherals to XENIX FORTRAN programs.

FORTRAN

XENIX COBOL for the Micro Environment

Intel's offering of Microfocus COBOL is a mainframe-caliber compiler for ANSI 1974 COBOL programs, enabling XENIX-based systems to compile and run existing COBOL programs with minimal source code modification. XENIX COBOL also contains features specifically aimed at facilitating the interactive

*XENIX is a trademark of Microsoft Corporation

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

BASIC

RTRAN COBOL F
TRAN COBOL FO
RAN COBOL FOR
AN COBOL FORT
N COBOL FORTR
COBOL FORTRA

program development of new applications in a microcomputer environment.

These features include a facility for dynamically loading sub-programs from disk as required which effectively removes limits on the size of the application code that can be run. XENIX COBOL augments the functionality of the ANSI standard with additional compiler features, such as interactive screen-handling, that further increase convenience and programmer productivity.

Users can license a separate run-time support package. This enables OEMs to pass COBOL applications onto customers at a much lower cost than that involved in transferring full COBOL packages.

XENIX COBOL is one of only eleven COBOL compilers in existence—and the only one for microcomputers—that has been GSA-certified as error-free at the High Level. A special ANSI-defined communications module provides the user with a standard mechanism for program-to-program message-passing in multi-user networks such as those found in an "office of the future" settings.



Forms-2™ Support for Screen-Painting

XENIX COBOL supports FORMS-2, a powerful visual programming tool that speeds the creation of programs involving interactive screen-handling. In an extremely user-friendly environment, the user "paints" a form on the screen, and FORMS-2 generates the COBOL source code to support it. FORMS-2 results in greatly improved programmer productivity in a microcomputer, screen-building environment.

XENIX BASIC for Maximum Flexibility

Intel's offering of Microsoft BASIC opens a whole window of applications to the XENIX user. Since their BASIC is the same as that used on MS-DOS* based machines, most programs written for MS-DOS can now run on XENIX unchanged. When developing your own programs, BASIC is simple and easy for quick prototyping, yet complete enough for total development. Conforming to the ANSI X3.60 1978 subset standard, BASIC also has powerful extensions, 16 significant digit Double Precision floating point arithmetic, 80287 support, and assembly languages routine calling capabilities. From using applications to designing your own programs BASIC is easy, complete, and extremely flexible.

COBOL

Worldwide Service and Support

All XENIX systems are fully supported by Intel's worldwide staff of trained hardware and software engineers. Complete documentation is provided for all operating systems and application software languages, as well as for system hardware components. The XENIX and XENIX Languages warranty includes Hotline support, Software Updates, and Subscription Service.

Total Solutions for Interactive, Multi-User Applications

Intel's XENIX-based systems offer the most complete solutions for interactive, multi-user applications requiring fast, accurate throughput and a friendly programming environment. XENIX is complemented by industry-standard, high-level languages with which OEMs can create flexible and open end-user systems.

XENIX languages are completely portable—from one level of integration to another (chip to board to system).

Intel is paving the way into the future of VLSI and pioneering VLSI-based systems. We are committed to providing customers with smooth, uninterrupted application development on the latest VLSI-based systems - today and tomorrow.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Specifications

Required Hardware:	Required Software:	
• Any iAPX 286 based or iSBC® 286 based system including Intel's 286/300 family and iDIS systems • 196 KB memory • Two floppy disks or one hard disk • One 8" double-density or 5.25" double-density floppy disk drive for distribution of media	• Intel's XENIX 286 Operating System • Purchase of any XENIX Language requires signing of Intel's OEM License Agreement (OLA)	

Ordering Information

Language	Order Code	Product Contents	Warranty
COBOL	XNX 2867	One 8" diskette and one 5.25" diskette Level II COBOL Language Reference Manual—122158 Level II COBOL Operating Guide—122159 Forms II Utility Manual—122160 Level II COBOL Pocket Guide—122161	**90 days:** Software Updates, Subscription Service
	XNX 2868	Incorporation Fee for passing through the COBOL Runtime System	
FORTRAN	XNX 2862	One 8" diskette and two 5.25" diskettes Fortran Reference Manual Fortran User's Guide	**90 days:** Software Updates, Subscription Service
BASIC	XNX 2865	One 8" diskette and one 5.25 diskette BASIC Reference Manual BASIC User's Guide	**90 days:** Software Updates, Subscription Service

FORMS-2 is a trademark of Micro Focus

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2920 SOFTWARE SUPPORT PACKAGE

- **Complete software design and development support for the 2920**
- **Extends Intellec® Microcomputer Development System to support 2920 software development**

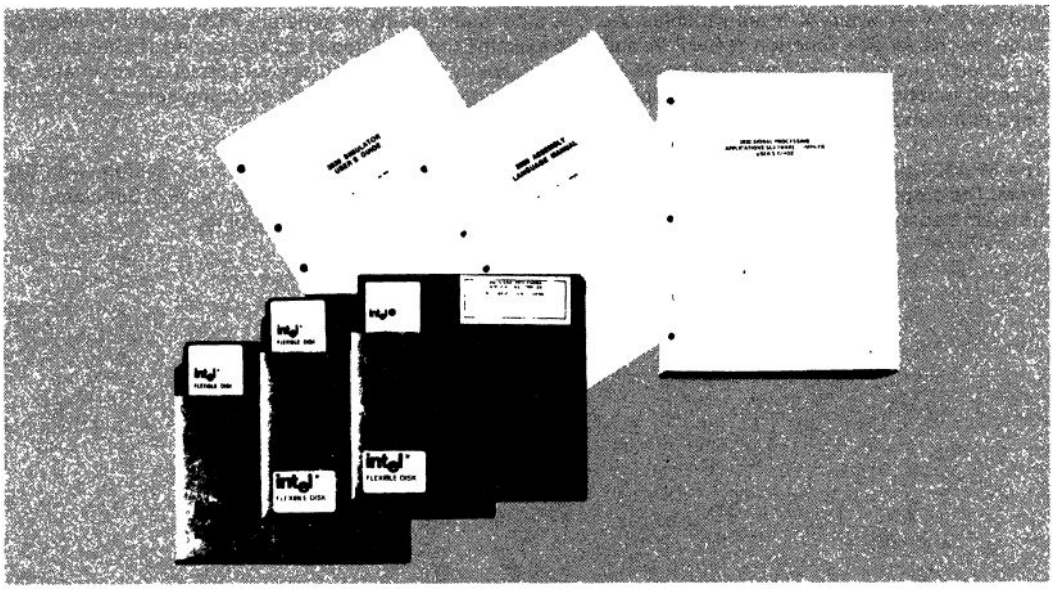
The 2920 Software Support Package furnishes a 2920 Signal Processing Applications Software/Compiler, 2920 Assembler, and 2920 Software Simulator. These three software design and development tools run on the Intellec® Microcomputer Development System.

The 2920 Signal Processing Application Software/Compiler is an interactive tool for designing software to be executed on the 2920 Signal Processor. The compiler accepts English-like statements from the user and generates 2920 assembly language code.

The assembler translates symbolic 2920 assembly language programs into the machine operation code. The user can load the code into the simulator for 2920 simulation or to the Universal PROM Programmer for 2920 EPROM programming.

The simulator, operating entirely in software, allows the user to test and symbolically debug 2920 programs. The user can specify input signals, simulate program execution, set up breakpoints, display input and output, and display and alter the contents of the 2920 registers and memory locations. The simulator can also stop or trace the program and constructively give the user access to the key elements inside a 2920 for analyzing his program.

The compiler, assembler, and simulator enable the designer to develop and test an entire program without a complete prototype design. The 2920 designer works on the Intellec® Microcomputer Development System rather than on a breadboard. The development system can program, store and recall programs or routines and aid in 2920 program design.



2920 Software Support Package

The following are trademarks of Intel Corporation and may be used only to identify Intel products: BXP, Intellec, Multibus, i, iSBC, Multimodule, ICE, iSBX, PROMPT, iCS, Library Manager, Promware, Insite, MCS, RMX, Intel, Megachassis, UPI, Intelevision, Microsmp, μScope and the combination of ICE, iCS, iSBC, iSBX, MCS, or RMX and a numerical suffix

Intel Corporation 1980

Sept 1980
1662208

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2920 SIGNAL PROCESSING APPLICATIONS SOFTWARE/COMPILER

- **Compiler generates 2920 Assembly Language Code**
- **Extensive command set for designing electrical filters**
- **Graphics capability enhances analysis of filter response or piecewise linear function approximations**
- **Powerful MACRO capability for executing frequently used routines**
- **Interactive software support tool for 2920 Signal Processor**
- **Extends Intellec® Microcomputer Development System support of the 2920**
- **Contains MACRO library for several standard filters and signal processing functions**

The 2920 Signal Processing Applications Software/Compiler (SPAS20) is an interactive tool for designing software to execute on the 2920 Signal Processor.

The SPAS20 package can be visualized as being comprised of four inter-related sections: A compiler section, a filter design section, a curve fitting section, and a MACRO section.

Among the abilities of SPAS20 are: ability to generate 2920 assembly language code directly from specifications of signal processing building blocks such as filters and waveform generators; ability to generate 2920 assembly language code for several classes of algebraic equations such as $Y = C * X$, $Y = C * Y$, and $Y = C * X + Y$ where X, Y are variables and C is a constant; ability to generate 2920 assembly language code for one variable function $Y(X) = F(X)$; ability to examine time and frequency responses of filter sections specified by continuous or sampled poles and zeroes; ability to examine piecewise linear approximation of specific function; ability for users to implement more complex commands by grouping sets of commonly used commands into a MACRO.

The SPAS20 package runs under ISIS-II on any Intellec® Microcomputer Development System with 64K RAM. The output of SPAS20 can be assembled with the 2920 assembler, tested with the 2920 Simulator, and programmed into the 2920 chip with the Universal PROM Programmer for prototyping.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FUNCTIONAL DESCRIPTION

The 2920 Signal Processing Applications Software/Compiler gives the analog designer a "high level language" for his 2920 applications—it decreases the need to code 2920 assembly language. Furthermore, the compiler is interactive. This feature enables the designer to define a filter, or transfer function, graph their response, and change their parameters many times, without having to program and test in an actual 2920 implementation.

Once a filter is realized by moving poles and zeros in the continuous and sampled planes, the filter may be coded and written onto an ISIS file. Similarly, after a function Y = F(X) has been defined, the code for a piecewise linear approximation can be stored onto an ISIS file. Several other file commands are available to store and retrieve command sequences for SPAS20 sessions.

SPAS20 Command Language

DEFINE	This command defines a pole or zero by associating it with a number (i.e., POLE 3), and with real and imaginary coordinates in the continuous or sampled plane. This command also defines a symbol by associating a name with a numeric value, or a MACRO by providing a pointer to a specified command sequence.
GRAPH/ OGRAPH	This command graphically displays the values of object(s) specified. For example, GRAPH GAIN and GRAPH PHASE are used to display filter response. The OGRAPH command will "overgraph" the new response over the old response, after any changes have been made. (You may also graph Group Delay, Step, and Impulse.)
MOVE	Allows the definition of a pole or zero to be changed—its coordinates, its plane, or both.
REMOVE	Deletes the definition of a pole, zero, symbol, or macro.
HELP	Types an explanatory message on the console, pertaining to a command or its attributes.
FIT	This command performs curve fitting, i.e. it approximates an arbitrary user supplied function with a piecewise linear function.
DATA	This command allows for specification of a set of vertices (i.e. X – Y coordinate pairs) which determine a piecewise linear approximation of some defined function, filter response characteristics, etc.
HOLD	Command to correct attenuation due to sample-and-hold distortion: if ON, it corrects absolute gain by sin(x)/x and phase by adding x, where x=TS*FREQ*π. It corrects group delay by subtracting π*TS.
EVALUATE	Gives the decimal numeric value of any expression.
CODE	Creates 2920 assembly language code for given poles, and zeros, equations, and user defined functions.

The SPAS20 compiler also recognizes the following commands for file handling:

PUT/ APPEND	Writes out objects (commands) to a specified file, either creating a new one or appending an existing one. This enables the user to store all or part of a SPAS20 session on a diskette to be brought back later with the INCLUDE command.
DISPLAY	Copies the contents of a file to the console.
INCLUDE	Executes a sequence of instructions from a diskette file as if they were typed in from the console.
LIST	Creates a file containing all console interactions.

In addition to naming macros for specific command sequences, compound and conditional commands may be formed using all of the above statements. These compound commands are:

IF	Establishes conditional flow of control within a block of commands.
REPEAT	Used for repetition of a block of commands; executes indefinitely or until a condition is met (using WHILE, UNTIL, and END statements).
COUNT	Establishes the number of times a command sequence is to be executed, in a looping fashion.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SPAS20 MACRO Facility

A macro is a sequence of commands that is stored on a temporary diskette file. The command sequence is executed when the macro name is entered as a command. This saves repetitive entry of the sequence, and permits alogorithms to be saved on diskette for future use. This SPAS20 facility allows you to do the following:

- Display the text of any macro.
- Define a macro, specifying its name and any parameters that are to be used by the block. This definition is followed by the contents of the macro (commands) and the EM statement to end its definition.
- Invoke a macro by entering its name and appropriate values for any parameters.
- List the names of all defined macros.
- Remove any or all macros.

Intel also supplies several MACRO library files containing the following commonly needed MACROs:

- Filter design MACROS
  - — Butterworth filter
  - — Chebyshev filter
  - — Bilinear transform
  - — Evaluate gain or phase of digital filter in parallel form
  - — Time response simulation
- Function design MACROs
  - — Code and error optimization
  - — Calculate instertitial error
- MACROs for generation of 2920 code
  - — Code for all-POLE filter
  - — Input and A/D conversion
  - — Multiplication
  - — Division
  - — Logarithm functions
  - — Square-root functions
  - — Sinewave oscillator

SAMPLE SPAS20 FILTER DESIGN SESSION

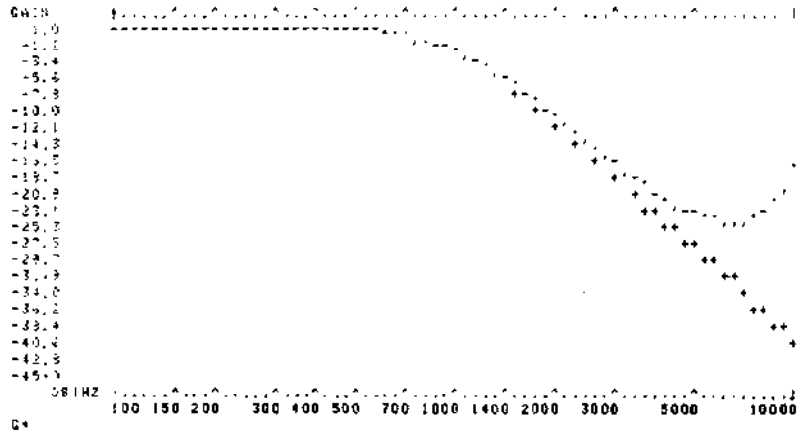
```
-:FI : SPAS20 . SPT
ISIS-II 2920 SIGNAL PROCESSING APPLICATIONS COMPILER, V2.0
*
*DEFINE POLE 1 = -.707,.707        ; CREATE A POLE IN CONTINUOUS S-PLANE
*
*P2          LIST ALL POLES AND ZEROS
POLE 1 = -707 00000,707 00000,CONTINUOUS
*
*FSCALE = 100,10000        ; ESTABLISHES FREQUENCY RANGE OF INTEREST
*
*YSCALE = -45.1        ; ESTABLISHES MAGNITUDE RESPONSE RANGE OF INTEREST
*
*GRAPH GAIN        ; PLOT MAGNITUDE RESPONSE OF POLE PAIR

GAIN
  1.0
 -1.2
 -3.4
 -5.6
 -7.8
-10.0
-12.1
-14.3
-16.5
-18.7
-20.9
-23.1
-25.3
-27.5
-29.7
-31.9
-34.0
-36.2
-38.4
-40.6
-42.8
-45.0
    DB:HZ
          100 150 200   300 400 500   700 1000 1400 2000  3000      5000      10000
*<
*
*, THE UNITS USED IN GRAPHING GAIN ARE SHOWN IN THE LOWER LEFT CORNER
*: GAIN IN DECIBELS IS GRAPHED VERSES FREQUENCY IN HERTZ
*
*! PREPARE TO MOVE TO THE DIGITAL DOMAIN.
*: SAMPLE RATE MUST BE SPECIFIED
*
*TS = 1/13020        ; RATE FOR 192 INSTRUCTION PROGRAM AND 10MHZ CLOCK
TS = 7.6805004/10**5
*
```

