MODEL 230
INTELLEC SERIES II
MICROCOMPUTER DEVELOPMENT SYSTEM

Complete microcomputer development center for Intel MCS-86, MCS-80, MCS-85 and MCS-48 microprocessor families

LSI electronics board with CPU, RAM, ROM, I/O, and interrupt circuitry

64K bytes RAM memory

Self-test diagnostic capability

Eight-level nested, maskable priority interrupt system

Built-in interfaces for high speed paper tape reader/punch, printer, and universal PROM programmer

Integral CRT with detachable upper/lower case typewriter-style full ASCII keyboard

Powerful ISIS-II Diskette Operating System software with relocating macroassembler, linker, and locater

1 million bytes (expandable to 2.5M bytes) of diskette storage

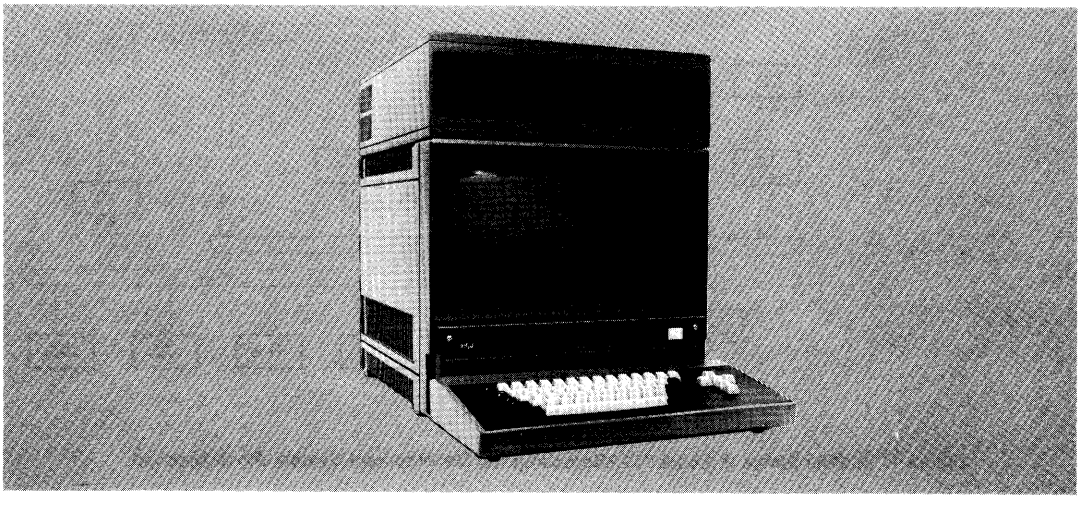
Supports PL/M and FORTRAN high level languages

Standard MULTIBUS with multiprocessor and DMA capability

Compatible with standard Intellec/iSBC expansion modules

Software compatible with previous Intellec systems

The Model 230 Intellec Series II Microcomputer Development System is a complete center for the development of microcomputer-based products. It includes a CPU, 64K bytes of RAM, 4K bytes of ROM memory, a 2000-character CRT, a detachable full ASCII keyboard, and dual double density diskette drives providing over 1 million bytes of on-line data storage. Powerful ISIS-II Diskette Operating System software allows the Model 230 to be used quickly and efficiently for assembling and/or compiling and debugging programs for Intel's MCS-86, MCS-80, MCS-85, or MCS-48 microprocessor families without the need for handling paper tape. ISIS-II performs all file handling operations, leaving the user free to concentrate on the details of his own application. When used in conjunction with an optional in-circuit emulator (ICE) module, the Model 230 provides all the hardware and software development tools necessary for the rapid development of a microcomputer-based product.



FUNCTIONAL DESCRIPTION

Hardware Components

The Intellec Series II Model 230 is a packaged, highly integrated microcomputer development system consisting of a CRT chassis with a 6-slot cardcage, power supply, fans, cables, and five printed circuit cards. A separate, full ASCII keyboard is connected with a cable. A second chassis contains two floppy disk drives capable of double-density operation along with a separate power supply, fans, and cables for connection to the main chassis. A block diagram of the Model 230 is shown in Figure 1.

CPU Cards — The master CPU card contains its own microprocessor, memory, I/O, interrupt and bus interface circuitry fashioned from Intel's high technology LSI components. Known as the integrated processor board (IPB), it occupies the first slot in the cardcage. A second slave CPU card is responsible for all remaining I/O control including the CRT and keyboard interface. This card, mounted on the rear panel, also contains its own microprocessor, RAM and ROM memory, and I/O interface logic, thus, in effect, creating a dual processor environment. Known as the I/O controller (IOC), the slave CPU card communicates with the IPB over an 8-bit bidirectional data bus.

Memory and Control Cards — In addition, 32K bytes of RAM (bringing the total to 64K bytes) is located on a separate card in the main cardcage. Fabricated from Intel's 16K RAMs, the board also contains all necessary address decoding and refresh logic. Two additional boards in the cardcage are used to control the two double-density floppy disk drives.

Expansion — Two remaining slots in the cardcage are available for system expansion. Additional expansion of 4 slots can be achieved through the addition of an Intellec Series II expansion chassis.

System Components

The heart of the IPB is an Intel NMOS 8-bit microprocessor, the 8080A-2, running at 2.6 MHz. 32K bytes of RAM memory are provided on the board using Intel 16K RAMs. 4K of ROM is provided, preprogrammed with system bootstrap "self-test" diagnostics and the Intellec Series II System Monitor. The eight-level vectored priority interrupt system allows interrupts to be individually masked. Using Intel's versatile 8259A interrupt controller, the interrupt system may be user programmed to respond to individual needs.

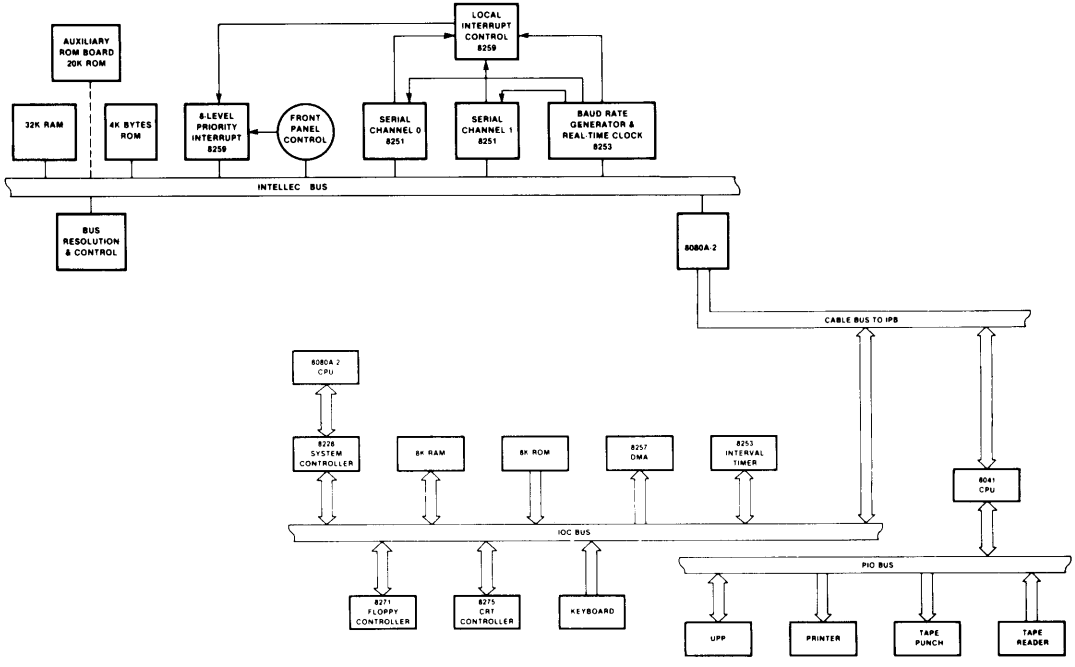


Figure 1. Intellec Series II Model 230 Microcomputer Development System Block Diagram

Input/Output

IPB Serial Channels — The I/O subsystem in the Model 230 consists of two parts: the IOC card and two serial channels on the IPB itself. Each serial channel is RS232 compatible and is capable of running asynchronously from 110 to 9600 baud or synchronously from 150 to 56K baud. Both may be connected to a user defined data set or terminal. One channel contains current loop adapters. Both channels are implemented using Intel's 8251A USART. They can be programmatically selected to perform a variety of I/O functions. Baud rate selection is accomplished programmatically through an Intel 8253 interval timer. The 8253 also serves as a real-time clock for the entire system. I/O activity through both serial channels is signaled to the system through a second 8259 interrupt controller, operating in a polled mode nested to the primary 8259.