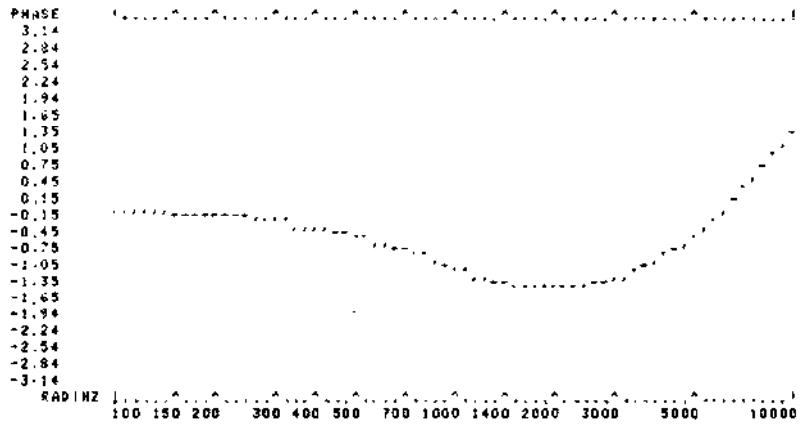

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SAMPLE SPAS20 FILTER DESIGN SESSION (Cont'd.)

```
*MOVE POLE TO Z        ; CONVERT FILTER TO DIGITAL VIA MATCHED-Z TRANSFORMATION
1 POLES/ZEROES MOVED
*
*P         ; LIST TRANSFORMED POLE
POLE 1 = 0 71092836.0 34110369.2
*
*; COMPARE RESPONSES OF THE ANALOG AND DIGITAL FILTERS BY GRAPHING THE
*; NEW RESPONSE OVER THE OLD
*
*OGRAPH GAIN
```




```
G*
*
*; PLUS SIGNS INDICATE OLD CURVE
*; NOTE THAT THE DIGITAL FILTER RESPONSE BEGINS TO INCREASE AGAIN
*; AT HALF THE SAMPLE RATE ( 6510 HZ )
*
*; THE PHASE CHARACTERISTICS OF THIS FILTER CAN BE EXAMINED
*
*YSCALE = -PI,PI        ; ESTABLISHES RANGE OF INTEREST
*
*GRAPH PHASE
```




```
P*
*
*PUT :F1:POLE PZ        ; SAVE THE POLE LOCATION IN A DISK FILE BACKUP
*
*MODE POLE 1 INST(1)        ; GENERATE 2920 ASSEMBLY CODE FOR THIS FILTER
B,=! 33989990  B2=-0 50541914
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SAMPLE SPAS20 FILTER DESIGN SESSION (Cont'd.)

```
        OPTIMIZED 2920 CODE IS NOW GENERATED  TO SAVE SPACE, SOME
     OF THE SCREEN OUTPUT HAS BEEN DELETED  NORMALLY ALL ATTEMPTS
     BY THE COMPILER TO GENERATE CODE ARE ECHOED ON THE SCREEN  )

INST=10
 POLE 1 = 0.71089438.0 34116779.Z
  BEST: PERROR = 3 3795874/10**5.1 58846567/10**5

; NOTE: MAKE SURE SIGNAL IS <0 74635371
LDA OUT2_P1,OUT1_P1,R00
    ; OUT2_P1=1 00000000*OUT1_P1
LDA OUT1_P1,OUT0_P1,R00
    ; OUT1_P1=1 00000000*OUT0_P1
SUB OUT0_P1,OUT1_P1,R05
    ; OUT0_P1=1 00000000*OUT0_P1-0 031250000*OUT1_P1
ADD OUT0_P1,OUT0_P1,R03
    ; OUT0_P1=1 12500000*OUT0_P1-0 03315625D*OUT1_P1
ADD OUT0_P1,OUT1_P1,R02
    ; OUT0_P1=1 12500000*OUT0_P1+0 21484375*OUT1_P1
SUB OUT0_P1,OUT2_P1,R01
    ; OUT0_P1=1 12500000*OUT0_P1+0 21484375*OUT1_P1-0 50000000*OUT2_P1
SUB OUT0_P1,OUT2_P1,R08
    ; OUT0_P1=1 12500000*OUT0_P1+0 21484375*OUT1_P1-0 50390625*OUT2_P1
ADD OUT0_P1,OUT2_P1,R11
    ; OUT0_P1=1 12500000*OUT0_P1+0 21484375*OUT1_P1-0 50341796*OUT2_P1
SUB OUT0_P1,OUT2_P1,R09
    ; OUT0_P1=1 12500000*OUT0_P1+0 21484375*OUT1_P1-0 50537109*OUT2_P1
ADD OUT0_P1,IN0_P1,R00
    ; OUT0_P1=1 12500000*OUT0_P1+0 21484375*OUT1_P1-0 50537109*OUT2_P1+1 00000000*IN0_P1
*
*; THE CODE COMMAND SPECIFIED THAT THE POLE PAIR BE CODED IN LESS THAN 11
*; INSTRUCTIONS, SO 10 INSTRUCTIONS WERE GENERATED, WITH COMMENTS
*; THE FINAL ERROR IN RADIUS AND ANGLE FOR THE POLE PAIR WAS OF THE
*; ORDER OF 1/10**5 AS INDICATED ABOVE IN PERROR
*; THIS OPTIMIZED 2920 ASSEMBLY CODE CAN NOW BE APPENDED TO A FILE
*; WHICH MAY CONTAIN OTHER CODED FUNCTIONAL BLOCKS OF A 2920 PROGRAM
*
*EXIT
```

SAMPLE SPAS20 CURVE FITTING SESSION

```
-;        DEMONSTRATION OF THE SPAS20 CURVE-FITTING PACKAGE
-
-
-SPAS20.SFT

ISIS-II 2920 SIGNAL PROCESSING APPLICATIONS SOFTWARE/COMPILER, V2.0
*LIST XCURFD.829
*
*
*;  THE CURVE FITTING COMMANDS IN SPAS20 WILL GENERATE 2920 CODE TO CALCULATE
*;  SOME FUNCTION SUCH AS  Y**3.   Y**3 COULD BE COMPUTED ON THE 2920 CHIP
*;  WITH TWO MULTIPLIES USING ABOUT 10 INSTRUCTIONS AND THE DAR.  HOWEVER IT
*;  WOULD TIE UP THE DAR TOO LONG.  THE CODE GENERATED BY THE CURVE FITTING
*;  COMMANDS DOES NOT USE THE DAR.
*
*CODE FIT YSCURFD(X) = Y**3 PRROR<.05   ;ERROR BOUND OF .05
*
*CODE      ; HERE IS THE CODE GENERATED.
LDA TEMP,X,R00
    ; TEMP=1.00000000*X
LDA YCURFD,X,R01
    ; YCURFD=0.50000000*X
ADD YCURFD,X,R06
    ; YCURFD=0.51562500*X
ADD TEMP,X,R01
    ; TEMP=0.50000000*X+1.00000000*TEMP
ADD YCURFD,TEMP,R05
    ; YCURFD=1.00000000*YCURFD+0.03125000*TEMP
SUB YCURFD,TEMP,R02
    ; YCURFD=1.00000000*YCURFD-0.21875000*TEMP
ADD TEMP,X,R00
    ; TEMP=1.00000000*X+1.00000000*TEMP
ADD YCURFD,TEMP,R03
    ; YCURFD=1.00000000*YCURFD+0.00390625D0*TEMP
SUB YCURFD,TEMP,R04
    ; YCURFD=1.00000000*YCURFD-0.05859375D0*TEMP
LDA YCURFD,YCURFD,L02
    ; YCURFD=4.00000000*YCURFD-3.23437500*TEMP
  *
  *EXIT      ; THE FUNCTION WAS CODED IN THIS MANY INSTRUCTIONS;
  INST = 10.00000000
  *
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
*ERROR    ; THE CODE APPROXIMATES X**3 WITHIN THIS ERROR;
ERROR = 0.046875000
*
*DATA 0 THRU 1   ;EXAMINE THE PIECEWISE LINEAR FUNCTIONS VERTICLS.
DATA 0.00000000 THRU 1.00000000 = 0.00000000 AT 0.00000000,&
0.06562501 AT 0.40000000,&
0.26562500 AT 0.66666669,&
0.95312500 AT 1.00000000
*
*GRAPH DATA(X)   ;THE DATA ARRAY APPROXIMATES THE FUNCTION AND CAN BE GRAPHED.
FUNCTION  '------^------^-----^------^------^------^------^-----^------^------!
0.95                                                                      .'
0.91                                                                    .'
0.86                                                                  .'
0.82                                                                .'
0.77                                                              .'
0.73                                                            .'
0.68                                                          .'
0.64                                                        .'
0.59                                                      .'
0.54                                                    .'
0.50                                                  .'
0.45                                                .'
0.41                                              .'
0.36                                            .'
0.32                                          .'
0.27                                        .'
0.23                                      .'
0.18                                    .'
0.14                                  .'
0.09                                .'
0.05                              .'
0.00      ....------''''''''''''''
          '------^------^-----^------^------^------^------^-----^------^------'
*         0.00 0.1    0.2    0.3    0.4    0.5    0.6    0.7    0.8    0.9    1.00
*GRAPH X**3   ; THE DIFFERANCE BETWEEN THE CODED AND THE ACTUAL APPEARS AS "+".
FUNCTION  '------^------^-----^------^------^------^------^-----^------^------!
1.00                                                                      +'
0.95                                                                    +'
0.90                                                                  +.'
0.86                                                                +.
0.81                                                              ++-
0.76                                                            +.-
0.71                                                          +.'
0.67                                                        ++.
0.62                                                      +.-
0.57                                                    ++.'
0.52                                                  +.-
0.48                                                ++-
0.43                                              +.'
0.38                                            +.'
0.33                                          .-'
0.29                                        +.-+
0.24                                      .'
0.19                                    ++.-'
0.14                                  ++.--'
0.10                                +.--'
0.05                        +++++++..--'
0.00      +++++++++++++++------'''
          '------^------^-----^------^------^------^------^-----^------^------!
*         0.00 0.1    0.2    0.3    0.4    0.5    0.6    0.7    0.8    0.9    1.00
*GRA (X**3)-DATA(X)   ; THE ERROR WILL BE GRAPHED.
FUNCTION  '------^------^-----^------^------^------^------^-----^------^------!
 0.047
 0.043
 0.039
 0.036
 0.032
 0.028                                                                  -
 0.025                                                                -
 0.021                                                              .'
 0.017                                                            .'
 0.014                                                          -'
 0.010                                                        .'
 0.006                                                      -'
 0.003                                                    .'
-0.001    '.                                          .'
-0.005      -                                       .'
-0.008       '.                                   .'
-0.012         -                                .'
-0.016          '.                            .'
-0.020            '.                        .'
-0.023              '-.                   .-'
-0.027                 '---.         .---'
-0.031                      '''-----'''
          '------^------^-----^------^------^------^------^-----^------^------1
*         0.00 0.1    0.2    0.3    0.4    0.5    0.6    0.7    0.8    0.9    1.00
*EXIT   ; THAT'S ALL FOLKS
-
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2920 ASSEMBLER

2920 program development on Intellec® Microcomputer Development Systems

Translates symbolic assembly language instructions into 2920 machine code

Produces Assembly Listing, Object Code File, and Error Diagnostics

Output used for 2920 programming with the Intellec PROM Programmer or the 2920 Simulator for program debug

The 2920 Assembler translates symbolic 2920 Assembly Language instructions into the appropriate machine operation codes. Through this facility, the programmer is able to symbolically program 2920 hardware operations. Compared to machine code, these symbolic references provide faster programming, easier debugging, and greater reliability.

The Assembler produces an object code file (executable machine code), a complete assembly listing, and error diagnostics. The object code output from the Assembler may be loaded directly into the Intel Universal PROM Programmer for programming the 2920 EPROM. The object code may also be loaded to the 2920 Simulator for 2920 system design and debug.

The 2920 Assembler runs under the ISIS-II Operating System on the Intellec Microcomputer Development Systems.