IOC Interface — The remainder of system I/O activity takes place in the IOC. The IOC provides interface for the CRT, keyboard, and standard Intellec peripherals including printer, high speed paper tape reader/punch, and universal PROM programmer. The IOC contains its own independent microprocessor, also an 8080A-2. The CPU controls all I/O operations as well as supervising communications with the IPB. 8K bytes of ROM contain all I/O control firmware. 8K bytes of RAM are used for CRT screen refresh storage. These do not occupy space in Intellec Series II main memory since the IOC is a totally independent microcomputer subsystem.

Integral CRT

Display — The CRT is a 12-inch raster scan type monitor with a 50/60 Hz vertical scan rate and 15.5 kHz horizontal scan rate. Controls are provided for brightness and contrast adjustments. The interface to the CRT is provided through an Intel 8275 single chip programmable CRT controller. The master processor on the IPB transfers a character for display to the IOC, where it is stored in RAM. The CRT controller reads a line at a time into its line buffer through an Intel 8257 DMA controller and then feeds one character at a time to the character generator to produce the video signal. Timing for the CRT control is provided by an Intel 8253 interval timer. The screen display is formatted as 25 rows of 80 characters. The full set of ASCII characters are displayed, including lower case alphas.

Keyboard — The keyboard interfaces directly to the IOC processor via an 8-bit data bus. The keyboard contains an Intel UPI-41 Universal Peripheral Interface, which scans the keyboard, encodes the characters, and buffers the characters to provide N-key rollover. The keyboard itself is a high quality typewriter style keyboard containing the full ASCII character set. An upper/lower case switch allows the system to be used for document preparation. Cursor control keys are also provided.

Peripheral Interface

A UPI-41 Universal Peripheral Interface on the IOC board performs similar functions to the UPI-41 on the PIO board in the Model 210. It provides interface for other standard Intellec peripherals including a printer, high speed paper tape reader, high speed paper tape punch, and universal PROM programmer. Communication between the IPB and IOC is maintained over a separate 8-bit bidirectional data bus. Connectors for the four devices named above, as well as the two serial channels, are mounted directly on the IOC itself.

Control

User control is maintained through a front panel, consisting of a power switch and indicator, reset/boot switch, run/halt light, and eight interrupt switches and indicators. The front panel circuit board is attached directly to the IPB, allowing the eight interrupt switches to connect to the primary 8259A, as well as to the Intellec Series II bus.

Diskette System

The Intellec Series II double density diskette system provides direct access bulk storage, intelligent controller, and two diskette drives. Each drive provides ½ million bytes of storage with a data transfer rate of 500,000 bits/second. The controller is implemented with Intel's powerful Series 3000 Bipolar Microcomputer Set. The controller provides an interface to the Intellec Series II system bus, as well as supporting up to four diskette drives. The diskette system records all data in soft sector format. The diskette system is capable of performing seven different operations: recalibrate, seek, format track, write data, write deleted data, read data, and verify CRC.

Diskette Controller Boards — The diskette controller consists of two boards, the channel board and the interface board. These two PC boards reside in the Intellec Series II system chassis and constitute the diskette controller. The channel board receives, decodes and responds to channel commands from the 8080A-2 CPU in the Model 230. The interface board provides the diskette controller with a means of communication with the diskette drives and with the Intellec system bus. The interface board validates data during reads using a cyclic redundancy check (CRC) polynomial and generates CRC data during write operations. When the diskette controller requires access to Intellec system memory, the interface board requests and maintains DMA master control of the system bus, and generates the appropriate memory command. The interface board also acknowledges I/O commands as required by the Intellec bus. In addition to supporting a second set of double density drives, the diskette controller may co-reside with the Intel single density controller to allow up to 2.5 million bytes of on-line storage.

MULTIBUS Capability

All Intellec Series II models implement the industry standard MULTIBUS. MULTIBUS enables several bus masters, such as CPU and DMA devices, to share the bus and memory by operating at different priority levels. Resolution of bus exchanges is synchronized by a bus clock signal derived independently from processor clocks. Read/write transfers may take place at rates up to 5 MHz. The bus structure is suitable for use with any Intel microcomputer family.

SPECIFICATIONS

Host Processor (IPB)

RAM — 64K (system monitor occupies 62K through 64K)
ROM — 4K (2K in monitor, 2K in boot/diagnostic)

Diskette System Capacity (Basic Two Drives)

Unformatted
Per Disk: 6.2 megabits
Per Track: 82.0 kilobits

Formatted
Per Disk: 4.1 megabits
Per Track: 53.2 kilobits

Diskette Performance

Diskette System Transfer Rate — 500 kilobits/sec

Diskette System Access Time
Track-to-Track: 10 ms
Head Settling Time: 10 ms

Average Random Positioning Time — 260 ms

Rotational Speed — 360 rpm
Average Rotational Latency — 83 ms
Recording Mode — M^2FM

Physical Characteristics

Width — 17.37 in. (44.12 cm)
Height — 15.81 in. (40.16 cm)
Depth — 19.13 in. (48.59 cm)
Weight — 73 lb (33 kg)

Keyboard
Width — 17.37 in. (44.12 cm)
Height — 3.0 in. (7.62 cm)
Depth — 9.0 in. (22.86 cm)
Weight — 6 lb (3 kg)

Dual Drive Chassis
Width — 16.88 in. (42.88 cm)
Height — 12.08 in. (30.68 cm)
Depth — 19.0 in. (48.26 cm)
Weight — 64 lb (29 kg)

Electrical Characteristics

DC Power Supply

Volts Supplied	Amps Supplied	Typical System Requirements
+ 5 ± 5%	30	14.25
+ 12 ± 5%	2.5	0.2
− 12 ± 5%	0.3	0.05
− 10 ± 5%	1.5	15
* + 15 ± 5%	1.5	1.3
* + 24 ± 5%	1.7	

*Not available on bus.

AC Requirements — 50/60 Hz, 115/230V AC

Environmental Characteristics

Operating Temperature — 0° to 35°C (95°F)

Equipment Supplied

Model 230 chassis
Integrated processor board (IPB)
I/O controller board (IOC)
32K RAM board
CRT and keyboard
Double density floppy disk controller (2 boards)
Dual drive floppy disk chassis and cables
2 floppy disk drives (512K byte capacity each)
ROM-resident system monitor
ISIS-II system diskette with MCS-80/MCS-85 macroassembler

Reference Manuals

9800558 — A Guide to Microcomputer Development Systems (SUPPLIED)

9800550 — Intellec Series II Installation and Service Guide (SUPPLIED)

9800306 — ISIS-II System User's Guide (SUPPLIED)

9800556 — Intellec Series II Hardware Reference Manual (SUPPLIED)

9800301 — 8080/8085 Assembly Language Programming Manual (SUPPLIED)

9800292 — ISIS-II 8080/8085 Assembler Operator's Manual (SUPPLIED)

9800605 — Intellec Series II Systems Monitor Source Listing (SUPPLIED)

9800554 — Intellec Series II Schematic Drawings (SUPPLIED)

Reference manuals are shipped with each product only if designated SUPPLIED (see above). Manuals may be ordered from any Intel sales representative, distributor office or from Intel Literature Department, 3065 Bowers Avenue, Santa Clara, California 95051.

ORDERING INFORMATION

Part Number	Description
MDS-230	Intellec Series II Model 230 microcomputer development system (110V/60 Hz)
MDS-231	Intellec Series II Model 230 microcomputer development system (220V/50 Hz)

8086/8088 SOFTWARE DEVELOPMENT PACKAGE

PL/M-86 high level programming language

ASM86 macro assembler for 8086/8088 assembly language programming

LINK86 and LOC86 linkage and relocation utilities

CONV86 converter for conversion of 8080/8085 assembly language source code to 8086/8088 assembly language source code