Sample 2920 Assembly Listing

```
ISIS-II 2920 ASSEMBLER X102                                          PAGE   1

ASSEMBLER INVOKED BY: AS2920 SAW ASM DEBUG

SAWTOOTH WAVE GENERATOR

  LINE   LOC OBJECT SOURCE STATEMENT

     1              $TITLE('SAWTOOTH WAVE GENERATOR')
     2              ;
     3              ;
     4     0 0000EF     IN0                ; SAMPLE INPUT CHANNEL 0
     5     1 0000EF     IN0
     6     2 0000EF     IN0
     7     3 00BAE8     SUB Y,KP1,IN0      ; SIMULTANEOUSLY CALCULATE SAWTOOTH
     8     4 00BA0A     SUB Y,KP1,R1,IN0   ; BY SUBTRACTING 3/16 FROM Y
     9     5 0044EF     LDA DAR,Y,IN0      ; ALSO CHECK SIGN BIT OF Y
    10     6 78BAED     ADD Y,KP7,CMD$     ; IF Y NEGATIVE START NEXT TOOTH
    11     7 6000EF     CVTS               ; CONVERT SAMPLED INPUT TO DIGITAL (SIGN BIT)
    12     8 7082EF     LDA Y,KP0,CMD$     ; SUPPRESS SAWTOOTH IF INPUT WAS < 0
    13     9 4044EF     LDA DAR,Y          ; PREPARE TO OUTPUT SAWTOOTH
    14    10 4000EF     NOP                ; ANALOG LEVEL MUST SETTLE
    15    11 4000EF     NOP
    16    12 4000EF     NOP
    17    13 8000EF     OUT0               ; OUTPUT SAWTOOTH
    18    14 8000EF     OUT0
    19    15 8000EF     OUT0
    20    16 5000EF     EOP                ; PROGRAM WILL END IN THREE MORE INSTRUCTIONS
    21    17 8000EF     OUT0
    22    18 8000EF     OUT0
    23    19 8000EF     OUT0
    24
    25                  END


SYMBOL:                         VALUE:

Y                                    0


ASSEMBLY COMPLETE
ERRORS   =     0
WARNINGS =     0
RAMSIZE  =     1
ROMSIZE  =    20
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2920 SIMULATOR

Speeds test and debug of 2920 programs

Simulates 2920 internal operation

Operates on Intellec® Microcomputer Development Systems

Allows users to specify 2920 input signals, and display or alter ROM, RAM, and system variables

Output and internal data can be saved on disk for further analysis.

Provides ability to set breakpoints and to collect trace information

Easy-to-learn commands

The 2920 Simulator is a software facility that provides testing and symbolic debugging of 2920 programs in an Intellec Microcomputer Development Systems environment. The 2920 designers have the capability to specify the 2920 input signals, to set breakpoints, to collect and display 2920 input, output, system variables, and ROM and RAM data values during simulation. The 2920 Simulator accepts the hex format object files produced by the 2920 assembler. Output values and internal trace data may be saved on ISIS-II disk files for further analysis.

Functional Description

2920 Input Signal Specification

The four analog signal inputs to the 2920 processor can be specified as algebraic combinations of basic functions of time. The basic functions are SIN, COS, EXP, LOG, SQR, SAW, SQW, ABS

2920 Simulation

The simulation of 2920 machine instructions is performed in software. All 2920 internal registers, memory, input values, output values, and other system variables can be examined and modified. The internal processing of the 2920 is simulated. Time constants for the sample and hold capacitators are assumed to be zero. Calculation of input signals is performed in single precision floating point. The speed of simulation varies with the complexity of the input signal, breakpoint setting, and trace condition. Exclusive of I/O time requirements, 2920 instructions will be simulated at a rate of approximately several hundred instructions per second.

Breakpoint Capabilities

After each instruction is simulated, the breakpoint is evaluated to determine whether to stop or continue simulation. Conditional breakpoints are also provided for debugging purposes. Simulation can be manually stopped at any time by pressing the ESC key on the Intellec console.

Trace Capabilities

Based on the qualifier's condition, trace data records can be collected during simulation. The trace data records are stored in Intellec resident memory and are optionally written to the console for display or to a disk file for record.

Symbolic Debugging Capabilities

The 2920 Simulator allows the user to refer to program addresses symbolically. The user can load or save the symbols generated from the hex format object files or created during the debugging session. 2920 program memory in ROM can be disassembled, or filled with assembled instructions.

The 2920 Simulator is designed to provide users with powerful, easy-to-use commands. The user interfaces to the Simulator by entering commands to the Intellec console. The commands consist of one command line, terminated by one of the two line terminators — carriage return or line feed.

The 2920 Simulator offers two types of commands:

Simulation and Control Commands

Command	Operation
Simulate	Starts simulation of the input signals and the 2920 program in simulated ROM memory. Initial setting is "FOREVER."
Trace	Controls the trace selection. Initial setting is "TIME."
Qualifier	Sets qualifier condition during trace. Initial setting is "ALWAYS."
Breakpoint	Sets breakpoint condition during simulation. Initial setting is "NEVER."

AFN-01386A

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Interrogation and Utility Commands

Command	Operation
Display	Displays the values of symbols, RAM, ROM, input, output, registers and system variables
Change	Alters the values of symbols, RAM, ROM, input, register and system variables
Base	Establishes the mode of display for output data.
Suffix	Establishes the mode of display for input data.
Load	Fetches user symbol table and object code from input device.
Save	Sends user symbol table and object code to output device
Define	Enters symbol name and value to user symbol table
Console	Controls the console on/off display
List	Defines list device
Exit	Returns program control to ISIS-II.
Evaluate	Converts expression to equivalent values in binary, decimal, and hex
Remove	Deletes symbols from symbol table
Help	Provides a brief summary of the syntax for the command languages.
Graphics On/Off	Switches output mode between list and graphics.
X Size	Enters horizontal display size.

Keyword References

The 2920 Simulator provides users with keyword references to gain access to all of the numeric valued system variables including simulated 2920's memory, register, status flags and input/output. These keyword references can function as the evaluation command, display command, and change command.

- **2920 Processor Keyword References**

IN0	Analog input 0 in volts
IN1	Analog input 1 in volts
IN2	Analog input 2 in volts
IN3	Analog input 3 in volts
OUT0	Analog output 0 in volts (read only)
OUT1	Analog output 1 in volts (read only)
OUT2	Analog output 2 in volts (read only)
OUT3	Analog output 3 in volts (read only)
OUT4	Analog output 4 in volts (read only)
OUT5	Analog output 5 in volts (read only)
OUT6	Analog output 6 in volts (read only)
OUT7	Analog output 7 in volts (read only)
IN	Sampled and held analog input signal in volts
DAR	Digital to analog register (RAM location 40)
PC	Program counter (integer 1 to 192)
CY	Carry (integer 0 or 1)
OVF	Overflow (integer 0 or 1, read only)
OVE	Overflow enable (integer 0 or 1)

- **Software Simulator Keyword References**

TIME	Elapsed simulated time in seconds (read only)
TQUAL	Time when the qualifier last matched in seconds (read only)
COUNT	Number of instructions simulated since last SIMULATE command (integer, read only)
BUFFERSIZE	Number of trace data records (integer, read only)
TINST	Time between successive instructions in seconds (read only)
SIZE	Number of instructions in program disregarding actual EOP placement
TPROG	Time between successive program passes in seconds
VREF	Reference analog level voltage in volts

The above keyword references are designed to aid 2920 program debugging.

ISIS Compatibilities

The 2920 software simulator runs under the ISIS "submit" facility. The 2920 software simulator uses the ISIS-II line editing capabilities to correct errors in an input line on the Intellec console.

Sample 2920 Simulation Session

```
-
-SM2920.SFT

ISIS-II 2920 SIMULATOR, V1.1
*
*;   THIS IS THE SIMULATION OF THE 'SAWTOOTH GENERATOR'
*
*LIST SRC.LOC        ; LISTS THE SIMULATION SESSION TO AN ISIS FILE
*LOAD SRC.HEX        ; LOAD THE OBJECT CODE INTO THE 2920 SIMULATOR
*ROM 0 TO 5          ; DISPLAY SRC PROGRAM
ROM 000 = LDA .K,KP5,R00,NOP
ROM 001 = ADD .K,KP1,R05,NOP
ROM 002 = LDA .K,.K,R02,NOP
ROM 003 = SUB .OSC,.K,R00,NOP
ROM 004 = LDA DAR,.OSC,R00,NOP
ROM 005 = ADD .OSC,KP4,L01,CNDS
*TPROG=1/10000       ; SET THE SAMPLE RATE
*TRA=PC,RAM .K       ; SET THE ITEMS TO BE TRACED
*BASE=B              ; DISPLAY THE RESULTS IN BINARY
*SIMULATE FROM 0 TILL COUNT=3 ; SIMULATE THREE INSTRUCTIONS
                               TO VERIFY CONSTANT

   PC                          RAM 0
SIMULATION BEGUN
    1.000000000000000000000000    0.101000000000000000000000
    2.0000000E+0                  0.101000001000000000000000
    3.0000000E+0                  0.001010000100000000000000
SIMULATION TERMINATED
*QUALIFIER=PC=0      ; TRACE EVERY PROGRAM PASS
*TRACE=T,DAR,RAM .OSC ; SET THE ITEMS TO BE TRACED
*RAM .OSC=ONE        ; INITIALIZE THE RAM LOCATION
*BREAKPOINT=T>.00132 ; SIMULATE FOR TWO CYCLES
*BASE=D              ; SET THE BASE TO DECIMAL
*SIMULATE FROM 0     ; BEGIN SIMULATION
   T                  DAR                 RAM 1
```


Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
SIMULATION BEGIN
      0.00010000        0.83984375        0.84277344
      0.00020000        0.68359375        0.68554688
      0.00030000        0.52734375        0.52832026
      0.00040000        0.36718750        0.37109370
      0.00050000        0.21093750        0.21386719
      0.00060000        0.05468750        0.05664056
      0.00070000       -0.10156250        0.09941396
      0.00080000        0.73828125        0.74218745
      0.00090000        0.58203125        0.58496089
      0.00100000        0.42578125        0.42773443
      0.00110000        0.26953125        0.27050776
      0.00120000        0.11328125        0.11132819
      0.00130000       -0.04687500        0.95605459
SIMULATION TERMINATED
*GRAPH ON          ; SWITCHES THE DISPLAY MODE TO GRAPHICS
*TRACE=T,0,DAR,RAM .OSC,-1,-1,1,1   ; SETS ITEMS TO BE TRACED
*RAM .OSC=ONE      ; INITIALIZE THE RAM LOCATION
*SIMULATE FROM 0
    T                0                DAR               RAM 1             -1
         -1                1                  1
SIMULATION BEGUN
->*                                    1                                    *
0 *                                    1                 *                  *
. *                                    1                       *            *
0 *                                    1              *                     *
0 *                                    1         *                          *
0 *                                    1    *                               *
1 *                                  2 1                              1      *
0 *                                    1                       *            *
0 *                                    1                  *                 *
0 *                                    1             *                      *
0 *                                    1        *                           *
  *                                    1    *                               *
  *                                  2 1                                   3 *
SIMULATION TERMINATED
*EXIT
```

SPECIFICATIONS

Operating Equipment

Required Hardware

Intellec® Microcomputer Development System
RUNNING ISIS

Required Software

ISIS-II Diskette Operating System

Optional Hardware

Line Printer
Universal PROM Programmer

Optional Software

FORTRAN-80 (Product Code MDS-301)

Documentation Package

2920 Assembly User's Guide (9800987)
2920 Simulator User's Guide (9800988)
2920 Signal Processing Application Compiler User's Guide (121529)

Shipping Media

Flexible Diskettes

ORDERING INFORMATION

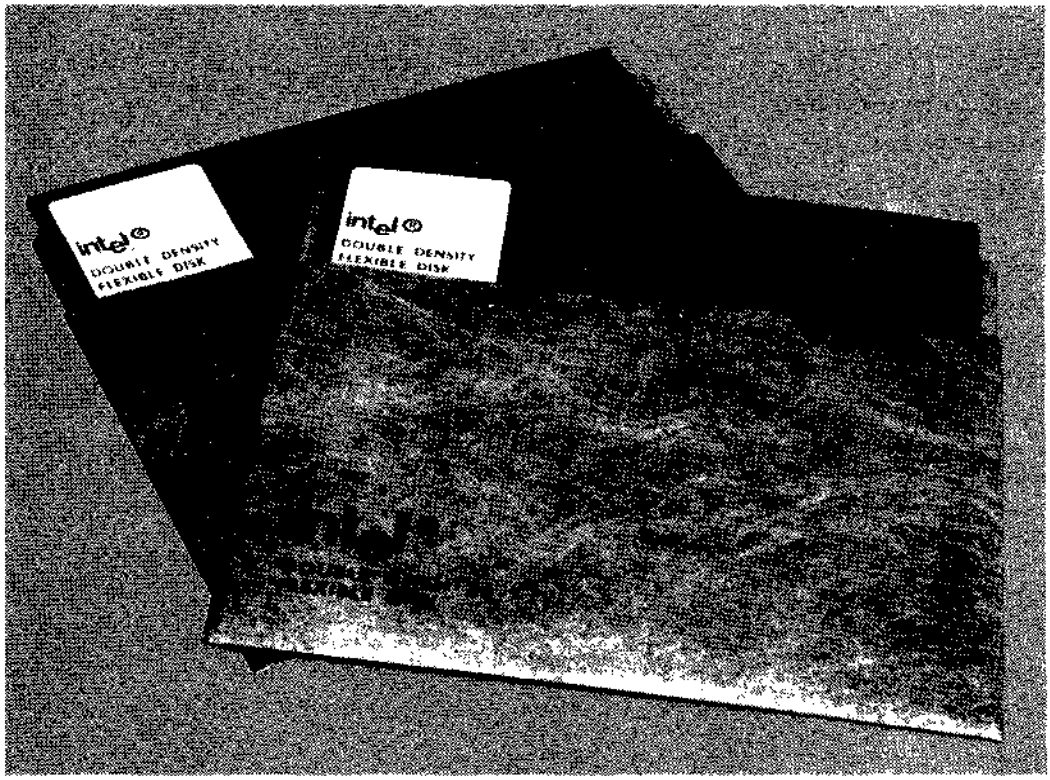
Product Code	Description
MCI-20-SPS	2920 Software Support Package Includes 2920 Signal Processing Application Software/Compiler and 2920 Assembler/Similator Software

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

MCS® -48
DISKETTE-BASED SOFTWARE SUPPORT PACKAGE

- **Extends Intellec microcomputer development system to support MCS-48 development**
- **MCS-48 assembler provides conditional assembly and macro capability**
- **Takes advantage of powerful ISIS-II file handling and storage capabilities**
- **Provides assembler output in standard Intel hex format**

The MCS-48 assembler translates symbolic 8048 assembly language instructions into the appropriate machine operation codes, and provides both conditional and macroassembler programming. Output may be loaded either to an ICE-49 module for debugging or into the iUP Universal PROM Programmer for 8748 PROM programming. The MCS-48 assembler operates under the ISIS-II operating system on Intel Development systems.



©INTEL CORPORATION, 1983

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FUNCTIONAL DESCRIPTION

The MCS-48 assembler translates symbolic 8048 assembly language instructions into the appropriate machine operation codes. The ability to refer to program addresses with symbolic names eliminates the errors of hand translation and makes it easier to modify programs when adding or deleting instructions. Conditional assembly permits the programmer to specify which portions of the master source document should be included or deleted in variations on a basic system design, such as the code required to handle optional external devices. Macro capability allows the programmer use of a single label to define a routine. The MCS-48 assembler will assemble the code required by the reserved routine whenever the macro label is inserted in the text. Output from the assembler is in standard Intel hex format. It may be either loaded directly to an in-circuit emulator (ICE-49) module for integrated hardware/software debugging, or loaded into the iUP Universal PROM Programmer for 8748 PROM programming. A sample assembly listing is shown in Table 1.

The MCS 48 assembler supports the 8048, 8049, 8050, 8020, 8021, 8022, 8041 and 8042. The MCS 48 assembler can also support CMOS versions of the 8048 family.

Table 1. Sample MCS-48 Diskette-Based

```
ISIS-II 8048 MACRO ASSEMBLER V1.0                                PAGE 1

LOC  OBJ                    SEQ          SOURCE STATEMENT

                              1          ;DECIMAL ADDITION ROUTINE: ADD BCD NUMBER
                              2          ;AT LOCATION 'BETA' TO BCD NUMBER AT 'ALPHA' WITH
                              3          ;RESULT IN 'ALPHA'. LENGTH OF NUMBER IS 'COUNT' DIGIT
                              4          ;PAIRS. (ASSUME BOTH 'BETA' AND 'ALPHA' ARE SAME LENGTH
                              5          ;AND HAVE EVEN NUMBER OF DIGITS OR MSD IS 0 IF
                              6          ;ODD)
                              7  INIT     #MACRO     AUGEND,ADDEND,CNT
                              8             MOV     R0,#AUGEND
                              9  L1         MOV     R1,#ADDEND
                             10             MOV     R2,#CNT
                             11             ENDM
                             12
001E                         13  ALPHA    EQU     30
0028                         14  BETA     EQU     40
0005                         15  COUNT    EQU     5
0100                         16             ORG     100H
                             17             INIT    ALPHA,BETA,COUNT
0100 B81E                    18+            MOV     R0,#ALPHA
0102 B928                    19+L1          MOV     R1,#BETA
0104 BA05                    20+            MOV     R2,#COUNT
0106 97                      21             CLR     C
0107 F0                      22  LP         MOV     A,@R0
0108 71                      23             ADDC    A,@R1
0109 57                      24             DA      A
010A A1                      25             MOV     @R0,A
010B 18                      26             INC     R0
010C 19                      27             INC     R1
010D EA07                    28             DJNZ    R2,LP
                             29             END

USER SYMBOLS
ALPHA 001E   BETA 0028   COUNT 0005   LP 0107
L1  0102

ASSEMBLY COMPLETE, NO ERRORS

ISIS-II ASSEMBLER SYMBOL CROSS REFERENCE V1.0                    PAGE 1

SYMBOL CROSS REFERENCE

ALPHA   30#   17
BETA    40#   17
COUNT   5#    17
INIT    7#    17
L1      9#
LP      22#   28
```

SPECIFICATIONS

Operating Environment

(All) Intel Microcomputer Development Systems
(Series II, Series III/Series IV)
Intel Personal Development System

Documentation Package

Titles of: User Guides
Operating Instructions
Reference Manuals

Ordering Information

Part Number	Description
MDS-D48*	MCS-48 Disk Based Assembler Requires Software License

SUPPORT:

Hotline Telephone Support, Software Performance Reports (SPR), Software Updates, Technical Reports, Monthly Newsletters are available.

*MDS is an ordering code only and is not used as a product name or trademark. MDS is a registered trademark of Mohawk Data Sciences Corporation.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8051
SOFTWARE PACKAGES

PL/M51 Software Package Contains the following:

- **PL/M51 Compiler which is designed to support all phases of software implementation**
- **RL51 Linker and Relocator which enables programmers to develop software in a modular fashion**
- **LIB51 Librarian which lets programmers create and maintain libraries of software object modules**

8051 Software Development Package Contains the following:

- **8051 Macro Assembler which gives symbolic access to 8051 hardware features**
- **RL51 Linker and Relocator program which links modules generated by the assembler**
- **CONV51 which enables software written for the MCS®-48 family to be up graded to run on the 8051**
- **LIB51 Librarian which lets programmers create and maintain libraries of software object modules**

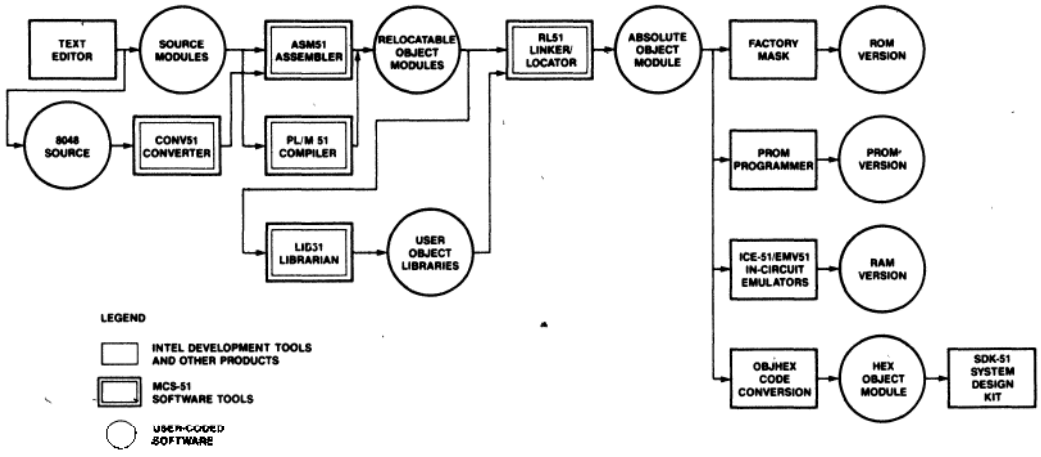


Figure 1. MCS®-51 Program Development Process

Intel Corporation Assumes No Responsibility for the Use of Any Circuitry Other Than Circuitry Embodied in an Intel Product. No Other Circuit Patent Licenses are Implied. Information Contained Herein Supercedes Previously Published Specifications On These Devices From Intel.
© INTEL CORPORATION, 1983

MARCH 1984
ORDER NUMBER: 182771-002

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PL/M 51 SOFTWARE PACKAGE

- **High-level programming language for the Intel MCS®-51 single-chip microcomputer family**
- **Compatible with PL/M 80 assuring MCS®-80/85 design portability**
- **Enhanced to support boolean processing**
- **Tailored to provide an optimum balance among on-chip RAM usage, code size and code execution time**
- **Allows programmer to have complete control of microcomputer resources**
- **Produces relocatable object code which is linkable to object modules generated by all other 8051 translators**
- **Extends high-level language programming advantages to microcontroller software development**
- **Improved reliability, lower maintenance costs, increased programmer productivity and software portability**
- **Includes the linking and relocating utility and the library manager**
- **Supports all members of the Intel MCS®-51 architecture**

PL/M 51 is a structured, high-level programming language for the Intel MCS-51 family of microcomputers. The PL/M 51 language and compiler have been designed to support the unique software development requirements of the single-chip microcomputer environment. The PL/M language has been enhanced to support Boolean processing and efficient access to the microcomputer functions. New compiler controls allow the programmer complete control over what microcomputer resources are used by PL/M programs.

PL/M 51 is largely compatible with PL/M 80 and PL/M 86. A significant proportion of existing PL/M software can be ported to the MCS-51 with modifications to support the MCS-51 architecture. Existing PL/M programmers can start programming for the MCS-51 with a small relearning effort.

PL/M 51 is the high-level alternative to assembly language programming for the MCS-51. When code size and code execution speed are not critical factors, PL/M 51 is the cost-effective approach to developing reliable, maintainable software.

The PL/M 51 compiler has been designed to support efficiently all phases of software implementation with features like a syntax checker, multiple levels of optimization, cross-reference generation and debug record generation.

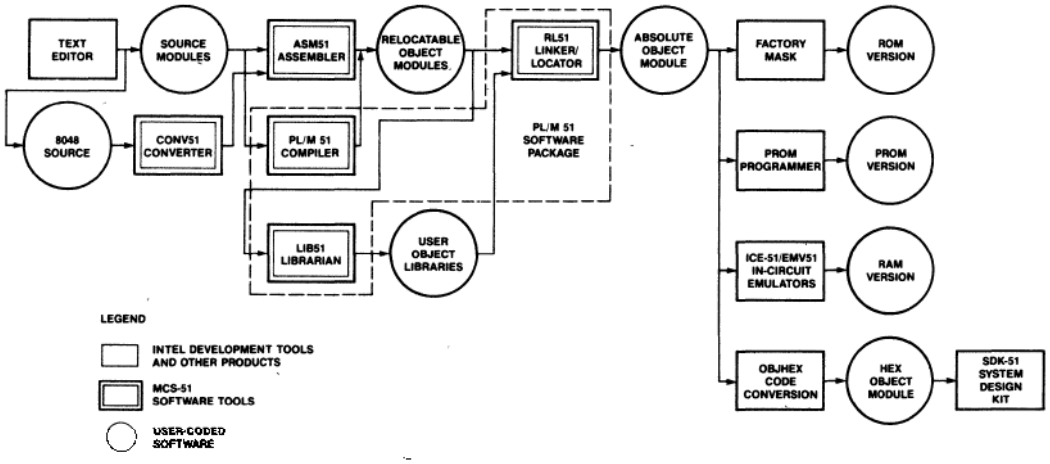


Figure 2. PL/M51 Software Package

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PL/M 51 Compiler

FEATURES

Major features of the Intel PL/M 51 compiler and programming language include:

Structured Programming

PL/M source code is developed in a series of modules, procedures, and blocks. Encouraging program modularity in this manner makes programs more readable, and easier to maintain and debug. The language becomes more flexible, by clearly defining the scope of user variables (local to a private procedure, for example).

Language Compatibility

PL/M 51 object modules are compatible with object modules generated by all other MCS-51 translators. This means that PL/M programs may be linked to programs written in any other MCS-51 language.

Object modules are compatible with In-Circuit Emulators and Emulation Vehicles for MCS-51 processors; the DEBUG compiler control provides these tools with symbolic debugging capabilities.

Supports Three Data Types

PL/M makes use of three data types for various applications. These data types range from one to sixteen bits and facilitate various arithmetic, logic, and address functions:

—Bit: a binary digit
—Byte: 8-bit unsigned number or,
—Word: 16-bit unsigned number.

Another powerful facility allows the use of BASED variables that map more than one variable to the same memory location. This is especially useful for passing parameters, relative and absolute addressing, and memory allocation.

Two Data Structuring Facilities

PL/M 51 supports two data structuring facilities. These add flexibility to the referencing of data stored in large groups.

—Array: Indexed list of same type data elements
—Structure: Named collection of same or different type data elements
—Combinations of Both: Arrays of structures or structures of arrays.

Interrupt Handling

A procedure may be defined with the INTERRUPT attribute. The compiler will generate code to save and restore the processor status, for execution of the user-defined interrupt handler routines.

Compiler Controls

The PL/M 51 compiler offers controls that facilitate such features as:

—Including additional PL/M 51 source files from disk
—Cross-reference
—Corresponding assembly language code in the listing file

Program Addressing Control

The PL/M 51 compiler takes full advantage of program addressing with the ROM (SMALL/MEDIUM/LARGE) control. Programs with less than 2 KB code space can use the SMALL or MEDIUM option to generate optimum addressing instructions. Larger programs can address over the full 64 KB range.

Code Optimization

The PL/M 51 compiler offers four levels of optimization for significantly reducing overall program size.

—Combination or "folding" of constant expressions; "Strength reductions" (a shift left rather than multiply by 2)
—Machine code optimizations; elimination of superfluous branches
—Automatic overlaying of on-chip RAM variables
—Register history: an off-chip variable will not be reloaded if its value is available in a register.

Error Checking

The PL/M 51 compiler has a very powerful feature to speed up compilations. If a syntax or program error is detected, the compiler will skip the code generation and optimization passes. This usually yields a 2X performance increase for compilation of programs with errors.

A fully detailed set of programming and compilation error messages is provided by the compiler and user's guide.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

BENEFITS

PL/M 51 is designed to be an efficient, cost-effective solution to the special requirements of MCS-51 Microsystem Software Development, as illustrated by the following benefits of PL/M use:

Low Learning Effort

PL/M 51 is easy to learn and to use, even for the novice programmer.

Earlier Project Completion

Critical projects are completed much earlier than otherwise possible because PL/M 51, a structured high-level language, increases programmer productivity.

Lower Development Cost

Increases in programmer productivity translate immediately into lower software development costs because less programming resources are required for a given programmed function.

Increased Reliability

PL/M 51 is designed to aid in the development of reliable software (PL/M programs are simple statements of the program algorithm). This substantially reduces the risk of costly correction of errors in systems that have already reached full production status, as the more simply stated the program is, the more likely it is to perform its intended function.

Easier Enhancements and Maintenance

Programs written in PL/M tend to be self-documenting, thus easier to read and understand. This means it is easier to enhance and maintain PL/M programs as the system capabilities expand and future products are developed.

RL51 Linker and Relocator

- **Links modules generated by the assembler and the PL/M compiler**
- **Locates the linked object to absolute memory locations**
- **Enables modular programming of software-efficient program development**
- **Modular programs are easy to understand, maintainable and reliable**

The MCS-51 linker and relocator (RL51) is a utility which enables MCS-51 programmers to develop software in a modular fashion. The utility resolves all references between modules and assigns absolute memory locations to all the relocatable segments, combining relocatable partial segments with the same name.

With this utility, software can be developed more quickly because small functional modules are easier to understand, design and test than large programs.

The total number of allowed symbols in user-developed software is very large because the assembler number of symbols' limit applies only per module, not to the entire program. Therefore programs can be more readable and better documented.

Modules can be saved and used on different programs. Therefore the software investment of the customer is maintained.

RL51 produces two files. The absolute object module file can be directly executed by the MCS-51 family. The listing file shows the results of the link/locate process.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

LIB51 Librarian

The LIB51 utility enables MCS-51 programmers to create and maintain libraries of software object modules. With this utility, the customer can develop standard software modules and place them in libraries, which programs can access through a standard interface. When using object libraries, the linker will call only object modules that are required to satisfy external references.

Consequently, the librarian enables the customer to port and reuse software on different projects —thereby maintaining the customer's software investment.

SPECIFICATIONS

Operating Environment

All Intel Microcomputer Development Systems or Intel Personal Development Systems

Documentation Package

PL/M 51 User's Guide
MCS-51 Utilities User's Guide

ORDERING INFORMATION

Part Number	Description
iMDX 352 Requires Software License	PL/M 51 Software Package

SUPPORT:

Hotline Telephone Support, Software Performance Report (SPR), Software Updates, Technical Reports, and monthly Technical Newsletters are available.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

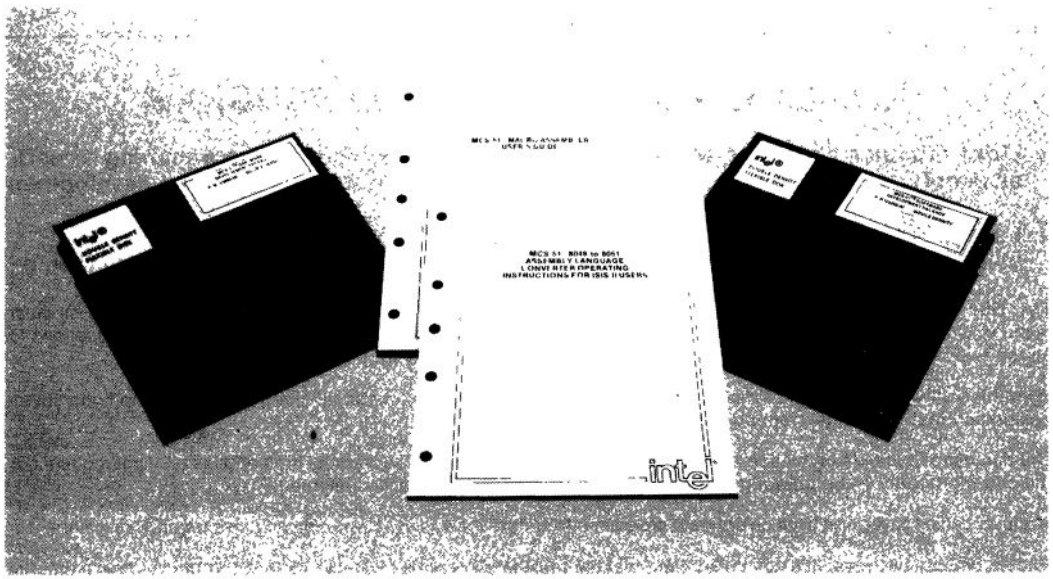
8051 SOFTWARE DEVELOPMENT PACKAGE

- **Symbolic relocatable assembly language programming for 8051 microcontrollers**
- **Extends Intellec® Microcomputer Development System to support 8051 program development**
- **Produces Relocatable Object Code which is linkable to other 8051 Object Modules**
- **Encourage modular program design for maintainability and reliability**
- **Macro Assembler features conditional assembly and macro capabilities**
- **CONV51 Converter for translation of 8048 assembly language source code to 8051 assembly language source code**
- **Provides upward compatibility from the MCS-48™ family of single-chip microcontrollers**

The 8051 software development package provides development system support for the powerful 8051 family of single chip microcomputers. The package contains a symbolic macro assembler and MCS-48 source code converter.

The assembler produces relocatable object modules from 8051 macro assembly language instructions. The object code modules can be linked and located to absolute memory locations. This absolute object code may be used to program the 8751 EPROM version of the chip. The assembler output may also be debugged using the ICE-51™ in-circuit emulator.

The converter translates 8048 assembly language instructions into 8051 source instructions to provide software compatibility between the two families of microcontrollers.



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8051 MACRO ASSEMBLER

- **Supports 8051 family program development on Intellec® Microcomputer Development Systems**
- **Gives symbolic access to powerful 8051 hardware features**
- **Produces object file, listing file and error diagnostics**
- **Object files are linkable and locatable**
- **Provides software support for many addressing and data allocation capabilities**
- **Symbolic Assembler supports symbol table, cross-reference, macro capabilities, and conditional assembly**

The 8051 Macro Assembler (ASM51) translates symbolic 8051 macro assembly language modules into linkable and locatable object code modules. Assembly language mnemonics are easier to program and are more readable than binary or hexadecimal machine instructions. By allowing the programmer to give symbolic names to memory locations rather than absolute addresses, software design and debug are performed more quickly and reliably. Furthermore, since modules are linkable and relocatable, the programmer can do his software in modular fashion. This makes programs easy to understand, maintainable and reliable.

The assembler supports macro definitions and calls. This is a convenient way to program a frequently used code sequence only once. The assembler also provides conditional assembly capabilities.

Cross referencing is provided in the symbol table listing, showing the user the lines in which each symbol was defined and referenced.

ASM51 provides symbolic access to the many useful addressing features of the 8051 architecture. These features include referencing for bit and byte locations, and for providing 4-bit operations for BCD arithmetic. The assembler also provides symbolic access to hardware registers, I/O ports, control bits, and RAM addresses. ASM51 can support all members of the 8051 family.

Math routines are enhanced by the MULtiply and DIVide instructions.

If an 8051 program contains errors, the assembler provides a comprehensive set of error diagnostics, which are included in the assembly listing or on another file. Program testing may be performed by using the iUP Universal Programmer and iUP F87/51 personality module to program the 8751 EPROM version of the chip.

ICE51 and EMV51 are available for program debugging.

RL51 LINKER AND RELOCATOR PROGRAM

- **Links modules generated by the assembler**
- **Locates the linked object to absolute memory locations**
- **Enables modular programming of software for efficient program development**
- **Modular programs are easy to understand, maintainable and reliable**

The 8051 linker and relocator (RL51) is a utility which enables 8051 programmers to develop software in a modular fashion. The linker resolves all references between modules and the relocator assigns absolute memory locations to all the relocatable segments, combining relocatable partial segments with the same name.

With this utility, software can be developed more quickly because small functional modules are easier to understand, design and test than large programs.

The number of symbols in the software is very large because the assembler symbol limit applies only per module not the entire program. Therefore programs can be more readable and better documented.

Modules can be saved and used on different programs. Therefore the software investment of the customer is maintained.