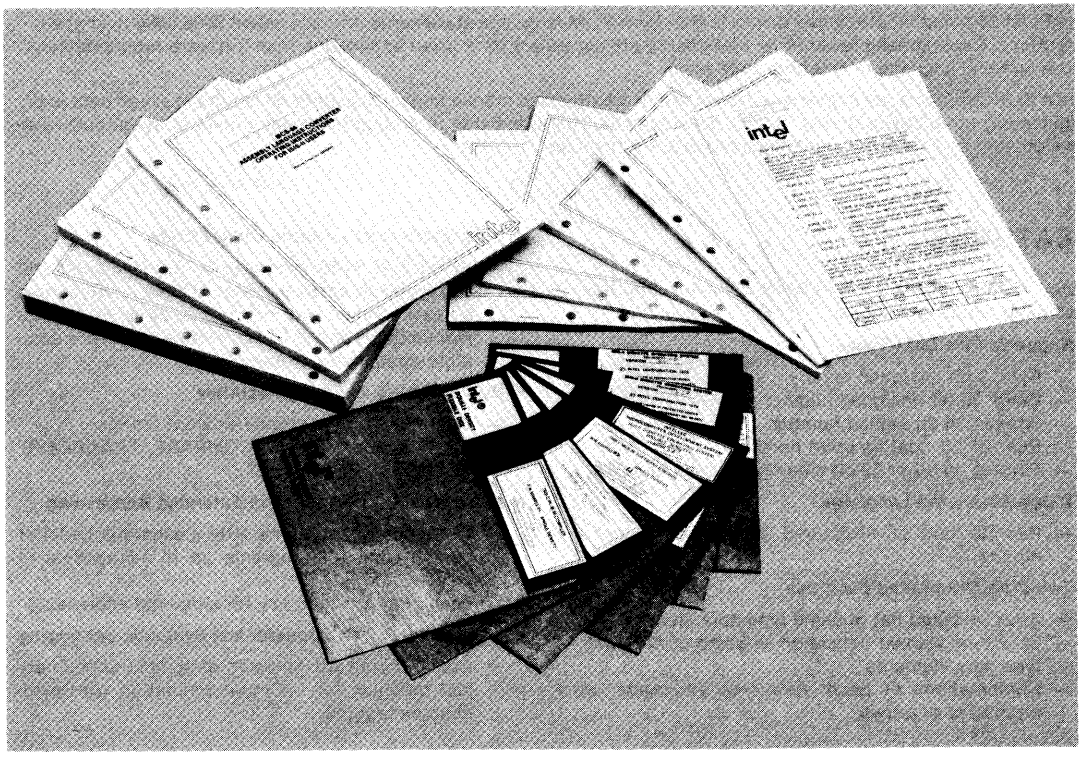
OH86 object-to-hexadecimal converter

LIB86 library manager

The 8086/8088 software development package provides a set of software development tools for the 8086 and the 8088 microprocessors and iSBC 86/12 single board computer. The package operates under the ISIS-II operating system on Intellec Microcomputer Development Systems—Model 800 or Series II—thus minimizing requirements for additional hardware or training for Intel Microcomputer Development System users.

The package permits 8080/8085 users to efficiently convert existing programs into 8086/8088 object code from either 8080/8085 assembly language source code or PL/M-80 source code.

For the new Intel Microcomputer Development System user, the package operating on an Intellec Model 230 Microcomputer Development System provides total 8086/8088 software development capability.



PL/M-86 HIGH LEVEL PROGRAMMING LANGUAGE

Sophisticated new compiler design allows user to achieve maximum benefits of 8086/8088 capabilities

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

Produces relocatable and linkable object code

Supports full extended addressing features of the 8086 and the 8088 microprocessors

Code optimization assures efficient code generation and minimum application memory utilization

Like its counterpart for MCS-80/85 program development, PL/M-86 is an advanced structured high level programming language. PL/M-86 is a new compiler created specifically for performing software development for the Intel 8086 and 8088 Microprocessors.

PL/M-86 has significant new capabilities over PL/M-80 that take advantage of the new facilities provided by the 8086 and the 8088 microprocessors, yet the PL/M-86 language remains upward compatible from PL/M-80.

With the exception of interrupts, hardware flags, and time-critical code sequences, PL/M-80 programs may be recompiled under PLM-86 with little or no conversion required. PL/M-86, 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 algorithmic language in which program statements can naturally express the program algorithm. This frees the programmer to concentrate on the system implementation without concern for burdensome details of 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 equivalent 8086/8088 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.

Since PL/M programs are implementation problem oriented and more compact, use of PL/M 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 compiler and programming language include:

- **Supports Five Data Types**
  - — 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
- **Block Structured Language**
  - — Permits use of structured programming techniques
- **Two Data Structuring Facilities**
  - — 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
- **Relocatable and Linkable Object Code**
  - — Permits PL/M-86 programs to be developed and debugged in small modules. These modules can be easily linked with other PL/M-86 or ASM86 object modules and/or library routines to form a complete application system.
- **Built-In String Handling Facilities**
  - — Operates on byte strings or word strings
  - — Six Functions: MOVE, COMPARE, TRANSLATE, SEARCH, SKIP, and SET
- **Automatic Support for 8086 Extended Addressing**
  - — Three compiler options offer a separate model of computation for programs up to 1-Megabyte in size
  - — Language transparency for extended addressing
- **Support for ICE-86 Emulator and Symbolic Debugging**
  - — Debug option for inclusion of symbol table in object modules for In-Circuit Emulation with symbolic debugging

- **Numerous Compiler Options**
  - — A host of 26 compiler options including:
    - Conditional compilation
    - Included file or copy facility
    - Two levels of optimization
    - Intra-module and inter-module cross reference
    - Arbitrary placement of compiler and user files on any available combination of disk drives
- **Reentrant and Interrupt Procedures**
  - — May be specified as user options

BENEFITS

PL/M-86 is designed to be an efficient, cost-effective solution to the special requirements of 8086/8088 Microcomputer Software Development, as illustrated by the following benefits of PL/M-86 use:

- **Reduced Learning Effort** — PL/M-86 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 less 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.
- **Simpler Project Development** — The Intellec Development Systems offer a cost-effective hardware base for the development of 8086 and 8088 designs. PL/M-86 and other elements of ISIS-II and the 8086/8088 Software Development Package are all that is needed for development of software for the 8086 and the 8088 microcomputers and iSBC 86/12 single board computer. This further reduces development time and costs because expensive (and remote) time sharing of large computers is not required. Present users of Intel Intellec Development Systems can begin to develop 8086 and 8088 designs without expensive hardware reinvestment or costly retraining.

SAMPLE PROGRAM

```
STATISTICS: DO;

/*The procedure in this module computes the mean and
variance of an array of data, X, of length N + 1, according
to the method of Kahan and Parlett (University of Cali-
fornia, Berkeley, Memo no. UCB/ERL M77/21.*/

STAT: PROCEDURE(X$PTR,N,MEAN$PTR,
      VARIANCE$PTR) PUBLIC;

DECLARE
      (X$PTR,MEAN$PTR,VARIANCE$PTR)
      POINTER,X BASED X$PTR (1) REAL,
      N INTEGER,
      MEAN BASED MEAN$PTR REAL,
      VARIANCE BASED VARIANCE$PTR REAL,
      (M,Q,DIFF) REAL,
      I INTEGER;

M = X(0);
M = 0.0;

DO I = 1 TO N;
   DIFF = X(I) - M;
   M = M + DIFF/FLOAT(I + 1);
   Q = Q + DIFF*DIFF*FLOAT(I)/FLOAT(I + 1);
   END;

   MEAN = M;
   VARIANCE = Q/FLOAT(N);

   END STAT;

   END STATISTICS;
```

ASM86 MACRO ASSEMBLER

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

ASM86 is the "high-level" macro assembler for the 8086/8088 assembly language. ASM86 translates symbolic 8086/8088 assembly language mnemonics into 8086/8088 machine code.

ASM86 should be used where maximum code efficiency and hardware control is needed. The 8086/8088 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 8086/8088 machine instructions. ASM86 will generate the shortest machine instruction possible given no forward referencing or given explicit information as to the characteristics of forward referenced symbols.

ASM86 offers many features normally found only in high-level languages. The 8086/8088 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.

FEATURES

Major features of the Intel 8086/8088 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 8086/8088 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.

- **Generates Relocatable and Linkable Object Code—Fully Compatible with LINK86, LOC86 and LIB86**
  - — Permits ASM86 programs to be developed and debugged in small modules. These modules can be easily linked with other ASM86 or PL/M-86 object modules and/or library routines to form a complete application system.

- **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.

BENEFITS

The 8086/8088 macro assembler allows the extensive capabilities of the 8086/8088 to be fully exploited. In any application, time and space critical routines can be effectively written in ASM86. The 8086/8088 assembler outputs relocatable and linkable object modules. These object modules may be easily combined with object modules written in PL/M-86—Intel's structured, high-level programming language. ASM86 compliments PLM-86 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 8086/8088 relocation and linkage utilities.

CONV86

MCS-80/85 to MCS-86 ASSEMBLY LANGUAGE CONVERTER UTILITY PROGRAM

Translates 8080/8085 Assembly Language Source Code to 8086/8088 Assembly Language Source Code

Provides a fast and accurate means to convert 8080/8085 programs to the 8086 and the 8088, facilitating program portability

Automatically generates proper ASM-86 directives to set up a "virtual 8080" environment that is compatible with PLM-86

In support of Intel's commitment to software portability, CONV86 is offered as a tool to move 8080/8085 programs to the 8086 and the 8088. A comprehensive manual, "MCS-86 Assembly Language Converter Operating Instructions for ISIS-II Users" (9800642), covers the entire conversion process. Detailed methodology of the conversion process is fully described therein.

CONV86 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 8086/8088 source file with embedded caution messages.

The OUTPUT file is an 8086/8088 source file.

CONV86 issues a caution message when it detects a potential problem in the converted 8086/8088 code.

A transliteration of the 8080/8085 programs occurs, with each 8080/8085 construct mapped to its exact 8086/8088 counterpart:

—Registers
—Condition flags
—Instructions
—Operands
—Assembler directives
—Assembler control lines
—Macros

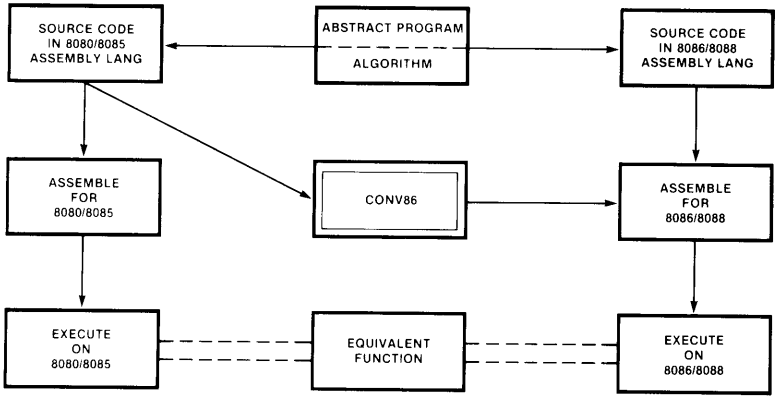
Because CONV86 is a transliteration process, there is the possibility of as much as a 15%-20% code expansion over the 8080/8085 code. For compactness and efficiency it is recommended that critical portions of programs be re-coded in 8086/8088 assembly language.

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 diagram illustrates the flow of the conversion process. Initially, the abstract program may be represented in 8080/8085 or 8086/8088 assembly language to execute on that respective target machine. The conversion process is porting a source destined for the 8080/8085 to the 8086 or the 8088 via CONV86.



PORTING 8080/8085 SOURCE CODE TO THE 8086/8088

LINK86

Automatic combination of separately compiled or assembled 8086/8088 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 LINK86 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

LINK86 combines object modules specified in the LINK86 input list into a single output module. LINK86 combines segments from the input modules according to the order in which the modules are listed.

Support for incremental linking is provided since an output module produced by LINK86 can be an input to another link. At each stage in the incremental linking process, unneeded public symbols may be purged.

LINK86 supports type checking of public and external symbols reporting an error if their types are not consistent.

LINK86 will link any valid set of input modules without any controls. However, controls are available to control the output of diagnostic information in the LINK86 process and to control the content of the output module.

LINK86 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 LOC86, and enter final testing with much of the work accomplished.

LOC86

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

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 8086/8088 memory

Abbreviated control syntax

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

LOC86 converts relative addresses in an input module to absolute addresses. LOC86 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.

LOC86 will relocate any valid input module without any controls. However, controls are available to control the output of diagnostic information in the LOC86 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.

OH86

Converts an 8086/8088 absolute object module to symbolic hexadecimal format

Converts an absolute module to a more readable format that can be displayed on a CRT or printed for debugging

Facilitates preparing a file for later loading by a symbolic hexadecimal loader, such as the iSBC Monitor or Universal PROM Mapper

The OH86 command converts an 8086/8088 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 that can be displayed or printed.

The module to be converted must be in absolute format; the output from LOC86 is in absolute format.

LIB86

LIB86 is a library manager program which allows you to:

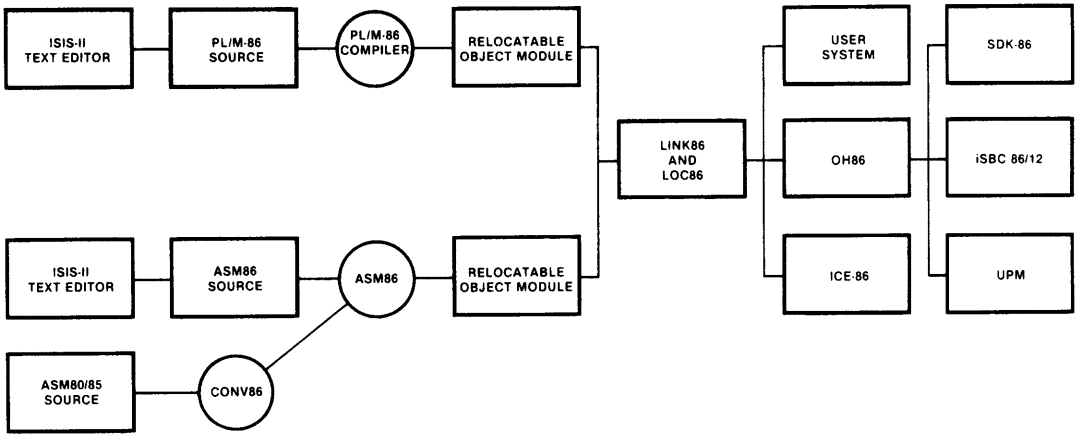
- **Create specially formatted files to contain libraries of object modules**
- **Maintain these libraries by adding or deleting modules**
- **Print a listing of the modules and public symbols in a library file**

Libraries can be used as input to LINK86 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, LIB86, creates and maintains files containing object modules. The operation of LIB86 is controlled by commands to indicate which operation LIB86 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 LIB86 program and returns control to ISIS-II


ISIS-II TEXT EDITOR
PL/M-86 SOURCE
PL/M-86 COMPILER
RELOCATABLE OBJECT MODULE
ISIS-II TEXT EDITOR
ASM86 SOURCE
ASM86
RELOCATABLE OBJECT MODULE
ASM80/85 SOURCE
CONV86
LINK86 AND LOC86
USER SYSTEM
OH86
ICE-86
SDK-86
iSBC 86/12
UPM

SPECIFICATIONS

Operating Environment

Required Hardware

Intellec Microcomputer Development System

— MDS-800, MDS-888
— Series II MDS-220 or MDS-230
64K Bytes of RAM Memory

Dual Diskette Drives

— Single or Double* Density

System Console

— CRT or Hardcopy Interactive Device

Optional Hardware

Universal PROM Programmer
Line Printer*
ICE-86™*

Required Software

ISIS-II Diskette Operating System

— Single or Double* Density

*Recommended

Documentation Package

PL/M-86 Programming Manual (9800466)
ISIS-II PL/M-86 Compiler Operator's Manual (9800478)
MCS-86 User's Manual (9800722)
MCS-86 Software Development Utilities Operating Instructions for ISIS-II Users (9800639)
MCS-86 Macro Assembly Language Reference Manual (9800640)
MCS-86 Macro Assembler Operating Instructions for ISIS-II Users (9800641)
MCS-86 Assembly Language Converter Operating Instructions for ISIS-II Users (9800642)
Universal PROM Programmer User's Manual (9800819A)

Flexible Diskettes

— Single and Double* Density

ORDERING INFORMATION

Part Number	Description
MDS-311	8086/8088 Software Development Package

Also available in the following development support packages:

Part Number	Description
SP86A-KIT	SP86A Support Package (for Intellec Model 800) Includes ICE-86 In-Circuit Emulator (MDS-86-ICE) and 8086/8088 Software Development Package (MDS-311)
SP86B-KIT	SP86B Support Package (for Series II) Includes ICE-86 In-Circuit Emulator (MDS-86-ICE), 8086/8088 Software Development Package (MDS-311), and Series II Expansion Chassis (MDS-201)

8089 ASSEMBLER SUPPORT PACKAGE

8089 I/O processor program generation on the Intellec Microcomputer Development System.

Relocatable object module compatible with the 8086 and 8088 Microprocessors.