RL51 produces two files. The absolute object module file can be directly executed by the 8051 family. The listing file shows the results of the link/locate process.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

CONV51
8048 TO 8051 ASSEMBLY LANGUAGE CONVERTER UTILITY PROGRAM

- **Enables software written for the MCS-48™ family to be upgraded to run on the 8051**
- **Maps each 8048 instruction to a corresponding 8051 instruction**
- **Preserves comments; translates 8048 macro definitions and calls**
- **Provides diagnostic information and warning messages embedded in the output listing**

The 8048 to 8051 Assembly Language Converter is a utility to help users of the MCS-48 family of microcomputers upgrade their deisgns with the high performance 8051 architecture. By converting 8048 source code to 8051 source code, the software investment developed for the 8048 is maintained when the system is upgraded.

The goal of the converter (CONV51) is to attain functional equivalence with the 8048 code by mapping each 8048 instruction to a corresponding 8051 instruction. In some cases a different instruction is produced because of the enhanced instruction set (e.g., bit CLR instead of ANL).

Although CONV51 tries to attain functional equivalence with each instruction, certain 8048 code sequences cannot be automatically converted. For example, a delay routine which depends on 8048 execution speed would require manual adjustment. A few instructions, in fact, have no 8051 equivalent (such as those involving P4-P7). Finally, there are a few areas of possible intervention such as PSW manipulation and interrupt processing, which at least require the user to confirm proper translation. The converter always warns the user when it cannot guarantee complete conversion.

CONV51 produces two files. The output file contains the ASM51 source program produced from the 8048 instructions. The listing file produces correlated listings of the input and output files, with warning messages in the output file to point out areas that may require users' intervention in the conversion.

LIB51 LIBRARIAN

The LIB51 utility enables MCS-51 programmers to create and maintain libraries of software object modules. With this utility, the customer can develop standard software modules and place them in libraries, which programs can access through a standard interface. When using object libraries, the linker will call only object modules that are required to satisfy external references.

Consequently, the librarian enables the customer to port and reuse software on different projects—thereby maintaining the customer's software investment.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SPECIFICATIONS

OPERATING ENVIRONMENT

All Intel Microcomputer Development Systems or Intel Personal Development System

Documentation Package:

MCS-51 Macro Assembler User's Guide
MCS-51 Utilities User's Guide for 8080/8085 Based Development System
MCS-51 8048-to-8051 Assembly Language Converter Operating Instructions for ISIS-II Users

ORDERING INFORMATION

Part Number	Description
MCI-51-ASM	8051 Software Development Package

*Requires Software License

SUPPORT:

Hotline Telephone Support, Software Performance Reporting (SPR), Software Updates, Technical Reports, Monthly Newsletter available.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

iRMX™ 51
REAL-TIME MULTITASKING EXECUTIVE

- **Software tool for family of 8051 microcontroller based applications**
- **Real-time, multitasking executive**
- **Supports remote task communication**
- **Small — 2.2K Bytes**
- **Reliable**
- **Simple user interface**
- **Compatible with BITBUS™/Distributed Control Modules (iDCM) product line: iSBX™ 344 & iRCB 44/10 boards**

The iRMX™ 51 Executive is a compact, easy to use, software tool for development and implementation of applications built on the high performance 8-bit family of 8051 microcontrollers. A few members of this expansive family are the 8051, 8044, and 8052 microcontrollers. Like the 8051 family, the iRMX 51 Executive incorporates many features that make it exceptionally well suited for real-time control applications requiring manipulation and scheduling of more than one job, and fast response to external stimuli.

The 8051 microcontroller family is the family of choice for applications such as: data acquisition and monitoring, process control, robotics, and machine control. Using the iRMX 51 Executive for a foundation can significantly reduce applications development time. Also, the iRMX 51 Executive fully supports Intel's BITBUS™ microcontroller interconnect expressly designed for reliable high performance real-time control.

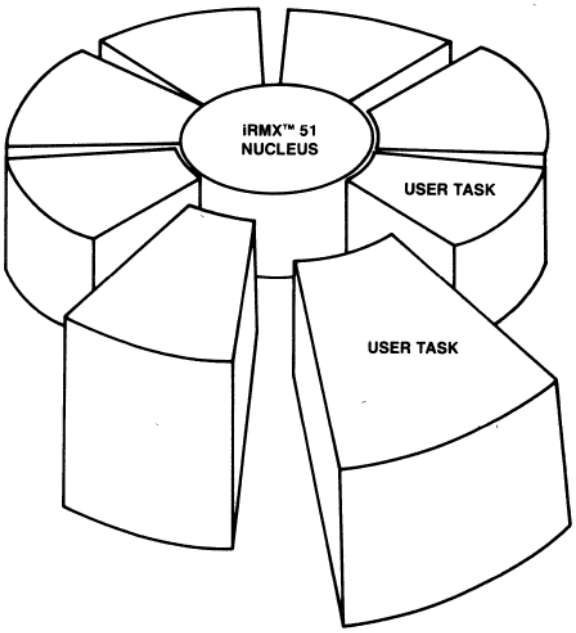


Figure 1. Structure Diagram

ORDER NUMBER 230972-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

ARCHITECTURE

Real-time and Multitasking

Real-time control applications must be responsive to the external environment and typically involve the execution of more than one function (task or set of tasks) in response to different external stimuli. Control of an industrial drying process is an example. This process could require monitoring of multiple temperatures and humidity; control of fans, heaters, and motors that must respond accordingly to a variety of inputs. The iRMX 51 Executive fully supports applications requiring response to stimuli as they occur ie. in real-time. This real-time response is supported for multiple tasks often needed to implement a control application.

Some of the facilities precisely tailored for development and implementation of real-time control application systems provided by the iRMX 51 Executive are: task management, interrupt handling, message passing, and when intergrated with communications support, message passing with different microcontrollers. Also, the iRMX 51 Executive is driven by events: interrupts, timers, and messages ensuring the application system always responds to the environment appropriately.

Task Management

A task is a program defined by the user to execute a particular control function or functions. Multiple programs or tasks may be required to implement a particular function such as 'controlling Heater 1'. The iRMX 51 Executive recognizes three different task states as one of the mechanisms to accomplish scheduling of up to eight tasks. Figure 2 illustrates the different task states and their relationship to one another.

The scheduling of tasks is priority based. The user can prioritize tasks to reflect their relative importance within the overall control scheme. For instance, if Heater 1 must go off line prior to Heater 2 then the task associated with Heater 1 shutdown could be assigned a higher priority ensuring the correct shutdown sequence. The RQ WAIT system call is also a scheduling tool. In this example the task implementing Heater 2 shutdown could include an instruction to wait for completion of the task that implements Heater 1 shutdown.

The iRMX 51 Executive allows for PREEMPTION of a task that is currently being executed. This means that if some external event occurs such as a catastrophic failure of Heater 1, a higher priority task associated with the interrupt, message, or timeout resulting from the failure will preempt the running task. Preemption ensures the emergency will be responded to immediately. This is crucial for real-time control application systems.

Interrupt Handling

The iRMX 51 executive supports sixteen interrupt sources as shown in Table 1. Four of these interrupt sources, excluding timer 0, can be as-

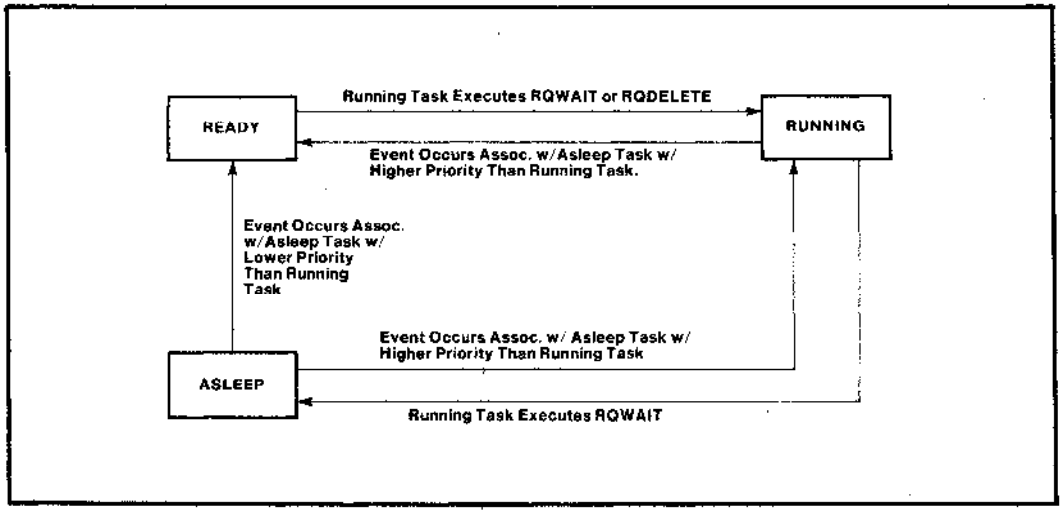


Figure 2. Task State Transition Diagram

230972-001

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

signed to a task. When one of the interrupts occurs the task associated with it becomes a running task (if it were the highest priority task in a ready state). In this way, the iRMX 51 Executive responds to a number of internal and external stimuli including time intervals designated by the user.

Table 1. iRMX™ 51 Interrupt Sources

INTERRUPT SOURCE	INTERRUPT NUMBER
External Request 0	00H
Timer 0	01H
External Request 1	02H
Timer 1	03H
Internal Serial Port 1	04H
Reserved	05H
Reserved	06H
Reserved	07H
Reserved	08H
Reserved	09H
Reserved	0AH
Reserved	0BH
Reserved	0CH
Reserved	0DH
Reserved	0EH
Reserved	0FH

Message Passing

The iRMX 51 Executive allows tasks to interface with one another via a simple message passing facility. This message passing facility can be extended to different processors when communications support is integrated within a BITBUS/iDCM system, for example. This facility provides the user with the ability to link different functions or tasks. Linkage between tasks/functions is typically required to support development of complex control applications with multiple sensors (inputs variables) and drivers (output variables). For instance, the industrial drying process might require a dozen temperature inputs, six moisture readings, and control of: three fans, two conveyor motors, a dryer motor, and a pneumatic conveyor. The data gathered from both the temperature and humidity sensors could be processed. Two tasks might be required to gather the data and process it. One task could perform a part of the analysis, then include a pointer to the next task to complete the next part of the analysis . The tasks could continue to move between one another.

REMOTE TASK COMMUNICATION

The iRMX 51 Executive system calls can support communication to tasks on remote controllers. This feature makes the iRMX 51 Executive ideal for applications using distributed architectures. Providing communication support saves significant application development time and allows for more effective use of this time. Intel's iDCM product line combines hardware and software to provide this function.

In an iDCM system, communication between nodes occurs via the BITBUS microcontroller interconnect. The BITBUS microcontroller interconnect is a high performance serial control bus specifically intended for use in applications built on distributed architectures. The iRMX 51 Executive provides BITBUS support.

BITBUS™/iDCM COMPATIBLE

A pre-configured version of the iRMX 51 Executive implements the BITBUS message format and provides all iRMX 51 facilities mentioned previously: task management, interrupt handling, and message passing. This version of the Executive is supplied in firmware on the iDCM Controller with the iDCM hardware products: the iSBX 344 BITBUS Controller MULTIMODULE and the iRCB 44/10 BITBUS Remote Controller boards. It is also supplied on diskette as part of the iRMX 510 iDCM Support Package to ease development of BITBUS systems.

SIMPLE USER INTERFACE

The iRMX 51 Executive's capabilities are utilized through system calls. These interfaces have been defined for ease of use and simplicity. Table 2 includes a listing of these interfaces and their functions. Note tasks may be created at system initialization or run-time using the CREATE TASK call.

Functions such as GET FUNCTION IDS, ALLOCATE/DEALLOCATE BUFFER, and SEND MESSAGE (Messages in the iRMX 51 Executive have a maximum size of 255 bytes.), support communication for distributed architectures. Architectures that define multiple remote stations requiring intelligent and dumb I/O manipulation. The remaining

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 2. iRMX™ 51 System Interfaces

COMMAND	DESCRIPTION
RQ SEND MESSAGE	Sends a message (a command from the BITBUS master, a response from a slave, or a simple message between tasks on the same BITBUS component) to another task.
RQ WAIT	Waits for an interrupt, an event time-out, a message, or any combination of the three.
RQ CREATE TASK	Causes a new sequence of code to be run as an iRMX 51 task with a specific function identification code and priority.
RQ DELETE TASK	Stops the specified task and removes it from all execution lists.
RQ ALLOCATE	Allocates a fixed-length buffer from the on-chip, scratch-pad RAM for general use, or, in BITBUS applications, for a BITBUS message buffer.
RQ DEALLOCATE	Returns an on-chip buffer to the system.
RQ SET INTERVAL	Set the time interval to be used as a separate event-timer for the task.
RQ ENABLE INTERRUPT	Allow external interrupts to signal the microcontroller.
RQ DISSABLE INTERRUPT	Stops all external interrupts from signaling the microcontroller.
RQ GET FUNCTION ID	Provides a list of the 8 function identification codes representing the tasks currently operating on the microcontroller.

interfaces allow the user to specify the system's response to the external environment — a must for real-time control.

Another feature that eases application development is automatic register bank allocation. The Executive will assign tasks to register banks automatically unless a specific request is made. The iRMX 51 Executive keeps track of the register assignments allowing the user to concentrate on other activities.

The user configures an iRMX 51 system simply by: specifying the initial set of task descriptors and configuration values, and linking the system via the RL 51 Linker and Locator Program with user programs. The nature of the task descriptors allows the user to develop programs, locate them in off-chip ROM, and access them without writing additional code. Programs may be written in ASM 51 or PL/M 51. (Intel's 8051 Software Development Package contains both ASM 51 and RL 51. The iRMX 51 Executive supplies the configuration file and macro defining initial task descriptors.) Figure 3 shows the relationships that exist in the system generation process.

RELIABILE

Real-time control applications require reliability. The nucleus requires about 2K bytes of code space, 40 bytes on-chip RAM, & 218 bytes external RAM. Streamlined code increases performance and reliability, and flexibility is not sacrificed as code may be added to either on-chip or external memory.

The iRMX 51 architecture and simple user interface further enhance reliablity and lower cost. For example, the straightforward structure of the user interfaces, and the transparent nature of the scheduling process contribute to reliability of the overall system by minimizing programming effort. Also, modularity increases reliability of the system and lowers cost by allowing user tasks to be refined independent of the system. In this way, errors are identified earlier and can be easily corrected in each isolated module.

In addition, users can assign tasks a Function ID that allows tracking of the tasks associated with a particular control/monitoring function. This feature reduces maintenance and trouble shooting time thus increasing system run time and decreasing cost.