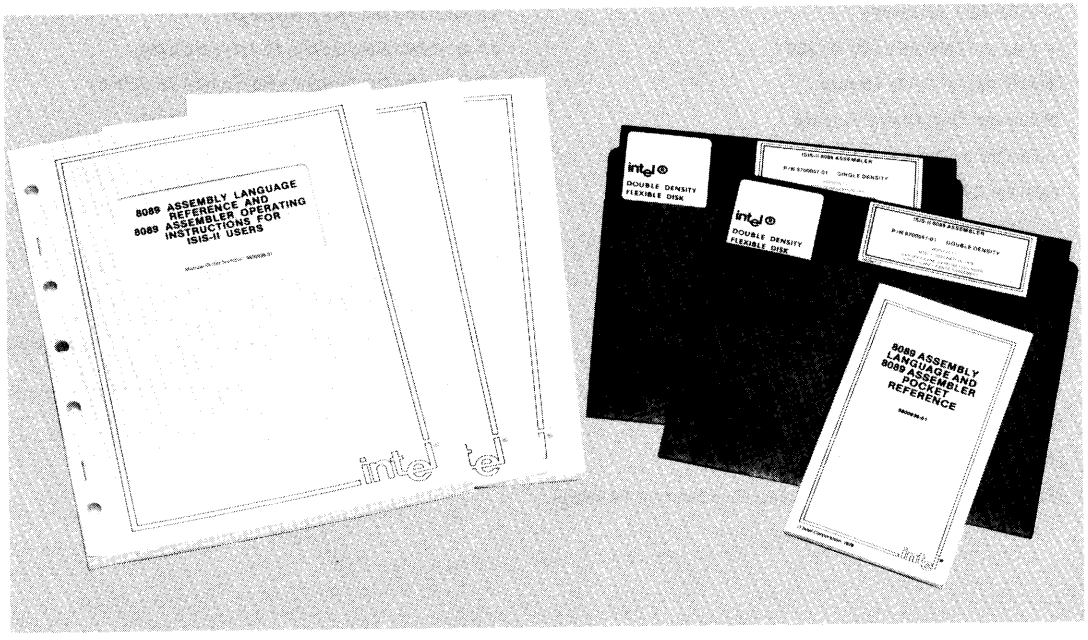
Supports 8089-based addressing modes with a structure facility that enables easy access to based data.

Fully detailed set of error messages.

Includes software development utilities to facilitate 8089 design.

- **—LINK86: Combines 8086 or 8088 object modules with 8089 object modules and resolves external references.**
- **—LOC86: Assigns absolute memory addresses to 8089 object modules.**
- **—OH86: Converts 8086/8088/8089 object code to symbolic hexadecimal format.**
- **—UPM86: A PROM programming aid which has been updated to support PROM programming for 8086, 8088 and 8089 applications.**

The 8089 Assembler Support Package extends Intellec microcomputer development system support to the 8089 i/O Processor. The assembler translates 8089 assembly language source instructions into appropriate machine operation codes. The 8089 Assembler Support Package allows the programmer to fully utilize the capabilities of the 8089 I/O Processor.



FUNCTIONAL DESCRIPTION

The 8089 Assembler Support Package contains the 8089 assembler (ASM89) as well as LINK86 and LOC86—relocation and linkage utilities, OH86—8086/8088/8089 object code to hexadecimal converter, and UPM86—PROM programming software updated to program object code in the 8086 formats. ASM89 translates symbolic 8089 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.

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 the 8089 into an 8086 or 8088 system.

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

A sample assembly listing is shown in table 1.

```
:8089 ASSEMBLER                                                    PAGE    1

ISIS-II 8089 ASSEMBLER V1.0 ASSEMBLY OF MODULE CONSOL
OBJECT MODULE PLACED IN :F0:CONSOL.OBJ
ASSEMBLER INVOKED BY ASM89 CONSOL.SRC


0000                    1  CONSOLE  SEGMENT
                        2
                        3  ; INITIALIZE 8275 CRT AND 8279 KEYBOARD CONTROLLERS
                        4
                        5  CONTROL        STRUC
                        6    ; 8275 PORTS
0000                    7    PARA75:  DS   1          ; PARAMETER PORT
0001                    8    NULL01:  DS   1
0002                    9    STAT75:  DS   1          ; STATUS/COMMAND PORT
                       10
                       11    ; 8279 PORTS
0003                   12    NULL02:  DS   3
0006                   13    STAT79:  DS   1          ; STATUS/COMMAND PORT
0007                   14  CONTROL        ENDS
                       15
0000  1138 0040      16    MOVI  GA.40H             ; SET PORT BASE ADDRESS
0004  0A4C 02 00      17    MOVBI [GA].STAT75.0      ; INITIALIZE 8275
0008  0A4C 00 4F      18    MOVBI [GA].PARA75.4FH
000C  0A4C 00 03      19    MOVBI [GA].PARA75.03H
0010  0A4C 00 6A      20    MOVBI [GA].PARA75.6AH
0014  0A4C 00 19      21    MOVBI [GA].PARA75.19H
                       22
0018  0A4C 06 00      23    MOVBI [GA].STAT79.0      ; INITIALIZE 8279
001C  0A4C 06 38      24    MOVBI [GA].STAT79.38H
                       25
0020                   26  CONSOLE ENDS
                       27
                       28        END

8089 ASSEMBLER                                                     PAGE    2

SYMBOL TABLE
------------

 DEFN VALUE TYPE  NAME
 ---- ----- ----  ----

    1  0000  SYM   CONSOLE
    5  0000  STR   CONTROL
    8  0001  SYM   NULL01
   12  0003  SYM   NULL02
    7  0000  SYM   PARA75
    9  0002  SYM   STAT75
   13  0006  SYM   STAT79

ASSEMBLY COMPLETE, NO ERRORS FOUND
```

Table 1. Sample 8089 Assembly Listing

SPECIFICATIONS

Operating Environment

Required Hardware

Intellec Microcomputer Development System

—MDS-800, MDS-888

—Series II Models 220 or 230

64K Bytes of RAM Memory

Minimum One Diskette Drive

—Single or Double* Density

System Console

—CRT or Hardcopy Interactive Device

Optional Hardware

Universal PROM Programmer*
Line Printer*

Required Software

ISIS-II Diskette Operating System

—Single or Double* Density

Documentation Package

8089 Assembler User's Guide (9800938)

8089 Assembler Pocket Reference (9800936)

MCS-86 Software Development Utilities
Operating Instructions for ISIS-II User's (9800639)

MCS-86 Absolute Object File Formats (9800821)

Universal PROM Programmer User's Manual (9800819)

Flexible Diskettes

—Single and Double* Density

*Recommended

ORDERING INFORMATION:

Part Number	Description
MDS-312	8089 Assembler Support Package

ICE-86™
8086 IN-CIRCUIT EMULATOR

Hardware in-circuit emulation

Full symbolic debugging

Breakpoints to halt emulation on a wide variety of conditions

Comprehensive trace of program execution, both conditional and unconditional

Disassembly of trace or memory from object code into assembler mnemonics

2K bytes of high speed ICE-86 mapped memory

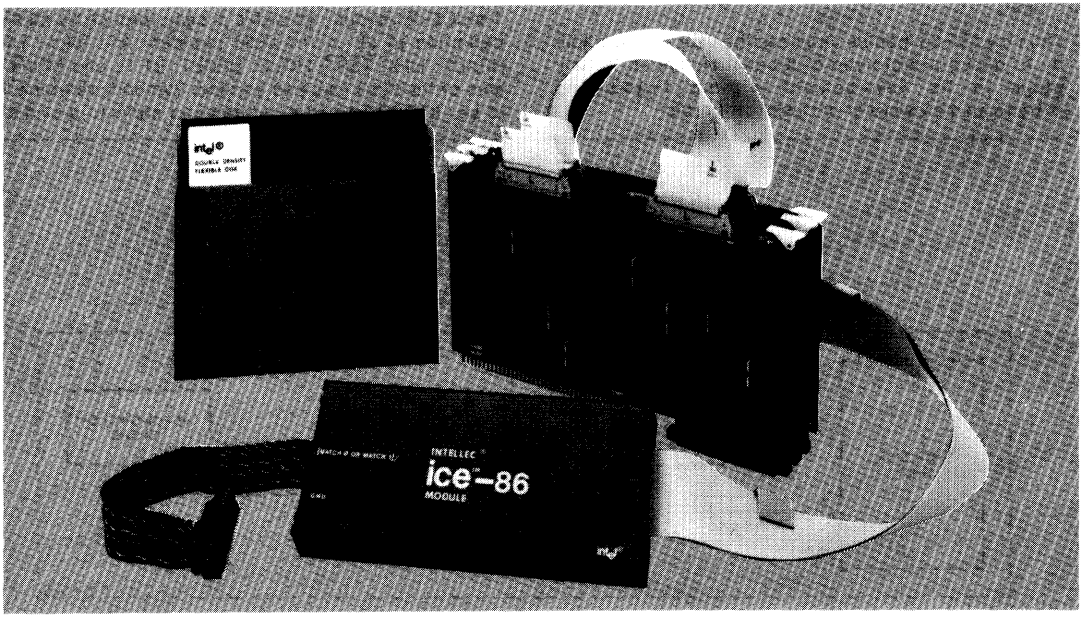
Software debugging with or without user system

Handles full 1 megabyte addressability of 8086

Compound commands

Command macros

The ICE-86 module provides In-Circuit Emulation for the 8086 microprocessor and the iSBC 86/12 Single Board Computer. It includes three circuit boards which reside in Intellec® Microcomputer Development Systems. A cable and buffer box connect the Intellec system to the user system by replacing the user's 8086. Powerful Intellec debug functions are thus extended into the user system. Using the ICE-86 module, the designer can execute prototype software in continuous or single-step mode and can substitute blocks of Intellec system memory for user equivalents. Breakpoints allow the user to stop emulation on user-specified conditions, and the trace capability gives a detailed history of the program execution prior to the break. All user access to the prototype system software may be done symbolically by referring to the source program variables and labels.