OPERATING ENVIRONMENT

The iRMX 51 Executive supports applications development based on any member of the high performance 8051 family of microcontrollers. The Executive is available on diskette with user linkable libraries or in the Distributed Control Modules (iDCM) controller preconfigured in on-

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

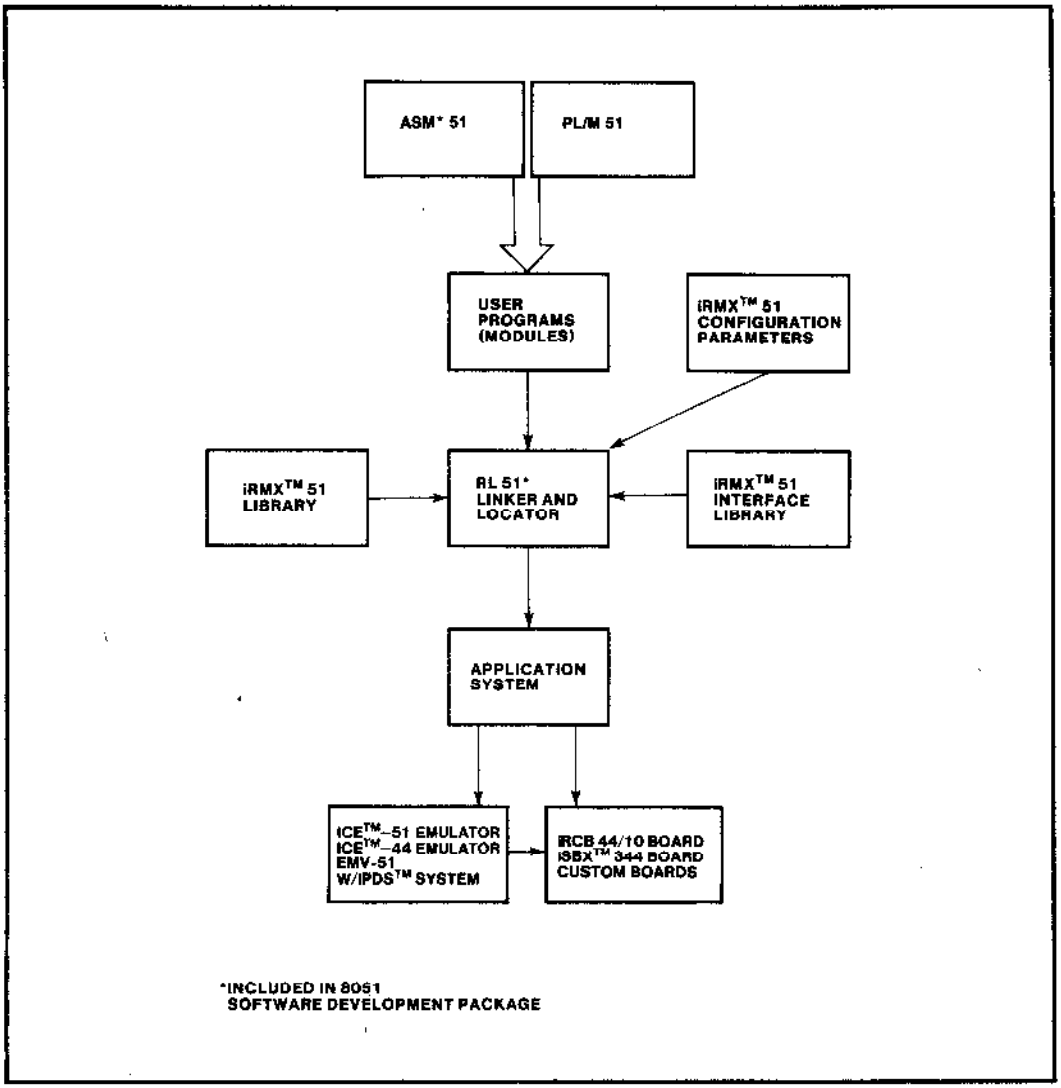


Figure 3. System Generation Process

chip ROM. (The iDCM controller is an 8044 component that consists of an 8051 microcontroller and SDLC controller on one chip with integral firmware.)

When in the iDCM environment (Figure 4), the iRMX 51 Executive can communicate with iRMX based systems like the System 286/310 or ISIS based systems like the Intel Portable Development System (iPDS) by using the iRMX 510 iDCM Support Package.

DEVELOPMENT ENVIRONMENT

Intel provides a complete development environment for the 8051 family of microcontrollers. This environment encompasses iDCM system (BITBUS based) applications also. Software development support consists of: the 8051 Software Development Package, and the iRMX 510 iDCM Support Package. Hardware tools consist of a variety of In Circuit Emulators (ICE), Intel's Portable Development System (iPDS) with EMV-51, and Intellec® Series II or III Development Systems.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

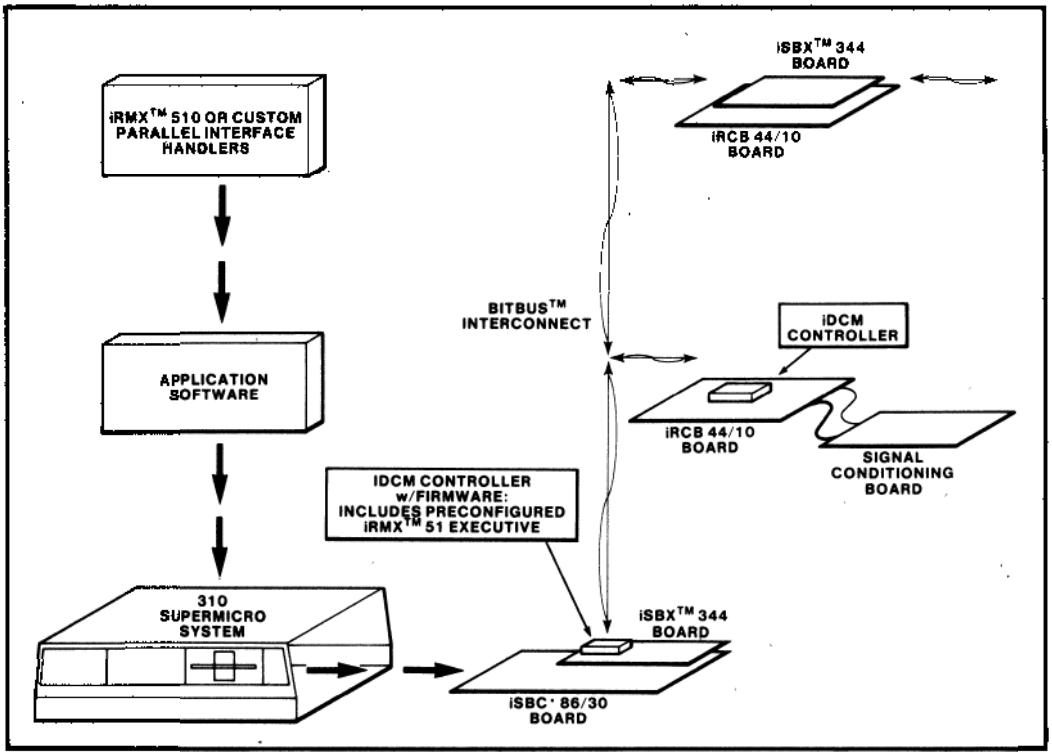


Figure 4. iDCM Operating Environment

SPECIFICATIONS

Supported Hardware

Microcontrollers

8051	80C51
8052	8044
8751	8744
8031	80C31
8032	8344

iDCM Product Line

iSBX 344 MULTIMODULE Board
iRCB 44/10 Remote Controller Board

Compatible Software

iRMX™ 510 iDCM Support Package

Development Tools

ICE™ 51 or ICE 44 Emulators
Intellec Series II or III Development System
iPDS System w/EMV-51
iRMX 510 iDCM Support Package
8051 Software Development Package

Reference Manual (Supplied)

146312-001 — Guide to Using the Distributed Control Modules

Ordering Information

Part Number	Description
iRMX 51BY	Executive for 8051 Family of Microcontrollers with Reference Manual. A, B, and F Media Formats Supplied

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

MCS®-96 SOFTWARE DEVELOPMENT PACKAGES

- **MCS®-96 Software Support Package**
- **PL/M-96 Software Package**

MCS®-96 SOFTWARE SUPPORT PACKAGE

- **Symbolic relocatable assembly language programming for the 8096 microcontroller family**
- **System Utilities for Program Linking and Relocation**
- **Extends Intellec® Microcomputer Development System to support MCS-96 program development**
- **Encourages modular program design for maintainability and reliability**

The MCS®-96 Software Support Package provides development system support for the MCS-96 family of 16-bit single chip microcomputers. The support package includes a macro assembler and system utilities.

The assembler produces relocatable object modules from MCS-96 macro assembly language instructions. The object modules then are linked and located to absolute memory locations.

The assembler and utilities run on the Intellec® Series III or equivalent Microcomputer Development System.

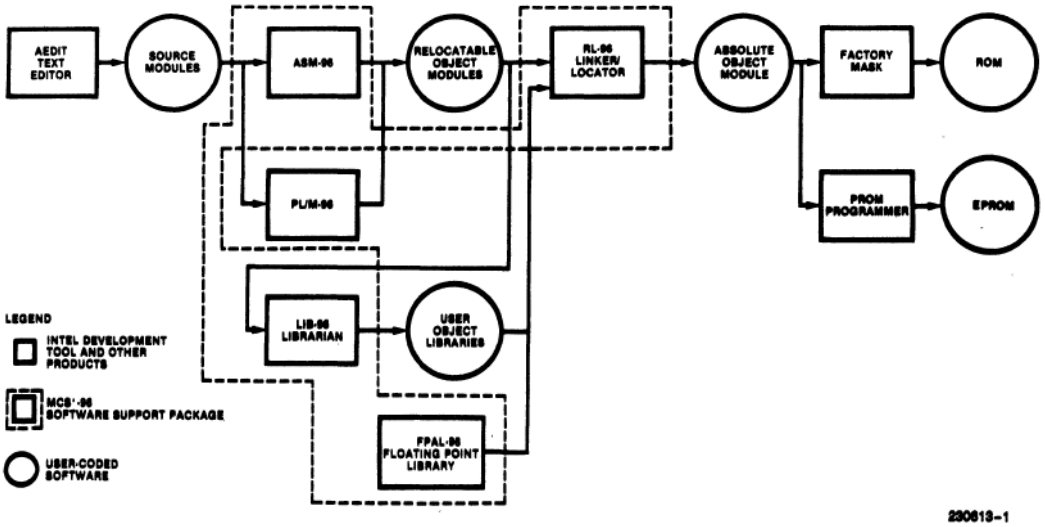


Figure 1. MCS®-96 Software Development Process

Intel Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied. Information contained herein supersedes previously published specifications on these devices from Intel. January 1984
©Intel Corporation, 1984.

Order Number: 230613-003

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8096 MACRO ASSEMBLER

- **Supports 8096 family program development on Intellec® Microcomputer Development System**
- **Gives symbolic access to powerful 8096 hardware features**
- **Object files are linkable and locatable**
- **Symbolic Assembler supports macro capabilities, cross reference, symbol table and conditional assembly**

ASM-96 is the macro assembler for the MCS family of microcontrollers. ASM-96 translates symbolic assembly language mnemonics into relocatable object code. Since the object modules are linkable and locatable, ASM-96 encourages modular programming practices.

The macro facility in ASM-96 allows programmers to save development and maintenance time since common code sequences only have to be done once. The assembler also provides conditional assembly capabilities.

ASM-96 supports symbolic access to the many features of the 8096 architecture. An "include" file is provided with all of the 8096 hardware registers defined. Alternatively, the user can define any subset of the 8096 hardware register set.

Math routines are supported with mnemonics for 16 × 16-bit multiply or 32/16-bit divide instructions.

The assembler runs on a Series III/Series IV Intellec Development Systems for high performance.

RL96 LINKER AND RELOCATOR PROGRAM

- **Links modules generated by ASM-96 and PL/M-96**
- **Locates the linked object module to absolute memory locations**
- **Encourages modular programming for faster program development**
- **Automated selection of required modules from Libraries to satisfy symbolic references**

RL96 is a utility that performs two functions useful in MCS-96 software development:

— The link function which combines a number of MCS-96 object modules into a single program.

— The locate functions which assigns an absolute address to all relocatable addresses in the MCS-96 object module.

RL96 resolves all external symbol references between modules and will select object modules from library files if necessary.

RL96 creates two files:

— The program or absolute object module file that can be executed by the targeted member of the MCS-96 family.

— The listing file that shows the results of link/locate, including a memory map symbol table and an optional cross reference listing.

The relocator allows programmers to concentrate on software functionality and not worry about the absolute addresses of the object code. RL96 promotes modular programming. The application can be broken down into separate modules that are easier to design, test and maintain. Standard modules can be developed and used in different applications thus saving software development time.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FPAL96 FLOATING POINT ARITHMETIC LIBRARY

- **Implements IEEE Floating Point Arithmetic**
- **Basic Arithmetic Operations +, −, ×, /, Mod Plus Square Root**
- **Supports Single Precision 32 Bit Floating Point Variables**
- **Includes an Error Handler Library**

FPAL96 is a library of single precision 32-bit floating point arithmetic functions. All math adheres to the proposed IEEE floating point standard for accuracy and reliability. An error handler to handle exceptions (for example, divide by zero) is included.

The following functions are included:

ADD	NEGATE
SUBTRACT	ABSOLUTE
MULTIPLY	SQUARE ROOT
DIVIDE	INTEGER
COMPARE	REMAINDER

LIB 96

The LIB 96 utility creates and maintains libraries of software object modules. The customer can develop standard modules and place them in libraries. Application programs can then call these modules using predefined interfaces.

LIB 96 uses the following set of commands:

—CREATE: Creates an empty library file.
—ADD: Adds object modules to a library file.
—DELETE: Deletes object modules from a library file.
—LIST: Lists the modules in the library file.
—EXIT: Terminates LIB 96

When using object libraries, RL96 will include only those object modules that are required to satisfy external references, thus saving memory space.

SPECIFICATIONS

Operating Environment

Required Hardware:
Intellec Microcomputer Development System
— Series III/Series IV

Documentation Package:

MCS-96 Macro Assembler User's Guide
MCS-96 Utilities User's Guide
MCS-96 Assembler and Utilities Pocket Reference Card
8096 Floating Point Arithmetic Library

ORDERING INFORMATION

Part Number

iMDX-355
Requires Software License

Description

MCS-96 Software Support Package

SUPPORT:

Hotline Telephone Support, Software Performance Report (SPR), Software Updates, Technical Reports, and Monthly Technical Newsletters are available.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PL/M-96 SOFTWARE PACKAGE

- **High level programming language for the Intel MCS®-96 microcontroller family**
- **Block structured language design encourages module programming**
- **Provides access to MCS®-96 on chip resources**
- **Produces relocatable object code which is linkable to object modules generated by other MCS®-96 translators**
- **Resident on iAPX-86 Intel microcomputer development systems for higher performance**
- **Includes a linking and relocating utility and the library manager**
- **IEEE Floating Point Library included for numeric support**
- **Compatible with PL/M-86 assuring design portability**

PL/M-96 is a structured, high-level programming language useful for developing software for the Intel MCS-96 family of microcontrollers. PL/M-96 was designed to support the software requirements of advanced 16 bit microcontrollers. Access to the on chip resources of the MCS-96 has been provided in PL/M-96.

PL/M-96 is compatible with PL/M-86. Programmers familiar with PL/M will find they can program in PL/M-96 with little relearning effort.

The PL/M-96 compiler translates PL/M-96 high level language statements into MCS-96 machine instructions. By programming in PL/M an engineer can be more productive in the initial software development cycle of the project. PL/M can also reduce future maintenance and support cost because PL/M programs are easier to understand. PL/M-96 was designed to complement Intel's ASM-96.

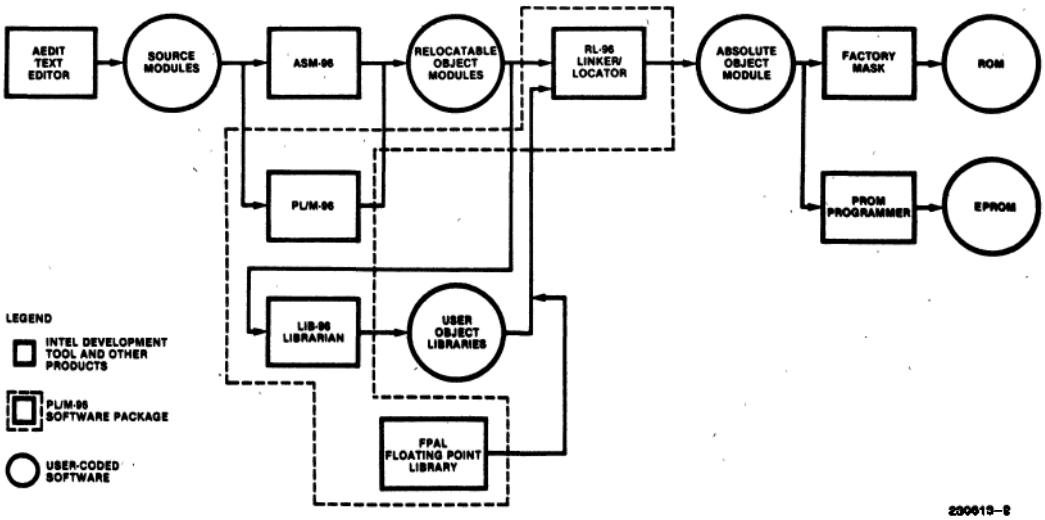


Figure 2. MCS®-96 Software Development Process

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PL/M-96 COMPILER

FEATURES

Major features of the PL/M-96 compiler and programming language include:

Structured Programming

Programs written in PL/M-96 are developed as a collection of procedures, modules and blocks. Structured programs are easier to understand, maintain and debug. PL/M-96 programs can be made more reliable by clearly defining the scope of user variables (for example, local variables in a procedure). REENTRANT procedures are also supported by PL/M-96.