A typical ICE-86 development configuration. It is based on a Model 230 Development System, which also includes a Double Density Diskette Operating System and a Model 201 Expansion Chassis (which holds the ICE-86 emulator). The ICE-86 module is shown connected to a user prototype system, in this case an SDK-86.

INTEGRATED HARDWARE/SOFTWARE DEVELOPMENT

The ICE-86 emulator allows hardware and software development to proceed interactively. This is more effective than the traditional method of independent hardware and software development followed by system integration. With the ICE-86 module, prototype hardware can be added to the system as it is designed. Software and hardware testing occurs while the product is being developed.

Conceptually, the ICE-86 emulator assists three stages of development:

1. It can be operated without being connected to the user's system, so ICE-86 debugging capabilities can be used to facilitate program development before any of the user's hardware is available.
2. Integration of software and hardware can begin when any functional element of the user system hardware is connected to the 8086 socket. Through ICE-86 mapping capabilities, Intellec memory, ICE memory, or diskette memory can be substituted for missing prototype memory. Time-critical program modules are debugged before hardware implementation by using the 2K-bytes of high-speed ICE-resident memory. As each section of the user's hardware is completed, it is added to the prototype. Thus each section of the hardware and software is "system" tested as it becomes available.
3. When the user's prototype is complete, it is tested with the final version of the user system software. The ICE-86 module is then used for real time emulation of the 8086 to debug the system as a completed unit.

Thus the ICE-86 module provides the user with the ability to debug a prototype or production system at any stage in its development without introducing extraneous hardware or software test tools.

SYMBOLIC DEBUGGING

Symbols and PL/M statement numbers may be substituted for numeric values in any of the ICE-86 commands. This allows the user to make symbolic references to I/O ports, memory addresses, and data in the user program. Thus the user need not remember the addresses of variables or program subroutines.

Symbols can be used to reference variables, procedures, program labels, and source statements. A variable can be displayed or changed by referring to it by name rather than by its absolute location in memory. Using symbols for statement labels, program labels, and procedure names simplifies both tracing and breakpoint setting. Disassembly of a section of code from either trace or program memory into its assembly mnemonics is readily accomplished.

Furthermore, each symbol may have associated with it one of the data types BYTE, WORD, INTEGER, SINTEGER (for short, 8-bit integer) or POINTER. Thus the user need not remember the type of a source program variable when examining or modifying it. For example, the command "!VAR" displays the value in memory of variable VAR in a format appropriate to its type, while the command "!VAR = !VAR + 1" increments the value of the variable.

The user symbol table generated along with the object file during a PL/M-86 compilation or an ASM-86 assembly is loaded into memory along with the user program which is to be emulated. The user may add to this symbol table any additional symbolic values for memory addresses, constants, or variables that are found useful during system debugging.

The ICE-86 module provides access through symbolic definition to all of the 8086 registers and flags. The READY, NMI, $\overline{\text{TEST}}$, HOLD, RESET, INTR, and MN/$\overline{\text{MX}}$ pins of the 8086 can also be read. Symbolic references to key ICE-86 emulation information are also provided.

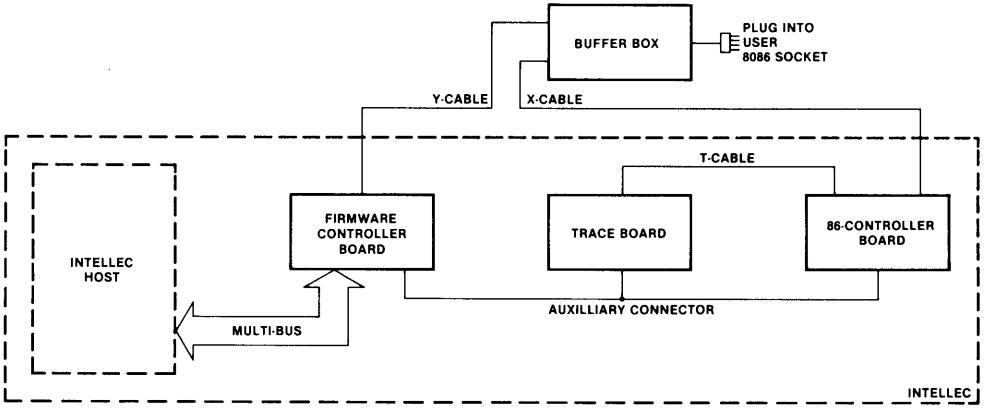
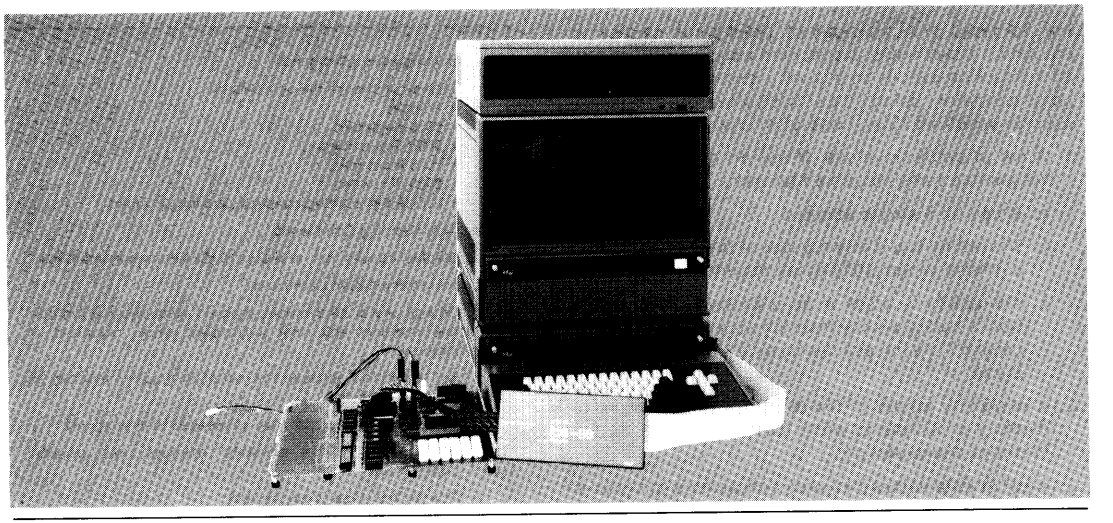


Figure 1. ICE-86 Block Diagram



MACROS AND COMPOUND COMMANDS

The ICE-86 module provides a programmable diagnostic facility which allows the user to tailor its operation using macro commands and compound commands.

A macro is a set of ICE-86 commands which is given a single name. Thus, a sequence of commands which is executed frequently may be invoked simply by typing in a single command. The user first defines the macro by entering the entire sequence of commands which he wants to execute. He then names the macro and stores it for future use. He executes the macro by typing its name and passing up to ten parameters to the commands in the macro. Macros may be saved on a disk file for use in subsequent debugging sessions.

Compound commands provide conditional execution of commands(IF), and execution of commands until a condition is met or until they have been executed a specified number of times (COUNT, REPEAT).

Compound commands and macros may be nested any number of times.

MEMORY MAPPING

Memory for the user system can be resident in the user system or "borrowed" from the Intellec System through ICE-86's mapping capability.

The ICE-86 emulator allows the memory which is addressed by the 8086 to be mapped in 1K-byte blocks to:

1. Physical memory in the user's system,
2. Either of two 1K-byte blocks of ICE-86 high speed memory,
3. Intellec memory,
4. A random-access diskette file.

The user can also designate a block of memory as nonexistent. The ICE-86 module issues an error message when any such "guarded" memory is addressed by the user program.

Command	Description
GO	Initializes emulation and allows the user to specify the starting point and breakpoints. Example: GO FROM .START TILL .DELAY EXECUTED where START and DELAY are statement labels.
STEP	Allows the user to single-step through the program.

Table 1. Summary of ICE-86 Emulation Commands

OPERATION MODES

The ICE-86 software is a RAM-based program that provides the user with easy-to-use commands for initiating emulation, defining breakpoints, controlling trace data collection, and displaying and controlling system parameters. ICE-86 commands are configured with a broad range of modifiers which provide the user with maximum flexibility in describing the operation to be performed.

Emulation

Emulation commands to the ICE-86 emulator control the process of setting up, running and halting an emulation of the user's 8086. Breakpoints and tracepoints enable ICE-86 to halt emulation and provide a detailed trace of execution in any part of the user's program. A summary of the emulation commands is shown in Table 1.

Breakpoints — The ICE-86 module has two breakpoint registers that allow the user to halt emulation when a specified condition is met. The breakpoint registers may be set up for execution or non-execution breaking. An execution breakpoint consists of a single address which causes a break whenever the 8086 executes from its queue an instruction byte which was obtained from

the address. A non-execution breakpoint causes an emulation break when a specified condition other than an instruction execution occurs. A non-execution breakpoint condition, using one or both breakpoint registers, may be specified by any one of or a combination of:

1. *A set of address values.* Break on a set of address values has three valuable features:
   a. Break on a single address.
   b. The ability to set any number of breakpoints within a limited range (1024 bytes maximum) of memory.
   c. The ability to break in an unlimited range. Execution is halted on any memory access to an address greater than (or less than) any 20-bit breakpoint address.
2. *A particular status of the 8086 bus* (one or more of: memory or I/O read or write, instruction fetch, halt, or interrupt acknowledge).
3. *A set of data values* (features comparable to break on a set of address values, explained in point one).
4. *A segment register* (break occurs when the register is used in an effective address calculation).

An external breakpoint match output for user access is provided on the buffer box. This allows synchronization of other test equipment when a break occurs.

Tracepoints — The ICE-86 module has two tracepoint registers which establish match conditions to conditionally start and stop trace collection. The trace information is gathered at least twice per bus cycle, first when the address signals are valid and second when the data signals are valid. If the 8086 execution queue is otherwise active, additional frames of trace are collected.

Each trace frame contains the 20 address/data lines and detailed information on the status of the 8086. The trace memory can store 1,023 frames, or an average of about 300 bus cycles, providing ample data for determining how the 8086 was reacting prior to emulation break. The trace memory contains the last 1,023 frames of trace data collected, even if this spans several separate emulations. The user has the option of displaying each frame of the trace data or displaying by instruction in actual ASM-86 Assembler mnemonics. Unless the user chooses to disable trace, the trace information is always available after an emulation.

Interrogation and Utility

Interrogation and utility commands give the user convenient access to detailed information about the user program and the state of the 8086 that is useful in debugging hardware and software. Changes can be made in both memory and the 8086 registers, flags, input pins, and I/O ports. Commands are also provided for various utility operations such as loading and saving program files, defining symbols and macros, displaying trace data, setting up the memory map, and returning control to ISIS-II. A summary of the basic interrogation and utility commands is shown in Table 2.

Memory/Register Commands
Display or change the contents of: • Memory • 8086 Registers • 8086 Status flags • 8086 Input pins • 8086 I/O ports • ICE-86 Pseudo-Registers (e.g. emulation timer)
Memory Mapping Commands
Display, declare, set, or reset the ICE-86 memory mapping.
Symbol Manipulation Commands
Display any or all symbols, program modules, and program line numbers and their associated values (locations in memory).
Set the domain (choose the particular program module) for the line numbers.
Define new symbols as they are needed in debugging.
Remove any or all symbols, modules, and program statements.
Change the value of any symbol.
TYPE
Assign or change the type of any symbol in the symbol table.
ASM
Disassemble user program memory into ASM-86 Assembler mnemonics.
PRINT
Display the specified portion of the trace memory.
LOAD
Fetch user symbol table and object code from the input file.
SAVE
Send user symbol table and object code to the output file.
LIST
Send a copy of all output (including prompts, input line echos, and error messages) to the chosen output device (e.g. disk, printer) as well as the console.
EVALUATE
Display the value of an expression in binary, octal, decimal, hexadecimal, and ASCII.
SUFFIX/BASE
Establish the default base for numeric values in input text/output display (binary, octal, decimal, or hexadecimal).
CLOCK
Select the internal (ICE-86 provided, for stand-alone mode only) or an external (user-provided) system clock.
RWTIMEOUT
Allows the user to time out READ/WRITE command signals based on the time taken by the 8086 to access Intellec memory or diskette memory.
ENABLE/DISABLE RDY
Enable or disable logical AND of ICE-86 Ready with the user Ready signal for accessing Intellec memory, ICE memory, or diskette memory.

Table 2. Summary of Basic ICE-86 Interrogation and Utility Commands

DIFFERENCES BETWEEN ICE-86 EMULATION AND THE 8086 MICROPROCESSOR

The ICE-86 module emulates the actual operation of the 8086 microprocessor with the following exceptions:

- The ICE-86 module will not respond to a user system NMI or RESET signal when it is out of emulation.
- Trap is ignored in single step mode and on the first instruction step of an emulation.
- The MIN/MAX line, which chooses the "minimum" or "maximum" configuration of the 8086, must not change dynamically in the user system.
- In the "minimum" mode, the user HOLD signal must remain active until HLDA is output by the ICE-86 emulator.
- The $\overline{RQ}/\overline{GT}$ lines in the "maximum" configuration are not supported.

The speed of run emulation by the ICE-86 module depends on where the user has mapped his memory. As the user prototype progresses to include memory, emulation becomes real time.

Memory Mapped To	Estimated Speed
User System	100% of real time*, up to 4 MHz clock
ICE	2 wait states per 8086-controlled bus cycle
Intellec	Approximately 0.02% of real time at 4 MHz clock
Diskette	**

*100% of real time is emulation at the user system clock rate with no wait states.

**The emulation speed from diskette is comparable to Intellec memory, but emulation must wait when a new page is accessed on the diskette.

DC CHARACTERISTICS OF ICE-86 USER CABLE

1. Output Low Voltages [V_{OL}(Max) = 0.4V]

	I_{OL} (Min)
AD0-AD15	8 mA (24 mA @ 0.5V)
A16/S3-A19/S7, $\overline{BHE}$/S7, $\overline{RD}$, $\overline{LOCK}$, QS0, QS1, $\overline{S0}$, $\overline{S1}$, $\overline{S2}$, $\overline{WR}$, M/$\overline{IO}$, DT/$\overline{R}$, $\overline{DEN}$, ALE, $\overline{INTA}$	8 mA (16 mA @ 0.5V)
HLDA	7 mA
$\overline{MATCH0\ OR\ MATCH1}$ (on buffer box)	16 mA

2. Output High Voltages [V_{OH} (Min) = 2.4V]

	I_{OH}(Min)
AD0-AD15	– 2 mA
A16/S3-A19/S7, $\overline{BHE}$/S7, $\overline{RD}$, $\overline{LOCK}$, QS0, QS1, $\overline{S0}$, $\overline{S1}$, $\overline{S2}$, $\overline{WR}$, M/$\overline{IO}$, DT/$\overline{R}$, $\overline{DEN}$, ALE, $\overline{INTA}$, HLDA	– 1 mA
$\overline{MATCH0\ OR\ MATCH1}$ (on buffer box)	– 0.8 mA

3. Input Low Voltages [V_{IL}(Max) = 0.8V]

	I_{IL} (Max)
AD0-AD15	– 0.2 mA
NMI, CLK	– 0.4 mA
READY	– 0.8 mA
INTR, HOLD, $\overline{TEST}$, RESET	– 1.4 mA
MN/$\overline{MX}$ (0.1 μf to GND)	– 3.3 mA

4. Input High Voltages [V_{IH}(Min) = 2.0V]

	I_{IH}(Max)
AD0-AD15	80 μA
NMI, CLK	20 μA
READY	40 μA
INTR, HOLD, $\overline{TEST}$, RESET	– 0.4 mA
MN/$\overline{MX}$ (0.1 μF to GND)	– 1.1 mA

5. $\overline{RQ}/\overline{GT0}$, $\overline{RQ}/\overline{GT1}$ are pulled up to + 5V through a 5.6K ohm resistor. No current is taken from user circuit at V_{CC} pin.

SPECIFICATIONS

ICE-86 Operating Environment

Required Hardware

Intellec microcomputer development system with:

1. Three adjacent slots for the ICE-86 module (Series II requires Model 201 Expansion Chassis.)
2. 64K bytes of Intellec memory. If user prototype program memory is desired, additional memory above the basic 64K is required.

System console
Intellec diskette operating system
ICE-86 module

Required Software

System monitor
ISIS-II, version 3.4 or subsequent
ICE-86 software

Equipment Supplied

Printed circuit boards (3)
Interface cable and emulation buffer module
Operator's manual
ICE-86 software, diskette-based

Emulation Clock

User system clock up to 4 MHz or 2 MHz ICE-86 internal clock in stand-alone mode

Physical Characteristics

Printed Circuit Boards

Width: 12.00 in (30.48 cm)
Height: 6.75 in (17.15 cm)
Depth: 0.50 in (1.27 cm)
Packaged Weight: 9.00 lb (4.10 kg)

Electrical Characteristics

DC Power

$V_{CC} = +5V +5\% -1\%$
$I_{CC} = 15A$ maximum; 11A typical
$V_{DD} = +12V \pm 5\%$
$I_{DD} = 120$ mA maximum; 80 mA typical
$V_{BB} = -10V \pm 5\%$ or $-12V \pm 5\%$ (optional)
$I_{BB} = 15$ mA maximum; 12 mA typical

Environmental Characteristics

Operating Temperature: 0° to 40°C

Operating Humidity: Up to 95% relative humidity without condensation.