Language Compatibility

PL/M-96 object modules are compatible with all other object modules generated by Intel MCS-96 translators. Programmers may choose to link ASM-96 and PL/M-96 object modules together.

PL/M-96 object modules were designed to work with other Intel support tools for the MCS-96. The DEBUG compiler control provides these tools with symbolic information.

Data Types Supported

PL/M-96 supports seven data types for programmer flexibility in various logical, arithmetic and addressing functions. The seven data types include:

—BYTE:	8-bit unsigned number
—WORD:	16-bit unsigned number
—DWORD:	32-bit unsigned number
—SHORTINT:	8-bit signed number
—INTEGER:	16-bit signed number
—LONGINT:	32-bit signed number
—REAL:	32-bit floating point number

Another powerful feature are BASED variables. BASED variables allow the user to map more than one variable to the same memory location. This is especially useful for passing parameters, relative and absolute addressing, and memory allocation.

Data Structures Supported

Two data structuring facilities are supported by PL/M-96. The user can organize data into logical groups. This adds flexibility in referencing data.

— Array: Indexed list of same type data elements

— Structure: Named collection of same or different type data elements

— Combinations of Both: Arrays of structures or structures of arrays

Interrupt Handling

Interrupts are supported in PL/M-96 by defining a procedure with the INTERRUPT attribute. The compiler will generate code to save and restore the program status word when handling hardware interrupts of the MCS-96.

Compiler Controls

Compile time options increase the flexibility of the PL/M-96 compiler. These controls include:

— Optimization

— Conditional compilation

— The inclusion of common PL/M-96 source files from disk

— Cross reference of symbols

— Optional assembly language code in the listing file

230613-003

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Code Optimizations

The PL/M-96 compilers has four levels of optimization for reducing program size.

— Combination of constant expressions; "Strength reductions" (e.g.: a shift left rather than multiply by two)

— Machine code optimizations; elimination of superfluous branches; reuse of duplicate code, removal of unreachable code

— Overlaying of on chip RAM variables

— Optimization of based variable operations

— Use of short jumps where possible

Built in Functions

An extensive list of built in functions has been supplied as part of the PL/M-96 language. Besides TYPE CONVERSION functions, there are built in functions for STRING manipulations. Functions are provided for interrogating the MCS-96 hardware flags such as CARRY and OVERFLOW.

Error Checking

If the PL/M-96 compiler detects a programming or compilation error, a fully detailed error message is provided by the compiler. If a syntax or program error is detected, the compiler will skip the code generation and optimization passes. This powerful PL/M-96 feature can yield a two times increase in throughput when a user is in the initial program development cycle.

BENEFITS

PLM-96 is designed to be an efficient, cost-effective solution to the special requirements of MCS-96 Microcontroller Software Development, as illustrated by the following benefits of PL/M use:

Low Learning Effort

PL/M-96 is easy to learn and to use, even for the novice programmer.

Earlier Project Completion

Critical projects are completed much earlier than otherwise possible because PL/M-96, a structured high-level language, increases programmer productivity.

Lower Development Cost

Increases in programmer productivity translate immediately into lower software development costs because less programming resources are required for a given programmed function.

Increased Reliability

PL/M-96 is designed to aid in the development of reliable software (PL/M programs are simple statements of the program algorithm). This substantially reduces the risk of costly correction of errors in systems that have already reached full production status. The more simply the program is stated, the more likely it is to perform its intended function.

Easier Enhancements and Maintenance

Programs written in PL/M tend to be self-documenting, thus easier to read and understand. This means it is easier to enhance and maintain PL/M programs as the system capabilities expand and future products are developed.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

RL96 LINKER AND RELOCATOR PROGRAM

- **Links modules generated by ASM-96 and PL/M-96**
- **Locates the linked object module to absolute memory locations**
- **Encourages modular programming for faster program development**
- **Automated selection of required modules from Libraries to satisfy symbolic references**

RL96 is a utility that performs two functions useful in MCS software development:

— The link function which combines a number of MCS object modules into a single program.

— The locate function which assigns an obsolute address to all relocatable addresses in the MCS-96 object module.

RL96 resolves all external symbol references between modules and will select object modules from library files if necessary.

RL96 creates two files:

— The program or absolute object module file that can be executed by the targeted member of the MCS family.

— The listing file that shows the results of link/locate, including a memory map symbol table and an optional cross reference listing.

The relocator allows programmers to concentrate on software functionality and not worry about the absolute addresses of the object code. RL96 promotes modular programming. The application can be broken down into separate modules that are easier to design, test and maintain. Standard modules can be developed and used in different applications thus saving software development time.

FPAL96 FLOATING POINT ARITHMETIC LIBRARY

- **Implements IEEE Floating Point Arithmetic**
- **Basic Arithmetic Operations +, −, ×, /, Mod Plus Square Root**
- **Supports Single Precision 32 Bit Floating Point Variables**
- **Includes an Error Handler Library**

FPAL96 is a library of single precision 32-bit floating point arithmetic functions. All math adheres to the proposed IEEE floating point standard for accuracy and reliability. An error handler to handle exceptions (for example, divide by zero) is included.

The following functions are included:

ADD	NEGATE
SUBTRACT	ABSOLUTE
MULTIPLY	SQUARE ROOT
DIVIDE	INTEGER
COMPARE	REMAINDER

230613-003

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

LIB 96

The LIB 96 utility creates and maintains libraries of software object modules. The customer can develop standard modules and place them in libraries. Application programs can then call these modules using predefined interfaces.

LIB 96 uses the following set of commands:

—CREATE: Creates an empty library file
—ADD: Adds object modules to a library file
—DELETE: Deletes object modules from a library file
—LIST: Lists the modules in the library file
—EXIT: Terminates LIB 96

When using object libraries, RL96 will include only those object modules that are required to satisfy external references, thus saving memory space.

SPECIFICATIONS

Operating Environment

Required Hardware:
Intellec Microcomputer Development System
— Series III/Series IV

Documentation Package:

PL/M-96 User's Guide
MCS-96 Utilities User's Guide
MCS-96 Assembler and Utilities Pocket Reference Card
8096 Floating Point Arithmetic Library

ORDERING INFORMATION

Part Number

iMDX-356
Requires Software License

Description

PL/M-96 Software Package

SUPPORT:

Hotline Telephone Support, Software Performance Report (SPR), Software Updates, Technical Reports, and Monthly Technical Newsletters are available.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Development Productivity Tools 4

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

DEVELOPMENT PRODUCTIVITY TOOLS

INTRODUCTION

Improving an engineering team's productivity is a never ending task in today's competitive environments. Intel offers software tools and communication systems that optimize the usage of expensive engineering personnel and capital equipment. Software tools boost a programming team's productivity, thereby lowering development costs and shortening product development times. Communication software provides further productivity gains by linking multi-computer engineering environments into highly effective networks.

One software tool that substantially increases software productivity is PSCOPE, a source level symbolic debugger. The PSCOPE debugger allows the high-level language programmer to completely debug his code at the same level at which it was written. Breakpointing, tracing, and patching are all done in a faster and less error-prone manner than through obsolete machine-level debuggers. As software testing and maintenance consume a greater portion of development life-cycle time and cost, PSCOPE debugging can significantly improve programming efficiency.

Another set of valuable software tools are Intel's Program Management Tools (PMTs), which provide the essential ingredients to manage large software development projects. PMTs decrease the time spent on tracking program changes and manually generating new systems, thereby giving engineers more time for software design, development, and testing. PMTs consist of a Software Version Control System (SVCS), and an automated software generation facility (MAKE). Together these tools control, examine, and automate the management of a software system that may contain many versions consisting of numerous modules.

Intel's software toolboxes are collections of utilities that perform a variety of productivity-oriented functions. The ISIS-II Software Toolbox offers conditional submit file control tools, source management tools, and other tools that operate at the ISIS-II command level. The 8086 Software Toolbox is a collection of 16-bit software tools that are valuable for text formatting and preparation, software testing and performance analysis, 286/287 software development, and a multitude of other applications.

Intel also offers AEDIT, an advanced editor that significantly improves programmer productivity. AEDIT was designed with the programmer in mind, and offers full screen editing, the ability to edit two files at once, features for manipulating large blocks of text, and dynamic macro command definition.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PSCOPE
HIGH-LEVEL PROGRAM DEBUGGER

- **Source-Level Debugging for High Productivity**
- **Breakpoint, Single-Step and Execution Trace by Statement Numbers, Procedure Names and Labels**
- **High-Level Code Patching**
- **Compatible with Intel's I²ICE™ Integrated Instrumentation and In-Circuit Emulation System for Target System Debugging**
- **Native CPU Execution for iAPX 88 and 86 Architectures**
- **Supports PL/M, Pascal, and FORTRAN Program Debugging**

PSCOPE is an interactive, symbolic debugger for high-level language programs. It allows users to scrutinize program execution at the source level, using high-level statement numbers, procedure and variable names and labels. This is typically a more productive way of debugging high-level language (HLL) programs than at the machine level.

Source-level debugging means that traditional functions, such as setting breakpoints or tracing execution flow, are more powerful in PSCOPE. For example, tracing procedure entry (or exit) points conveys much more information than tracing machine instructions. Single-step execution is more powerful, using statements and procedures, as well.

The productivity improvement from debugging in a high-level language is analogous to programming in a high-level language, when compared to assembly-level programming and debugging.

PSCOPE users may define high-level code patches, which are "compiled" and patched into the user's program. Code patches may be stored on disk, so they may be later incorporated into the program source file.

PSCOPE is an integral part of the advanced I²ICE Integrated Instrumentation and In-Circuit Emulation System. This allows a smooth migration from *program* debugging to *target system* debugging.

PSCOPE's symbol capacity is virtually unlimited. Symbols are paged to disk when necessary.

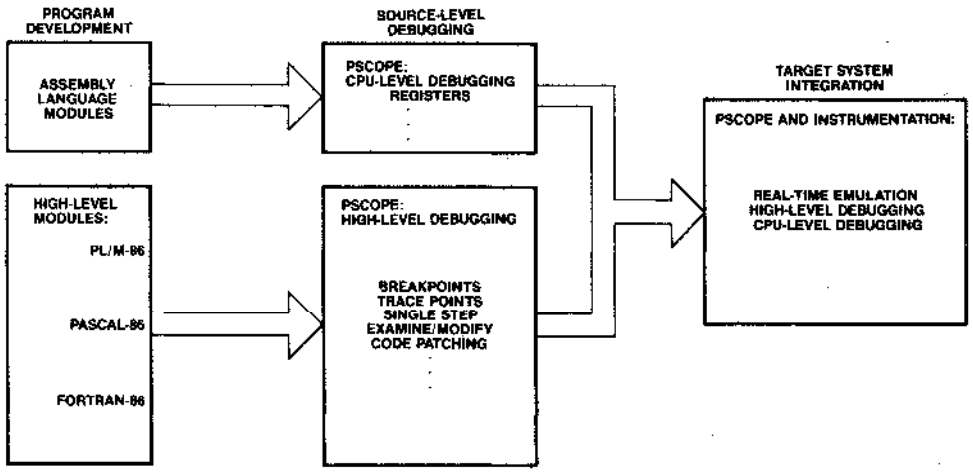


Figure 1. Debugging Methodology with PSCOPE

MAY 1983
ORDER NUMBER:210350-003

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SAMPLE DEBUGGER SESSION

```
SERIES-III Pascal-86, V1.1


Source File: :F2:MAXMIN.PAS
Object File: :F2:MAXMIN.OBJ
Controls Specified:  DEBUG.

STMT LINE NESTING       SOURCE TEXT: :F2:MAXMIN.PAS
   1    1  0  0         program calc(input,output);
   2    2  0  0         var a,b:integer;

   3    4  0  0          procedure sum(x,y:integer);
   4    5  1  0          var z:integer;
   5    6  1  0          begin
   6    7  1  1           z:=x+y;
   7    8  1  1           writeln('The sum is ',z);
   8    9  1  1          end;

   9   11  0  0           procedure difference(x,y:integer);
  10   12  1  0           var z:integer;
  11   13  1  0           begin
  12   14  1  1            z:=abs(x-y);
  13   15  1  1            writeln('The difference is ',z);
  14   16  1  1           end;

  15   18  0  0           procedure maxmin(x,y:integer);
  16   19  1  0           begin
  17   20  1  1            if x<y then writeln('The maximum is ',y,
                                                '   The minimum is ',x);
  18   21  1  1            if y<x then writeln('The maximum is ',x,
                                                '   The minimum is ',y);
  19   22  1  1            if x=y then writeln ('The two inputs are equivalent ');
  20   23  1  1           end;

  21   25  0  0         begin
  22   26  0  1           repeat (*forever*)
  23   27  0  2            write('Input two integers ');
  24   28  0  2            readln(a,b);

  25   30  0  2            sum(a,b);
  26   31  0  2            difference(a,b);
  27   32  0  2            maxmin(a,b);
  28   33  0  2           until 1<0
  29   34  0  2         end.
```

The program listing for the sample PSCOPE session illustrates the high-level nature of PSCOPE debugging. The program consists of the module CALC, the procedures SUM, DIFFERENCE, and MAXMIN, plus global and local variables. Users exercise and manipulate the program using these symbols. Code patches, stepping, tracing, etc. are all done on line numbers, procedures, labels, and symbolic names. To debug a program, just PSCOPE and a listing are required—no linkage maps, core dumps, locate maps, etc. are necessary. This is how high-level debugging relates to high-level programming.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FEATURES

Unlimited Breakpoints

Breakpoints may be set on statement numbers, procedure names, or program labels. Any number of breakpoint registers may be defined.

High-Level Trace Points

Execution trace points are defined the same way as program breaks. Any number of trace points may be defined. A trace message is displayed when execution reaches a trace point.

Conditional Break and Trace

Any break or trace point may be defined to automatically call a *debugger procedure,* which will execute PSCOPE commands and/or evaluate predefined conditions. The operations will be performed, and the condition will determine if the break or trace will be done.

GO

The GO command initiates program execution from any starting point. A set of stopping points may be specified ("GO TIL"), and break/trace registers may be used ("GO USING").

Source-Level Stepping

A program may be executed, one high-level statement at a time, using the LSTEP command. Also, entire procedures may be treated as single statements during stepping (PSTEP); the procedures will be executed, but not stepped through.

Examine/Modify Data

PSCOPE allows users to symbolically examine (and change the value of) program variables and data structures. All PL/M and Pascal types are supported, including numerics, dynamic and stack variables, arrays, and fields within structures.

Virtual Symbol Table

All user-program symbols are stored in a virtual symbol table. This means symbols will be paged to disk, if necessary.

Help File

Many PSCOPE commands, facilities, and error messages have help information describing their use. The HELP command is used for learning the PSCOPE command language, for quick reference of command syntax, and for learning the cause of command errors.

Debugger Procedures

PSCOPE has the facility for defining procedures in its command language. This block-structured command language allows users to *extend* the capability of the program under debug. Like macros with parameters, these procedures may also be used for generating compound and conditional debugger commands.

Code Patching

Program patches may be written in the debugger command language to augment or replace current program statements. These high-level code patches are much closer to actual program changes than machine-level patches, and are easy to use.

Built-in Editor

A menu-driven, CRT-oriented editor is built into PSCOPE. This is used for creating and editing program patches, debugger procedures, and command lines. One key is used to invoke the editor to alter the last command entered, or any debugger definition (literally, trace register, patch, etc.) may be edited selectively.

Debugger Command Language

GO/LSTEP/PSTEP—For controlling program execution.