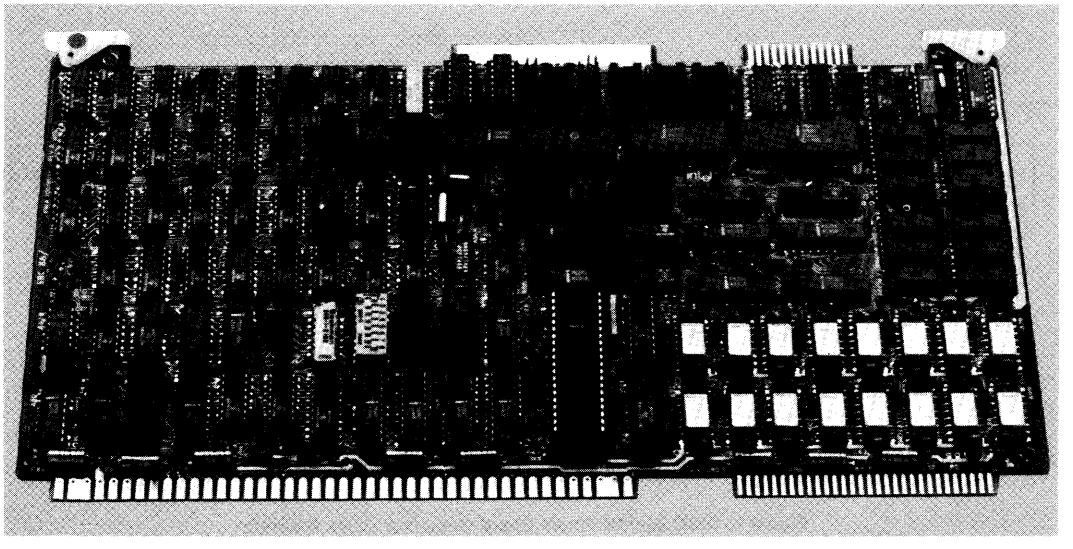
ORDERING INFORMATION

Part Number	Description
MDS-86-ICE	8086 CPU in-circuit emulator

iSBC 86/12A
SINGLE BOARD COMPUTER

- 8086 16 bit HMOS microprocessor central processor unit
- 32K-bytes of dual-port read/write memory expandable on-board to 64K-bytes with on-board refresh
- Sockets for up to 16K-bytes of read only memory expandable on-board to 32K-bytes
- System memory expandable to 1 megabyte
- 24 programmable parallel I/O lines with sockets for interchangeable line drivers and terminators
- Programmable synchronous/asynchronous RS232C compatible serial interface with software selectable baud rates
- Two programmable 16-bit BCD or binary timers/event counters
- 9 levels of vectored interrupt control, expandable to 65 levels
- Auxiliary power bus and power fail interrupt control logic for read/write memory battery backup
- MULTIBUS interface for multimaster configurations and system expansion
- Compatible with iSBC 80 family single board computers, memory, digital and analog I/O, and peripheral controller boards

The iSBC 86/12A Single Board Computer is a member of Intel's complete line of OEM microcomputer systems which take full advantage of Intel's LSI technology to provide economical self-contained computer based solutions for OEM applications. The iSBC 86/12A board is a complete computer system on a single 6.75 x 12.00-inch printed circuit card. The CPU, system clock, read/write memory, nonvolatile read only memory, I/O ports and drivers, serial communications interface, priority interrupt logic and programmable timers, all reside on the board. Full MULTIBUS interface logic is included to offer compatibility with the Intel OEM Microcomputer Systems family of Single Board Computers, expansion memory options, digital and analog I/O expansion boards and peripheral controllers.



FUNCTIONAL DESCRIPTION

Central Processing Unit

The central processor for the iSBC 86/12A board is Intel's 8086, a powerful 16-bit HMOS device. The 225 sq. mil chip contains 29,000 transistors and has a clock rate of 5MHz. The architecture includes four (4) 16-bit byte addressable data registers, two (2) 16-bit memory base pointer registers and two (2) 16-bit index registers, all accessed by a total of 24 operand addressing modes for complex data handling and very flexible memory addressing.

Instruction Set — The 8086 instruction repertoire includes variable length instruction format (including double operand instructions), 8-bit and 16-bit signed and unsigned arithmetic operators for binary, BCD and unpacked ASCII data, and iterative word and byte string manipulation functions. The instruction set of the 8086 is a superset of the 8080A/8085A family and with available software tools, programs written for the 8080A/8085A can be easily converted and run on the 8086 processor.

Architectural Features — A 6-byte instruction queue provides pre-fetching of sequential instructions and can reduce the 1.2μsec minimum instruction cycle to 400 nsec for queued instructions. The stack oriented architecture facilitates nested subroutines and co-routines, reentrant code and powerful interrupt handling. The memory expansion capabilities offer a 1 megabyte addressing range. The dynamic relocation scheme allows ease in segmentation of pure procedure and data for efficient memory utilization. Four segment registers (code, stack, data, extra) contain program loaded offset values which are used to map 16-bit addresses to 20-bit addresses. Each register maps 64K-bytes at a time and activation of a specific register is controlled explicitly by program control and is also selected implicitly by specific functions and instructions.

Bus Structure

The iSBC 86/12A microcomputer has three buses: an internal bus for communicating with on-board memory and I/O options, the MULTIBUS system bus for referencing additional memory and I/O options, and the dual-port bus which allows access to RAM from the on-board CPU and the MULTIBUS system bus. Local (on-board) accesses do not require MULTIBUS communication, making the system bus available for use by other MULTIBUS masters (i.e. DMA devices and other single board computers transferring to additional system memory). This feature allows true parallel processing in a multiprocessor environment. In addition, the MULTIBUS interface can be used for system expansion through the use of other 8- and 16-bit iSBC computers, memory and I/O expansion boards.

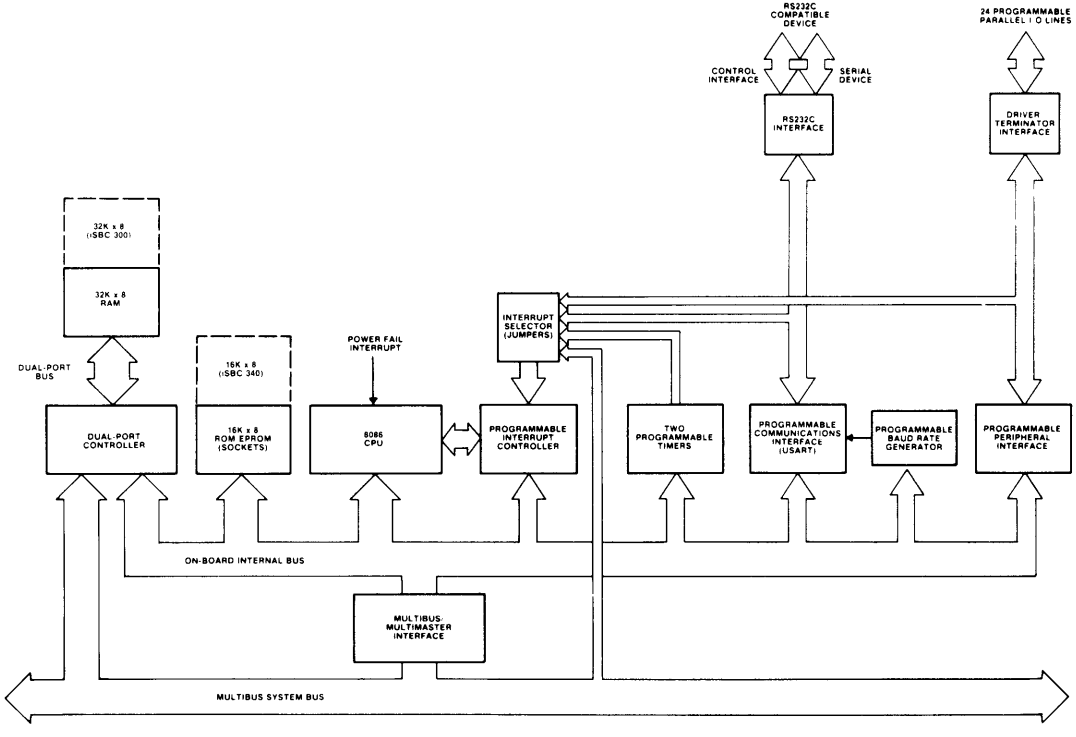


Figure 1. iSBC 86/12A Single Board Computer Block Diagram