DEFINE/DISPLAY/MODIFY/REMOVE—For manipulating debugger objects (such as break registers, patches, and procedures), or program objects (variables and data structures).

CALL/RETURN—For executing debugger procedures.

WRITE/CI—For console input and output.

DO/END—For defining command blocks.

REPEAT/COUNT—For repetition of commands or blocks.

IF/THEN/ELSE—For conditional execution of commands or blocks.

INCLUDE/PUT/APPEND—For saving/restoring commands and definitions to and from disk.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

BENEFITS

Shortened Development Cycle

The ability to define debugger procedures and make code patches is very useful. It actually allows users to *extend* the capability of the program under debug. After debug sessions, users typically make program changes or enhancements. This involves the use of an editor, compiler and linkage tools that create a "new" load module for debugging. Since PSCOPE allows these changes and enhancements to be made *in the debugger*, the number of Edit/Compile/Link iterations is lowered. More confidence can be placed on a program during debugging, because its capabilities have been more fully exercised.

Improved Debugging Productivity

PSCOPE provides users with the same conceptual interface to program debugging that was used in program design. This includes the high-level language constructs such as statements, procedures, labels and symbolic variables and data structures. Functions such as program trace and single-step execution are more meaningful with statements and procedures than machine instructions; therefore the improvement in debugging productivity is analogous to the programming productivity using high-level languages.

More Reliable Software

Debugger procedures may be used to automate the software testing process. The procedure may repeatedly generate test values, execute the program with the input values, and record the results. Running more comprehensive tests, plus being able to "batch" the tests, yields more reliable software.

Easy to Learn and Use

An extensive command language, which is similar to block-structured languages such as PL/M and Pascal, is very easy to use in an interactive debug session. The HELP facility makes learning to use PSCOPE extremely fast as well. The "Literally" facility and debugger procedures also allow users to extend and tailor the command language to suit individual needs.

Improved Software Management

The use of debugger procedures allows parts of a software system to be debugged independently. Procedures can be substituted for program stubs, allowing programmers to debug different pieces of the system separately. This results in improved project management.

SPECIFICATIONS

Supports Intel's standard 86/88 languages:

—PL/M 86/88
—Pascal 86/88
—FORTRAN 86/88

PSCOPE runs on an Intellec® Series III or Series IV Microcomputer Development System, either standalone or in an NDS-II network configuration. A 512K application memory space is recommended for most applications.

ORDERING INFORMATION

Order Code	Description
iMDX-333	PSCOPE Program Debugger (for Series III and Series IV)
III-951A	PSCOPE Program Debugger and I^2ICE Base Software for Series III with 8" single density disk drive
III-951B	PSCOPE Program Debugger and I^2ICE Base Software for Series III with 8" double density disk drive
III-951C	PSCOPE Program Debugger and I^2ICE Base Software for Series IV with 5¼" double density disk drive

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
-run :f1:pscope
SERIES-III PSCOPE-86, V1.0
*
*
*define literally d = 'define'
*d literally l = 'literally'
*d l br = 'brkreg'
*d l tr = 'trcreg'
*
*
*load :f1:maxmin.86
*dir
DIR of :CALC
PO_OUTPUT . . . . . TEXT (file)
PO_INPUT . . . . . TEXT (file)
B . . . . . integer
A . . . . . integer
SUM . . . . . procedure
  X . . . . . integer
  Y . . . . . integer
  Z . . . . . integer
DIFFERENCE . . . . . procedure
  X . . . . . integer
  Y . . . . . integer
  Z . . . . . integer
MAXMIN . . . . . procedure
  X . . . . . integer
  Y . . . . . integer
*
*
*pstep
[Step at :CALC#21]
*pstep
            INPUT TWO INTEGERS:
[Step at :CALC#22]
*pstep
                (input)   19  4
[Step at :CALC#23]
*pstep
                THE SUM IS   76
[Step at :CALC#24]
*pstep
          THE DIFFERENCE IS   15
[Step at :CALC#25]
*pstep
             THE MAXIMUM IS   19
             THE MINIMUM IS    4
[Step at :CALC#21]
*
*
*define patch #5 til #6 =  z=x+y
*
*go til #21
            INPUT TWO INTEGERS:
                (input)   19  4
                THE SUM IS   23
          THE DIFFERENCE IS   15
             THE MAXIMUM IS   19
             THE MINIMUM IS    4
[break at #21]
*
*
*define proc PR1 = do
.*write 'the numbers and product are: ',a,b,a*b
.*write using ('@,>') 'break ? '
.*if CI == 'y' then return true
..*            else return false endif
.*end
*
*d br B3 = #21 call PR1
*go using b3
            INPUT TWO INTEGERS:
                (input)   23 24
                THE SUM IS   47
          THE DIFFERENCE IS    1
             THE MAXIMUM IS   24
             THE MINIMUM IS   24
the numbers and the product are:  +23 +24 +552
break ? y
[break at #21]
*
*
*exit
PSCOPE terminated
```

The Literally facility allows users to abbreviate, redefine and extend the command language to suit individual needs.

Any PL/M-86, Pascal-86 or FORTRAN-86 program may be loaded. All symbolic names may be displayed, in total or by type. Symbols defined at debug time may be displayed as well. All program types are supported, including numerics, user-defined types, and records. The symbols' types are displayed by the DIR command as well.

Several flavors of stepping are offered. This example illustrates PSTEP, a line-by-line step where procedures are executed as a single step. This program contains five steps in the main body, with three being procedure calls.

There appears to be a bug in the program, as the sum is displayed incorrectly. Looking at the program, we notice that X and Y were multiplied instead of added, at line #5. A code patch is defined, and the program executes correctly.

This illustrates the facility where a *debug procedure* (PR1) is called when reaching a breakpoint at line #21. Here, some values are displayed, and a condition is evaluated (in this case, a query to the user). Had the condition been false, program execution would continue with no break. The high-level constructs in the command language make this a very powerful facility.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

PROGRAM MANAGEMENT TOOLS

- **Increase Software Engineering Productivity**
- **Decrease Software Administration Overhead**
- **Allow Users to Control, Automate and Examine the Evolution of a Software Project**
- **Enhance the Capability of Networked (NDS-II) and Standalone Development Systems**
- **SVCS Simplifies Administration of Software Modules and Systems**
- **MAKE Automatically Generates New Releases of Software Systems**
- **Both Tools Easily Incorporated Into Existing Software Development Methodologies**

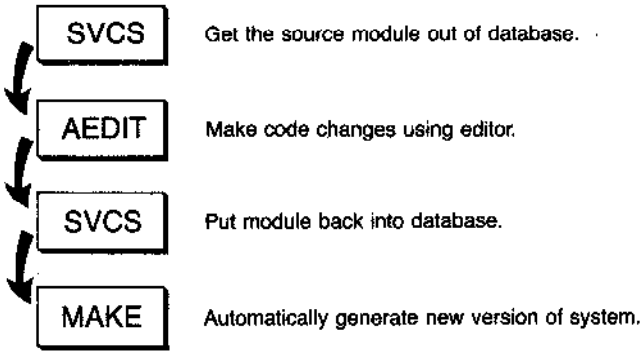
Intel's Program Management Tools (PMTs) provide the essential ingredients to manage large software development projects. PMTs decrease the time spent on tracking program changes and manually generating new systems, thereby giving engineers more time for software design, development, and testing.

PMTs consist of a "Software Version Control System" (SVCS), and an automated software generation facility (MAKE). Together these tools control, examine, and automate the management of a software system that may contain many versions consisting of numerous modules.

SVCS controls and documents software changes for all file types. SVCS handles storage and retrieval of different versions of a given module, controls update privileges, prevents different users from making changes independently, and requires all changes be thoroughly documented by recording who made what changes, when and why.

MAKE produces the specification of a "minimum-work" job required to generate a new system. This job (i.e. submit file) typically includes compiles and links of the latest versions of specified source and object modules. If a newer source module exists for any specified object module, MAKE will specify a compile of this module, replacing the older module in the completed program. Unnecessary links and compiles, however, are eliminated. MAKE does the minimum work required to ensure consistent, up-to-date software, thus saving many hours of compiles and links.

Incorporating PMTs into an existing project is easy. PMTs work with existing operating systems and software tools (editors, compilers, utilities) and require very little relearning. New users can quickly gain expertise in using PMTs by working through the examples contained in the PMT Tutorial Manual and Diskette, which are included with every PMT software package. Program Management Tools are ideal in a networked (NDS-II) environment, where multi-version software control is critical. PMTs are also extremely valuable on standalone systems (with Winchester disk) as well.



OPTIMAL CONTROL OF A SOFTWARE PROJECT.

Intel Corporation Assumes No Responsibility for the Use of Any Circuitry Other Than Circuitry Embodied in an Intel Product. No Other Circuit Patent Licenses are Implied. Information Contained Herein Supercedes Previously Published Specifications On These Devices From Intel.
© INTEL CORPORATION, 1983

MARCH 1984
ORDER NUMBER: 210667-003

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

SOFTWARE VERSION CONTROL SYSTEM (SVCS)

- **Simplifies Administration of Software Modules and Systems**
- **Maintains Change History Information on Every Module**
- **Prevents Users From Accidently Deleting System Software or Making Simultaneous Module Changes**
- **Offers an Effective Software Version Generation and Control Mechanism**

Intel's Software Version Control System (SVCS) is a utility that greatly simplifies software system housekeeping. SVCS automatically controls and documents software modules in a large project, eliminating costly manual administration by a project leader or librarian.

SVCS maintains a system database of software modules called units. Each unit is divided into four classes: Source, which contains the unit's source code; Object, which contains the unit's object code; History, which contains the unit's history file; and Composition, which can be arbitrarily used by the user.

Users interact with the database by using SVCS administrative and access commands. Project managers use administrative commands to create new system databases, add and delete database units, set unit access rights, and create and name new system variants. Programmers use SVCS access commands to check out and return database modules when making system changes. For every change made, SVCS records *what* changed, *who* changed it, *when* it was changed, and *why*.

SVCS variant generation and control enable project administrators to effectively create and identify new versions of software systems. Stable versions may be write protected and placed in the public domain, working versions may be identified and accessible only to programming personnel, and special versions may be created for customized releases. In addition, version control can minimize software archival, maintenance, and support administrative overhead.

AUTOMATED SOFTWARE GENERATION (MAKE)

- **Automatically Creates New Software Systems, Using the Latest Versions of Source Modules**
- **Automatically Determines Which Source Modules Need Recompiling**
- **Eliminates Unnecessary Compiles and Links**
- **Works Closely with SVCS for Generating Complete, Up-To-Date Systems**
- **Easily Adopted into Existing Development Methodologies**
- **Offers Many Powerful Macro Constructs**

MAKE is a utility that greatly simplifies the generation of software systems. MAKE produces a "minimum-work" submit file that can generate a complete, up-to-date system without any unnecessary compiles and links. MAKE can reduce system generation times from hours to minutes while concurrently minimizing administrative overhead.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

MAKE accepts a text input file that instructs it how to generate a new software system. The input file specifies all modules required to generate the new system and includes a description of system dependencies. It also specifies specific system operations, such as compiles, links, SVCS operations, line-printer spoolings, and other system commands. MAKE uses this input file in conjunction with the time and date stamps on each module to determine the optimum system generation procedure that eliminates all unnecessary compiles and links.

Typically a MAKE input file is created once at the start of a project. Very occasionally during the life of the project it may need modification. A powerful set of macros makes the creation and subsequent modification of a generation procedure an easy task. Overall, the management of the MAKE input file is negligible compared to maintaining numerous submit files for system generations.

The close relationship between SVCS and MAKE help simplify the overall job of software control at all levels. For example, the very latest version of a source module may not be stable enough to be included in a generation. A less functional, but more reliable version may exist. Since SVCS keeps unique versions distinct, an SVCS-module containing the more reliable version may be specified in the MAKE input file.

BENEFITS: SVCS AND MAKE

Intel's Program Management Tools eliminate common problems such as:

"We've modified module F00, which has introduced a new set of problems. Now we can't restore it back to the earlier version."

"Module F002 has been modified; no one seems to know who changed it, or why."

"We often have several programmers making changes to the same modules. Trying to avoid simultaneous changes is a lot of effort, and we waste time synthesizing two sets of changes into one module."

"To ensure that we release up-to-date, correct software, we periodically go into "release mode" for a few days. Everyone stops work completely while we find the latest versions, and then start the generation from the ground up. It literally takes days, when we could be making productive changes."

SVCS and MAKE together provide a service that fits easily into your existing design methodology, and solves administrative problems such as those described above.

SPECIFICATIONS

Networked, Multi-User Software Control

NDS-II with at least one Intellec Microcomputer Development System
iNDX, ISIS-III(N) System Software

Standalone Use

Intellec Series III with Model 750 Winchester Disk or Intellec Series IV
SVCS and MAKE will not operate on ISIS-II local floppies or Model 740 Hard Disks.
SVCS and MAKE may be exported from any workstation in an NDS-II configuration.

Documentation

"A User's Guide to Program Management Tools" (121958)

SOFTWARE SUPPORT

This product includes a 90-day initial support consisting of new software releases, updates, subscription services (software performance reports and technical reports), and telephone hotline support. Additional software support services are available separately.

ORDERING INFORMATION

Part Number	Description
iMDX-332	Intel Program Management Tools

210567-003

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

ISIS-II SOFTWARE TOOLBOX

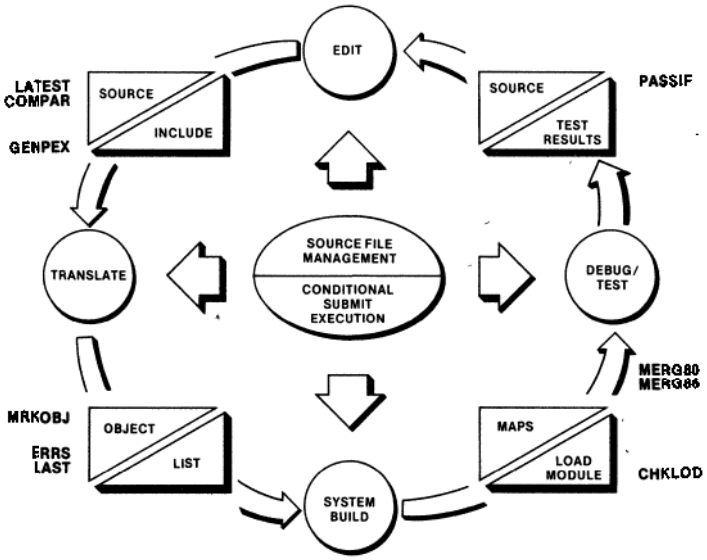
- **Significantly Improves Programmer Productivity**
- **Collection of Utilities that Speed Up Software Design**
- **Enhances Capabilities of ISIS-II Operating System**
- **Most Utilities will Operate on NDS-I Workstations, and Remote Hard Disks**
- **Provides Source File Management, Showing Source Changes, and Performing Version Control**
- **Provides Conditional Control and "Structured Programming" to Submit Files**
- **Runs on Model 800, Series II, and Series III Intellec® Development Systems**

The ISIS-II Software Toolbox is a collection of system utilities that perform a variety of "productivity-oriented" functions. There are two major subsets of Toolbox tools, in addition to numerous ad hoc utilities. These subsets provide Conditional Submit File Control and Source File Management.

The Conditional Submit File Control tools provide "structured programming" at the ISIS-II command level. Jumps, Calls, Returns, etc. are supported, as well as conditional command execution, based on assertions such as file existence, program errors, file matching, and string matching.

The Source Management Tools support version number tracking, and allow users to identify which versions of each source module were used to create a load module. There is also a tool which compares source files and reports all differences.

The tools outside of the two major subsets assist the programmer in some very specific development and debugging tasks. One tool manages all PUBLIC/EXTERNAL declarations in a system. Another merges the locate maps into a program listing, giving absolute symbolic debugging information. There's a directory sorter, a file compactor, and a tool to display just the last block of a file.



MANY TOOLS IN THE TOOLBOX ENHANCE SPECIFIC PHASES OF THE DEVELOPMENT CYCLE. OTHERS IMPROVE PRODUCTIVITY IN ALL PHASES.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP