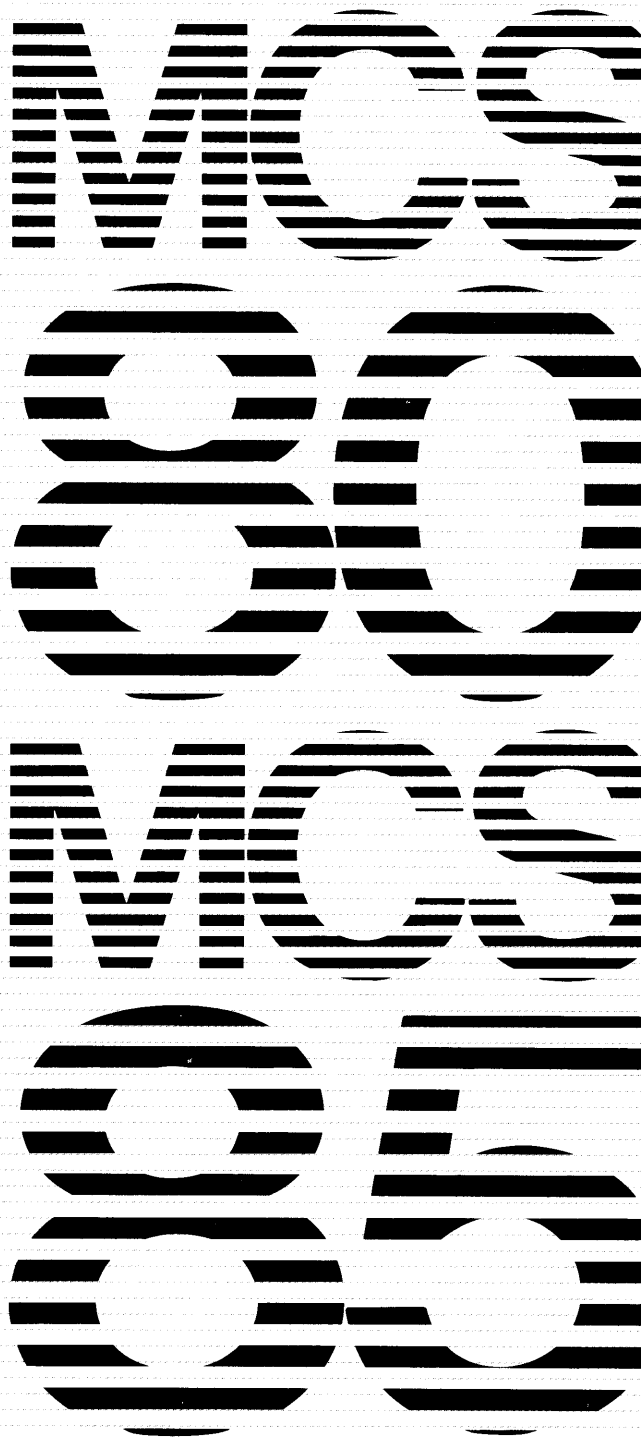
# Chapter 7
# Development Aids



# ICE-80™
# 8080 IN-CIRCUIT EMULATOR

**Connects Intellec® system to user configured system via an external cable and 40-pin plug, replacing the user system 8080**

**Allows real-time (2 MHz) emulation of user system 8080**

**Shares Intellec® RAM, ROM, and PROM memory and Intellec® I/O facilities with user system**

**Checks for up to three hardware and four software break conditions**

**Offers full symbolic debugging capabilities**

**Eliminates need for extraneous debugging tools residing in user system**

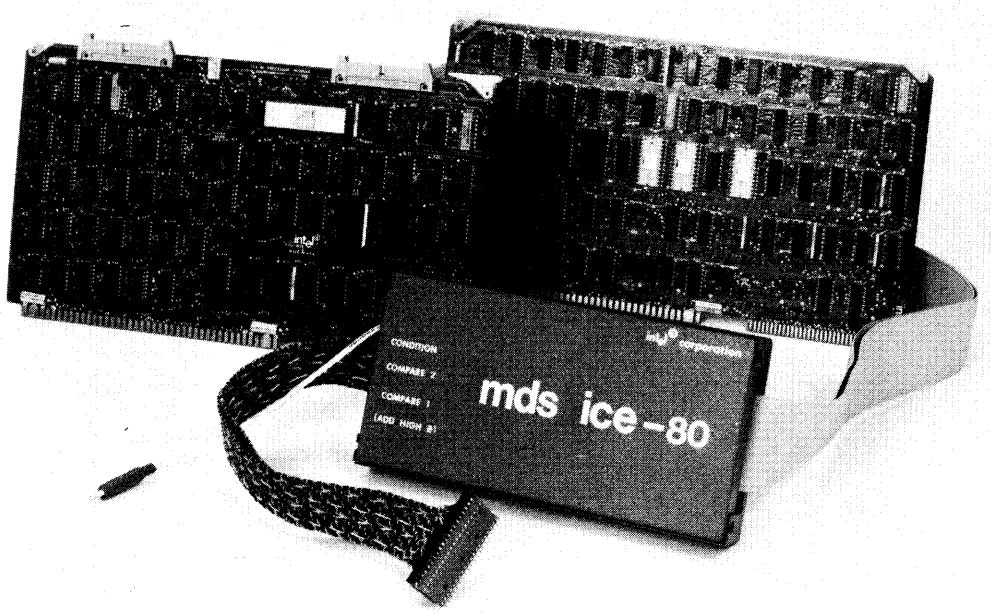
**Provides address, data, and 8080 status information on last 44 machine cycles emulated**

**Provides capability to examine and alter CPU registers, main memory, pin, and flag values**

**Integrates hardware and software development efforts**

**Available in diskette or paper tape versions**

The Intellec ICE-80 8080 In-Circuit Emulator is an Intellec resident module designed to interface with any user configured 8080 system. With ICE-80 as a replacement for a prototype system 8080, the designer may emulate the system's 8080 in real time, single step the system's program, and substitute Intellec memory and I/O for user system equivalents. Powerful Intellec debug functions are extended into the user system. For the first time the designer may examine and modify his system with symbolic references instead of absolute values.



## FUNCTIONAL DESCRIPTION

### Integrated Hardware/Software Development

Use of the ICE-80 module enables the system integration phase, which can be so costly and frustrating when attempting to mesh completed hardware and software products, to become a convenient two-way debug tool when begun early in the design cycle. The user prototype need consist of no more than an 8080 CPU socket and a user bus to begin integration of software and hardware development efforts. With the ICE-80 mapping capabilities, system resources may be accessed for missing prototype hardware. Hardware designs may be tested using system software to drive the final product. A functional block diagram of the ICE-80 module is shown in Figure 7-1.

### Symbolic Debugging Capability

ICE-80 provides for user-defined symbolic references to program memory addresses and data. Symbols may be substituted for numeric values in any of the ICE-80 commands. The user is thus relieved from looking up addresses of variables or program subroutines.

**Symbol Table —** The user symbol table generated along with the object file during a PL/M-80 compilation or a MAC80 or resident assembly, is loaded to memory along with the user program to be emulated. The user may add to this symbol table any additional symbolic values for memory addresses, constants, or variables found useful during system debugging. By referring to symbolic memory addresses, the user may be assured of examining, changing, or breaking at the intended location.

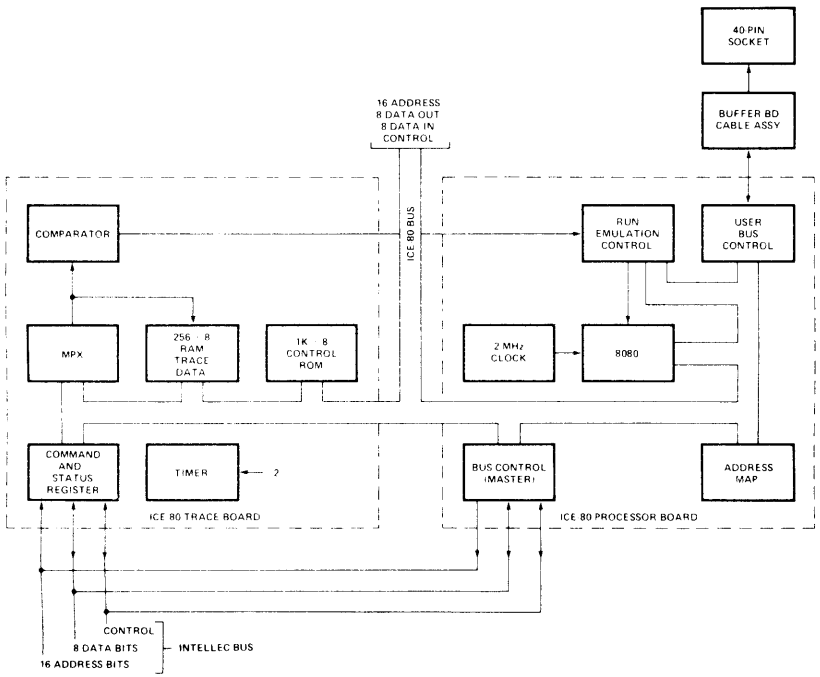
**Symbolic Reference —** ICE-80 provides symbolic definition of all 8080 registers, flags, and selected pins. The following symbolic references are also provided for user convenience: TIMER, a 16-bit register containing the number of $\phi_2$ clock pulses elapsed during emulation; ADDRESS, the address of the last instruction emulated; INTERRUPTENABLED, the user 8080 interrupt mechanism status; and UPPERLIMIT, the highest RAM address occupied by user memory.

### Debug Capability Inside User System

ICE-80 provides for user debugging of full prototype or production systems without introducing extraneous hardware or software test tools. ICE-80 connects to the user system through the socket provided for the user 8080 in the user system (see Figure 7-2). Intellec memory is used for the execution of the ICE-80 software, while I/O provides the user with the ability to communicate with ICE-80 and receive information on the generation of the user system. A sample ICE-80 debug session is shown in Figure 7-3.

### I/O Mapping and Memory

Memory and I/O for the user system may be resident in the user system or "borrowed" from the Intellec system through ICE-80's mapping capability.




**7-1. Functional Block Diagram of ICE-80 Module**

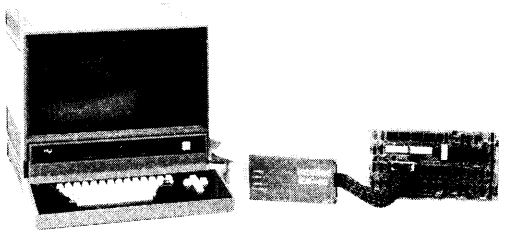


**Figure 7-2. ICE-80 Module Installed in User System**

**Memory Blocking** — ICE-80 separates user memory into 16 4K blocks. User I/O is divided into 16 16-port blocks. Each block of memory or I/O may be defined independently. The user may assign system equivalents to take the place of devices not yet designed for the user system during prototyping. In addition, memory or I/O may be accessed in place of user system devices during prototype or production checkout.

**Error Messages** — The user may also designate a block of memory or I/O as nonexistent. ICE-80 issues error messages when memory or I/O designated as nonexisting is accessed by the user program.

## Real-Time Trace

ICE-80 captures valuable trace information while the user is executing programs in real time. The 8080 status, the user memory or port addressed, and the data read or written (snap data), is stored for the last 44 machine cycles executed. This provides ample data for determining how the user system was reacting prior to emulation break. It is available whether the break was user initiated or the result of an error condition. For detailed information on the actions of CPU registers, flags, or other system operations, the user may operate in single or multiple step sequences tailored to system debug needs.

## Hardware

The heart of the ICE-80 is a microcomputer system utilizing Intel's 8080 microprocessor as its nucleus. This system communicates with the Intellec host processor via I/O commands. Host processor commands and ICE-80 status are interchanged through registers on the ICE-80 trace board. ICE-80 and the system also communicate through a control block resident in the Intellec main memory, which contains detailed configuration and status information transmitted at an emulation break. ICE-80 hardware consists of two PC boards — the processor and trace boards residing in the Intellec chassis — and a 6-foot cable interfacing to the user system. The trace and processor boards communicate with the system on the bus, and also with each other on a separate ICE-80 bus. ICE-80 connects to the user system through a cable that plugs directly into the socket provided for the user's 8080.

## Trace Board

The trace board talks to the system as a peripheral device. It receives commands to ICE-80 and returns ICE-80 responses. While ICE-80 is executing the user program, the trace board collects data for each machine cycle emulated (snap data). The information is continuously stored in high-speed bipolar memory.

**Breakpoint** — The trace board also contains two 24-bit hardware breakpoint registers which can be loaded by the user. While in emulation mode, a hardware comparitor is constantly monitoring address and status lines for a match to terminate an emulation. A user probe is also available for attachment to any user signal. When this signal goes true a break condition is recognized.

**Interrogation** — The trace board signals the processor board when a command to ICE-80 or break condition has been detected. The ICE-80 CPU then sends data stored on the trace board to the control block in memory. Snap data, along with information on 8080 registers and pin status and the reason for the emulation break, are then available for access during interrogation mode. Error conditions, if present, are transmitted and automatically displayed for the user.

## Processor Board

An 8080 CPU resides on the processor board. During emulation it executes instructions from the user's program. At all other times it executes instructions from the control program in the trace module's ROM.

**Timing** — The processor board contains an internal clock generator to provide clocks to the user emulation CPU at 2 MHz. The CPU can alternately be driven by a clock derived from user system signal lines. The clock source is selected by a jumper option on the board. A timer on the trace board counts the $\phi_2$ clock pulses during emulation and can provide the user with the exact timing of the emulation.

**On/Off Control** — The processor board turns on an emulation when ICE-80 has received a run command from the system. It terminates emulation when a break condition is detected on the trace board, or the user's program attempts to access memory or I/O ports designated as nonexistent in the user system, or the user 8080 is inactive for a quarter of a second.

**Status Storage** — The address map located on the processor board stores the assigned location of each user memory or I/O block. During emulation the processor board determines whether to send/receive information

```
ISIS 8080 MACROASSEMBLER, V1.0                PAGE 1

                ;USER PROGRAM TO OUTPUT A SERIES OF
                ;CHARACTERS TO SDK-80 CONSOLE DEVICE
                ;
1320                    ORG     1320H
01E3            C0      EQU     1E3H    ;SDK-80 CONSOLE OUT DRIVER
                ;
1320 0601       START:  MVI     B,1     ;SET UP B VALUE
1322 3A3613             LDA     DAT1    ;LOAD A WITH DAT1 VALUE
135 4F          LOOP:   MDV     C,A
1326 CDE301             CALL    C0      ;SEND C VALUE TO CONSOLE
1329 79                 MOV     A,C     ;RESTORE A
132A 93                 SBB     B       ;SUBTRACT B FROM A
132B 323713             STA     RSLT    ;STORE RESULT IN RSLT
132E FE40               CPI     40H     ;LAST VALUE TO PRINT
1330 C22513             JNZ     LOOP    ;LOOP AGAIN IF A>40H
1333 C32013             JMP     START   ;ELSE RESTART WHOLE PROCEDURE
                ;
1336 5A         DAT1:   DB      5AH
1337            RSLT:   DS      1
0000                    END

ISIS, V1.0      INITIAL ICE-80 SESSION
-ICE80          (Note: The SDK-80 Monitor has already been used to initialize the SDK-80 Board)
ISIS ICE-80, V1.0
(1) **XFORM MEMORY 0 TO 1 U
    *XFORM IO 0FH U
(2) *LOAD PROG. HEX
    ERR = 067
    STAT = 11H TYPE = 06H CMND = 07H ADDR = 1320H GOOD = 06H BAD = 04H
    *CHANGE MEMORY 1321H = FFH
    ERR = 067
    STAT = 11H TYPE = 06H CMND = 07H ADDR = 1321H GOOD = FFH BAD = FDH
    *LOAD PROG. HEX
(3) *GO FROM START UNTIL RSLT WRITTEN
    EMULATION BEGUN
(4) ERR = 067
    STAT = 11H TYPE = 07H CMND = 02H
(5) *DISPLAY CYCLES 5

    STAT = A2H ADDR = 1326H DATA = CDH
    STAT = 82H ADDR = 1327H DATA = E3H
    STAT = 82H ADDR = 1328H DATA = 01H
    STAT = 04H ADDR = FFFFH DATA = 13H
    STAT = 04H ADDR = FFFEH DATA = 29H
(6) *CHANGE DOUBLE REGISTER SP = 13FFH
    *BASE HEX
    *EQUATE STOP = 1333H
(7) *GO FROM START UNTIL STOP EXECUTED THEN DUMP
    EMULATION BEGUN
    B = 01H C = 41H D = 00H E = 00H H = 00H L = 00H F = 56H A = 40H P = 1320H * = 1333H S = 13FFH
    EMULATION TERMINATED AT 1333H
(8) *EXIT
    *FFFF
```

**Notes**

1. Set up user memory and I/O. The program is set up to execute in block 1 (1000H-1FFFH) of user memory, and requires access to the SDK-80 monitor (block 0) and I/O ports in block 0FH. Both ports and memory are defined as available to the user system. All other memory and I/O is initialized by ICE-80 as nonexistent (guarded).

2. A load command generates an error. The type and command numbers indicate that a data mismatch occurred on a write to memory command. The data to be written to address 1320H should have been 06H. When ICE-80 read the data after writing it, a 04H was detected. A change command to a different memory address hints that bit 1 does not go to 1 anywhere in this memory block. Examination indicates that a pin was shorted on the RAM located at 1300H-13FFH in the prototype system. The problem is fixed and a subsequent load succeeds.

3. A real-time emulation is begun. The program is executed from 'START' (1320H) and continues until 'RSLT' is written [in location 1328H, the contents of the accumulator is stored in (written into) 'RSLT'].

4. An error condition results: TYPE 07, CMND 02 indicate the program accessed is a guarded area.

5. The last 5 machine cycles executed are displayed. The last instruction executed was a call (CDH). The fourth and fifth cycles are a push operation (designated by status 04H) to store the program counter before executing the call. The stack pointer was not initialized in the program and is accessing memory location FFFFH.

6. After making a note to initialize the stack pointer in the next assembly, a temporary fix is effected by setting the stack pointer to the top of user available memory.

7. After setting the base for displays to hex and adding the symbol 'STOP' to the symbol table, emulation is started which will terminate when the instruction at 1333H ('STOP') is executed. When emulation terminates, a dump of the contents of user 8080 registers is requested. One can see that the value of the accumulator is set at 40H, the stack pointer is set at 13FFH, the last address executed (*) is 1333H, and the program counter has been set to 1320H.

8. Exit returns control to the MDS monitor.

**Figure 7-3. Sample ICE-80 Debug Session**

on the Intellec or user bus by consulting the address map. The processor board allows the ICE-80 CPU to gain access to the bus as a master to "borrow" Intellec facilities. At an emulation break, the processor board stores the status of specified 8080 input and output signals, disables all interaction with the user bus, and commands the trace board to send stored information to a control block in Intellec memory for access during interrogation mode.

### Cable Card

The cable card is included for cable driving. It transmits address and data bus information to the user system through a 40-pin connector that plugs into the user system in the socket designed for the 8080 when enabled by the processor module's user bus control logic.

### Software

The ICE-80 software driver is a RAM-based program providing easy to use English language commands for defining breakpoints, initiating emulation, and interrogating and altering the user system status recorded during emulation. ICE-80 commands are configured with a broad range of modifiers to provide the user with maximum flexibility in describing the operation to be performed. Listings of emulation commands, interrogation commands, and utility commands are provided in Tables 7-1, 7-2 and 7-3, respectively.

| Command | Operation |
|---|---|
| Go | Initiates real-time emulation and allows user to specify breakpoints, data retrieval, and conditions under which emulation should be reinitiated. |
| Step | Initiates emulation in single or multiple instruction increments. User may specify register dump or tailor diagnostic activity to his needs following each step, and define conditions under which stepping should continue. |
| Range | Delimits blocks of instructions for which register dump or tailored diagnostics are to occur. |
| Continue | Resumes real-time emulation. |
| Call | Emulates user system interrupt. |

**Table 7-1. ICE-80 Emulation Commands**

| Command | Operation |
|---|---|
| Base | Establishes mode of display for output data. |
| Display | Prints contents of memory, 8080 registers, input ports, 8080 flags, 8080 pins, snap data, symbol table, or other diagnostic data on list device. May also be used for base-to-base conversion, or for addition or subtraction in any base. |
| Change | Alters contents of memory, register, output port, or 8080 flag. |
| XFORM | Defines memory and I/O status. |
| Search | Looks through memory range for specified value. |

**Table 7-2. ICE-80 Interrogation Commands**

| Command | Operation |
|---|---|
| Load | Fetches user symbol table and object code from input device. |
| Save | Sends user symbol table and object code to output device. |
| Equate | Enters symbol name and value to user symbol table. |
| Fill | Fills memory range with specified value. |
| Move | Moves block of memory data to another area of memory. |
| Timeout | Enables/disables user CPU ¼ second wait state timeout. |
| List | Defines list device (diskette-based version only). |
| Exit | Returns program control to monitor. |

**Table 7-3. ICE-80 Utility Commands**

## SPECIFICATIONS

### Paper Tape-Based Operating Environment

**Required Hardware**
Intellec system
System console
Reader device
Punch device
ICE-80 module

**Required Software**
System monitor

### Diskette-Based Operating Environment

**Required Hardware**
Intellec system
32K bytes RAM memory
System console
Intellec diskette operating system
ICE-80 module

**Required Software**
System monitor
ISIS-II

### System Clock

Crystal controlled 2.185 MHz ± 0.01%. May be replaced by user clock through jumper selection.

### Physical Characteristics

**Width** — 12.00 in. (30.48 cm)
**Height** — 6.75 in. (17.15 cm)
**Depth** — 0.50 in. (1.27 cm)
**Weight** — 8.00 lb (3.64 kg)

### Electrical Characteristics

**DC Power Requirements**

$V_{CC}$ = + 5V, ± 5%
$I_{CC}$ = 9.81A max; 6.90A typ
$V_{DD}$ = + 12V, ± 5%
$I_{DD}$ = 79 mA max; 45 mA typ
$V_{BB}$ = − 9V, ± 5%
$I_{BB}$ = 1 mA max; 1μA typ

### Environmental Characteristics

**Operating Temperature** — 0°C to 40°C
**Operating Humidity** — Up to 95% relative humidity without condensation

### Equipment Supplied

Printed circuit modules (2)
Interface cables and buffer board
ICE-80 software driver, paper tape version
(ICE-80 software driver, diskette-based version is supplied with diskette operating systems)
Operator's Manual

## ORDERING INFORMATION

| Part Number | Description |
|---|---|
| MDS-80-ICE | 8080 CPU in-circuit emulator, cable assembly and interactive software included |

# UPP-103*
# UNIVERSAL PROM PROGRAMMER

**Replaces UPP-101, UPP-102 Universal PROM Programmers*

**Intellec® development system peripheral for PROM programming and verification**

**Provides personality cards for programming all Intel PROM families**

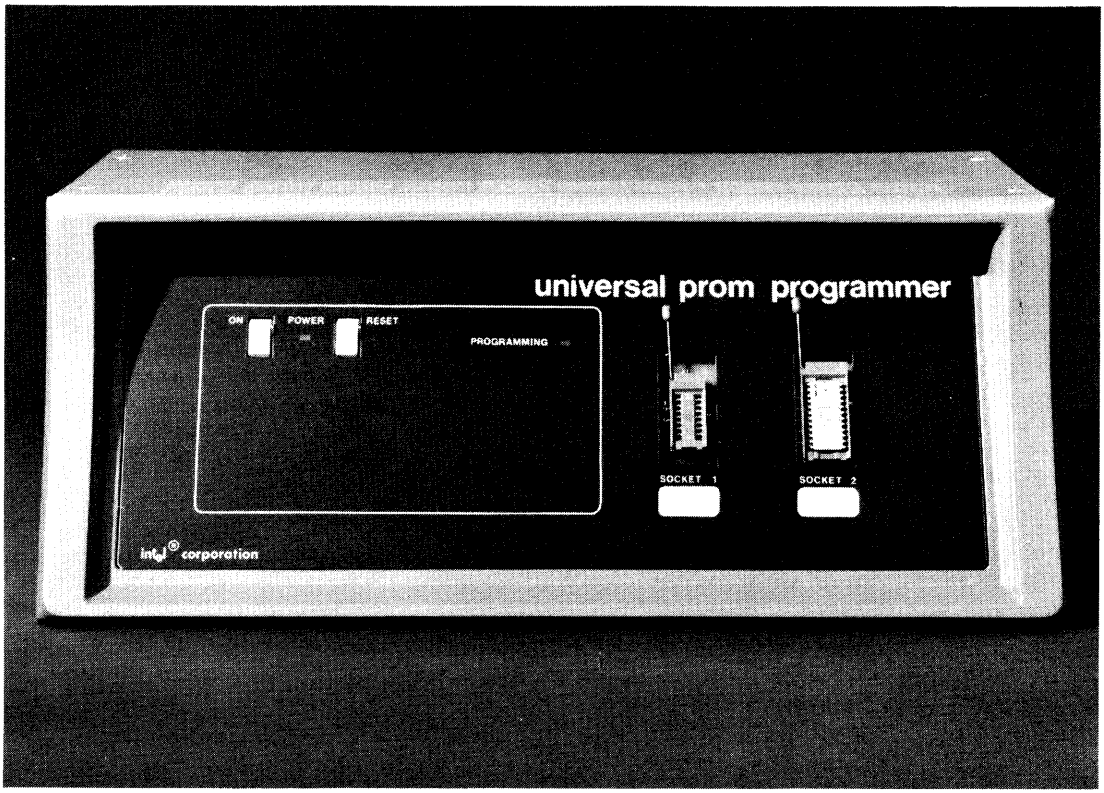
**Provides zero insertion force sockets for both 16-pin and 24-pin PROMs**

**Universal PROM mapper software provides powerful data manipulation and programming commands**

**Provides flexible power source for system logic and programming pulse generation**

**Holds two personality cards to facilitate programming operations using several PROM types**

The UPP-103 Universal PROM Programmer is an Intellec system peripheral capable of programming and verifying all of the Intel programmable ROMs (PROMs). In addition, the UPP-103 programs the PROM memory portions of the 8748 microcomputer, 8741 UPI, the 8755 PROM and I/O chip and the 2920 signal processor. Programming and verification operations are initiated from the Intellec development system console and are controlled by the universal PROM mapper (UPM) program.



## FUNCTIONAL DESCRIPTION

### Universal PROM Programmer

The basic Universal PROM Programmer (UPP) consists of a controller module, two personality card sockets, a front panel, power supplies, a chassis, and an Intellec development system interconnection cable. An Intel 4040-based intelligent controller monitors the commands from the Intellec System and controls the data transfer interface between the selected PROM personality card and the Intellec memory. A unique personality card contains the appropriate pulse generation functions for each Intel PROM family. Programming and verifying any Intel PROM may be accomplished by selecting and plugging in the appropriate personality card. The front panel contains a power-on switch and indicator, a reset switch, and two zero-force insertion sockets (one 16-pin and one 24-pin or two 24-pin). A central power supply provides power for system logic and for PROM programming pulse generation. The Universal PROM Programmer may be used as a table top unit or mounted in a standard 19-inch RETMA cabinet.

### Universal PROM Mapper

The Universal PROM Mapper (UPM) is the software program used to control data transfer between paper tape or diskette files and a PROM plugged into the Universal PROM Programmer. It uses Intellec system memory for intermediate storage. The UPM transfers data in 8-bit HEX, BNPF, or binary object format between paper tape or diskette files and the Intellec system memory. While the data is in Intellec system memory, it can be displayed and changed. In addition, word length, bit position, and data sense can be adjusted as required for the PROM to be programmed. PROMs may also be duplicated or altered by copying the PROM contents into the Intellec system memory. Easy to use program and compare commands give the user complete control over programming and verification operations. The UPM eliminates the need for a variety of personalized PROM programming routines because it contains the programming algorithms for all Intel PROM families. The UPM (diskette based version) is included with the Universal PROM Programmer.

## SPECIFICATIONS

### Hardware Interface

**Data** — Two 8-bit unidirectional buses

**Commands** — 3 write commands, 2 read commands, one initiate command

### Physical Characteristics

**Width** — 6 in. (14.7 cm)

**Height** — 7 in. (17.2 cm)

**Depth** — 17 in. (41.7 cm)

**Weight** — 18 lb (8.2 kg)

### Electrical Characteristics

**AC Power Requirements** — 50-60 Hz; 115/230V AC: 80W

### Environmental Characteristics

**Operating Temperature** — 0°C to 55°C

### Optional Equipment

**Personality Cards**

UPP-816: 2716 personality card

UPP-832: 2732 personality card

UPP-848: 8748, 8741 personality card with 40-pin adaptor socket

UPP-865: 3602, 3622, 3602A, 3622A, 3621, 3604, 3624, 3604A, 3624A, 3604AL, 36046-6, 3605, 3605A, 3625, 3625A, 3608, 3628, 3636

UPP-872: 8702A/1702A personality card

UPP-878: 8708/8704/2708/2704 personality card

UPP-955: 8755A personality card with 40-pin adaptor socket

**PROM Programming Sockets**

UPP-501: 16-pin/24-pin socket pair

UPP-502: 24-pin/24-pin socket pair

UPP-562: Socket adaptor for 3621, 3602, 3622, 3602A, 3622A

UPP-555: Socket adaptor for 3604AL, 36046-6, 3608, 3628, 3636

UPP-566: Socket adaptor for 3605, 3625, 3605A, 3625A

### Equipment Supplied

Cabinet

Power supplies

4040 intelligent controller module

Specified zero insertion force socket pair

Intellec development system interface cable

Universal PROM Mapper program (diskette-based version)

### Reference Manuals

**9800819** — Universal PROM Programmer User's Manual (SUPPLIED)

## ORDERING INFORMATION

| Part Number | Description |
|---|---|
| UPP-103 | Universal PROM programmer with 16-pin/24-pin socket pair and 24-pin/24-pin socket pair. |

# MODEL 220
# INTELLEC® SERIES II
# MICROCOMPUTER DEVELOPMENT SYSTEM

**Complete microcomputer development system in one package for MCS-86, MCS-85, MCS-80 and MCS-48™ microprocessor families**

**Single LSI electronics board with CPU, 32K bytes RAM memory, and 4K bytes ROM 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**

**Integral 250K-byte floppy disk with total storage capacity expandable to over 2M bytes**

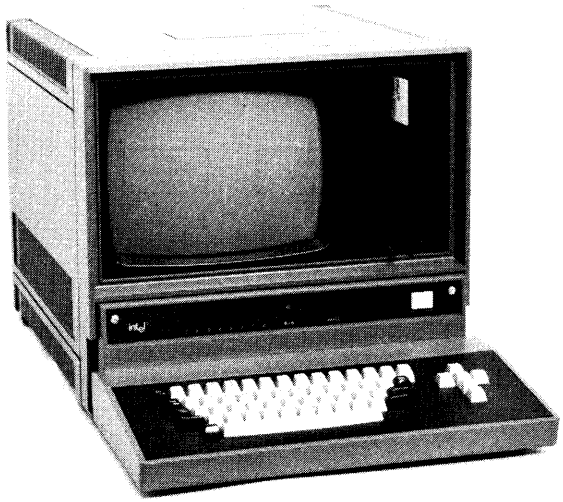
**Powerful ISIS-II Diskette Operating System with relocating macroassembler, linker, and locater**

**Standard MULTIBUS with multi-processor and DMA capability**

**Compatible with standard Intellec®/ iSBC™ expansion modules**

**Software compatible with previous Intellec® systems**

The Model 220 Intellec Series II Microcomputer Development System is a complete microcomputer development system integrated into one compact package. It includes a CPU with 32K bytes of RAM memory, 4K bytes of ROM memory, a 2000-character CRT, detachable full ASCII keyboard with cursor controls and upper/lower case capability, and a 250K-byte floppy diskette drive. Powerful ISIS-II Diskette Operating System software allows the Model 220 to be used quickly and efficiently for assembling and debugging programs for Intel's MCS-86, MCS-85, MCS-80, or MCS-48 microprocessor families without the need for paper tape handling. ISIS-II performs all file handling operations for the user, leaving him free to concentrate on the details of his own application. When used in conjunction with an optional in-circuit emulator (ICE) module, the Model 220 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 220 is a packaged, highly integrated microcomputer development system consisting of a CRT chassis with a 6-slot cardcage, power supply, fans, cables, single floppy diskette drive, and two printed circuit cards. A separate, full ASCII keyboard is connected with a cable.

**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 and floppy disk control. This card, mounted on the rear panel, also contains its own microprocessor, RAM and ROM memory, and I/O interface, 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, thus leaving the remaining 5 slots in the cardcage available for system expansion. A block diagram of the IOC is shown in Figure 7-4.

## 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.

## Input/Output

**IPB Serial Channels** — The I/O subsystem in the Model 220 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 data terminal. One channel contains current loop adapters. Both channels are implemented using Intel's 8251 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 interfaces for the CRT, keyboard, integral floppy disk and standard Intellec peripherals, including a printer, high speed paper tape reader/punch, and universal PROM programmer. The IOC contains its own independent microprocessor, also an 8080A-2. This 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 ROM are used for CRT screen refresh storage and the floppy disk buffer. These do not occupy any 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

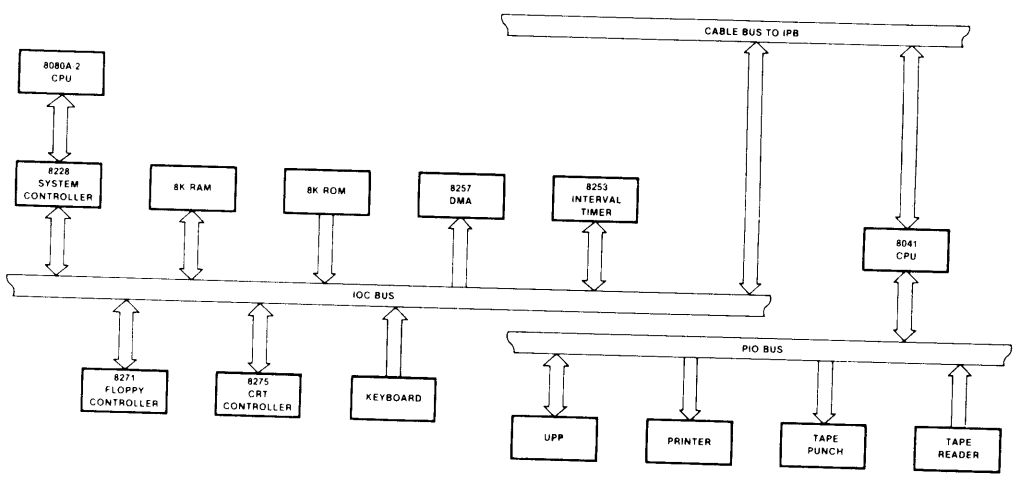



**Figure 7-4. I/O Controller (IOC) Block Diagram for the Model 220 Intellec Series II Microcomputer Development System**

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.

### Floppy Disk Drive

The floppy disk drive is controlled by an Intel 8271 single chip, programmable floppy disk controller. It transfers data via an Intel 8257 DMA controller between an IOC RAM buffer and the diskette. The 8271 handles reading and writing of data, formatting diskettes, and reading status, all upon appropriate commands from the IOC microprocessor.

### 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 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.

### 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.

### Expansion

The Model 220 may be expanded to 64K of RAM and up to 2.25M bytes of on-line diskette storage.

## SPECIFICATIONS

### Host Processor (IPB)

8080A-2 based, operating at 2.600 MHz.

**RAM** — 32K, expandable to 64K with iSBC 032 RAM boards (system monitor occupies 62K through 64K)

**ROM** — 4K (2K in monitor, 2K in boot/diagnostic)

**Bus** — MULTIBUS, maximum transfer rate of 5 MHz

**Clocks** — Host processor, crystal controlled at 2.6 MHz, bus clock, crystal controlled at 9.8304 MHz

### I/O Interfaces

2 Serial I/O Channels, RS232C, at 110-9600 baud (asynchronous) or 150-56K baud (synchronous). Baud rates and serial format fully programmable using Intel 8251A USARTs. Serial Channel 1 additionally provided with 20 mA current loop. Parallel I/O interfaces provided for paper tape punch, paper tape reader, printer, and UPP-103 Universal PROM Programmer.

### Interrupts

8-level, maskable, nested priority interrupt network initiated from front panel or user selected devices.

### Direct Memory Access (DMA)

Standard capability on MULTIBUS; implemented for user selected DMA devices through optional DMA module — maximum transfer rate of 2 MHz.

### Memory Access Time

**RAM** — 585 ns max

PROM — 450 ns max

### Diskette

**Diskette System Capacity** — 250K bytes (formatted)

**Diskette System Transfer Rate** — 160K bits/sec

**Diskette System Access Time**

Track-to-Track: 10 ms max
Average Random Positioning: 260 ms max
Rotational Speed: 360 rpm
Average Rotational Latency: 83 ms max
Recording Mode: FM

### Physical Characteristics

**Width** — 17.37 in. (44.12 cm)

**Height** — 15.81 in. (40.16 cm)

**Depth** — 19.13 in. (48.59 cm)

**Weight** — 86 lb (39 kg)

**Keyboard**
**Width** — 17.37 in. (44.12 cm)
**Height** — 3.0 in. (7.62 cm)
**Depth** — 9.0 in. (22.0 cm)
**Weight** — 6 lb (3 kg)

## Electrical Characteristics

**DC Power Supply**

| Volts Supplied | Amps Supplied | Typical System Requirements |
|---|---|---|
| + 5 ± 5% | 30.0 | 7.5 |
| + 12 ± 5% | 2.5 | 0.2 |
| − 12 ± 5% | 0.3 | 0.05 |
| − 10 ± 5% | 1.5 | 0.15 |
| + 15 ± 5% | 1.5 | 1.3* |
| + 24 ± 5% | 1.7 | 1.2* |

*Not available on bus.

**AC Requirements**
50-60 Hz. 115/230V AC

## Equipment Supplied

Model 220 chassis
Integrated processor board (IPB)
I/O controller board (IOC)
CRT and keyboard
250K-byte floppy disk drive
ROM resident system monitor
ISIS-II system diskette with MCS-80/MCS-85 macroassembler

## Reference Manuals

**9800558** — A Guide to Microcomputer Development Systems (SUPPLIED)

**9800559** — Intellec Series II Installation and Service Manual (SUPPLIED)

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

**9800556** — Intellec Series II Hardware Reference Manual (SUPPLIED)

**9800555** — Intellec Series II Hardware Interface Manual (SUPPLIED)

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

**9800605** — Intellec Series II System Monitor Source Listing (SUPPLIED)

**9800554** — Intellec Series II Schematic Drawing (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-220 | Intellec Series II Model 220 microcomputer development system (110V/60 Hz) |
| MDS-221 | Intellec Series II Model 220 microcomputer development system (220V/50 Hz) |

# 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**

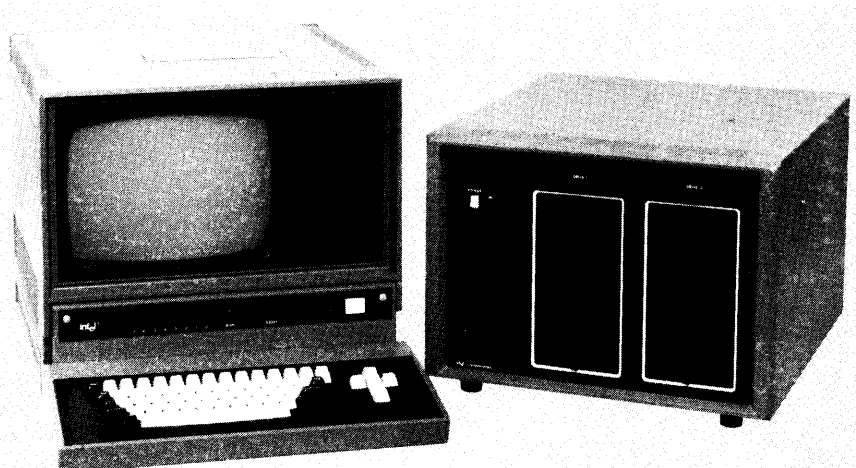
**Supports PL/M and FORTRAN high level languages**

**Standard MULTIBUS with multiprocessor and DMA capability**

**Compatible with standard Intellec®/iSBC**

**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.



Intel, Intellec, INSITE, Library Manager, MCS, MEGACHASSIS, MICROAMP, PROMPT, μSCOPE, MULTIBUS, RMX/80, UPI-41, ICE, and iSBC are trademarks of Intel Corp

© Intel Corporation, 1978

## 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 7-5.

**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.

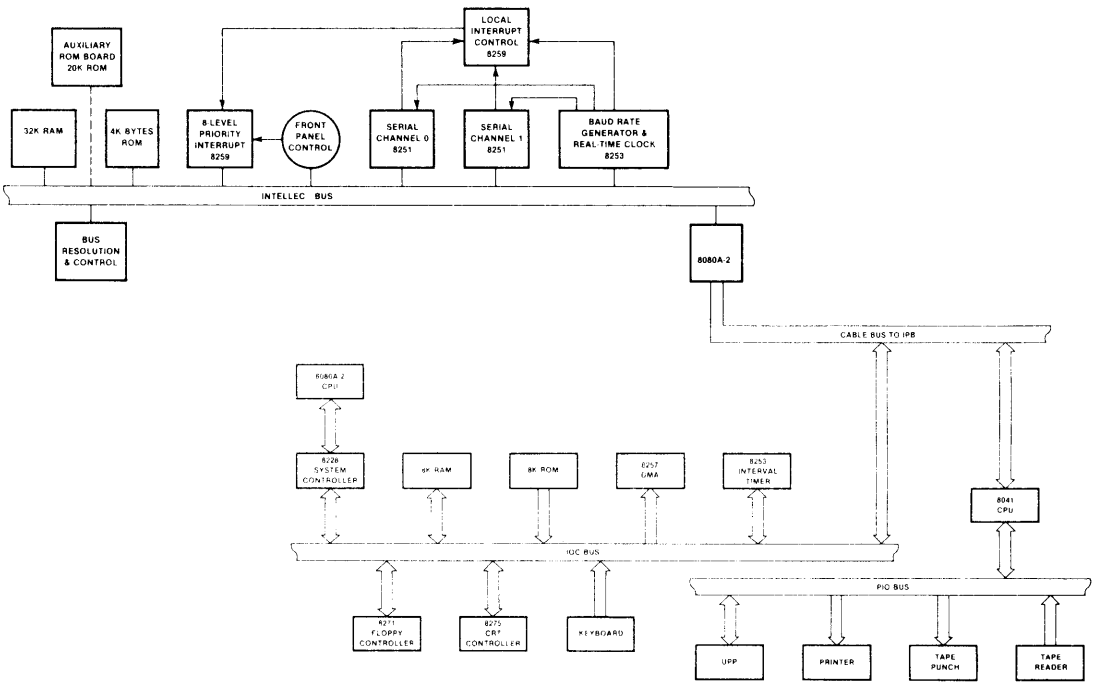



**Figure 7-5. Intellec Series II Model 230 Microcomputer Development System Block Diagram**

## 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) |

# ISIS-II
# DISKETTE OPERATING SYSTEM
# MICROCOMPUTER DEVELOPMENT SYSTEM

**Supports up to two hard disk drives (4 platters), four double density drives and two single density drives, providing up to 17 megabytes of storage in one system with up to 200 files per diskette, and 992 files per disk platter**

**Relocating MCS-80/MCS-85™ macro-assembler contains extended macro and conditional assembly capability**

**Command file facility allows console commands to be submitted from disk file**

**Disk operating system functions callable from user programs**

**Disk system text editor provides string search, substitution, insertions, and deletion commands**

**Supports resident, high level programming languages, PL/M, FORTRAN, BASIC, and COBOL**

**Provides dynamic allocation and deallocation of disk sectors for variable length files**

**Linker automatically combines separately assembled or compiled programs into single relocatable module**

**Library manager™ creates and updates program libraries**

**Supports all standard Intellec® peripherals**

**Provides access to all Intellec® monitor facilities**

The ISIS-II Microcomputer Development System Disk Operating System is a sophisticated, general purpose, high speed data handler and file manipulation system. It provides the ability to edit, assemble, compile, link, relocate, execute, and debug programs, and performs all user file management tasks. The ISIS-II operating system resides on the system disk and supports a broad range of user oriented design aid software. Total file management and input editing features greatly reduce software development time. The ISIS-II relocating macroassembler, linker, object locator, and library manager may be loaded from the disk in seconds. All passes of the assembler may be executed without the need for user intervention. Object code and listings may be directed to any output device, or stored as disk files. Powerful system console commands are provided in an easy to use context. Monitor mode may be entered by a special prefix to any system command or program call.

## FUNCTIONAL DESCRIPTION

The ISIS-II operating system resides on the system disk and supports a broad range of user oriented design aid software. Total file management and input editing features greatly reduce software development time. The ISIS-II relocating macroassembler, linker, object locator, and library manager may be loaded from the disk in seconds. All passes of the assembler may be executed without the need for user intervention. Object code and listings may be directed to any output device, or stored as disk files. A diagram of the ISIS-II system program development flow is shown in Figure 7-6.

### ISIS-II Files

A file is a user defined collection of information of variable length. ISIS-II also treats each of the standard Intellec system peripherals as files through preassignment of unique file names to each device. In this manner data may be copied from one device to another (i.e., tape reader to tape punch) using the same command required to copy one disk data file to another. ISIS-II provides automatic implementation of random access disk files. each file is identified by a user chosen name unique on its disk. Up to 200 files may be stored on each disk, 992 on each hard disk platter.

### ISIS-II System Commands

ISIS-II system commands are designed to provide the user with a powerful, easy to use program and file manipulation capability. Several commands have the capability of operating on several files at once via the wildcard file naming convention. As an example, the command DELETE ★ .OBJ deletes all files in the disk directory with the suffix .OBJ. A summary of ISIS-II system commands is presented in Table 7-4.

**Call Capability** — The delete, rename, and attribute assignment commands, along with a set of file I/O routines, are callable from user written programs. This allows the user to open, close, read, and write disk files, access standard peripheral devices, write error messages, and load other programs via simple program call statements.

| Command | Operation |
|---|---|
| Initialize disk | Initializes a diskette for use by the system. Requires only one disk drive. |
| Attribute assignment | Assigns specified attributes to a file, such as write-protect. |
| Copy | Creates copies of existing diskette files or transfers files from one device to another. |
| Delete | Removes a file from the diskette, thereby freeing space for allocation of other files. |
| Directory | Lists name, size, and attributes of files from a specified diskette directory. |
| Rename | Allows diskette files to be renamed. |
| Format | Initializes a diskette for use by the system. (Use with two or more drives.) |
| Debug | Loads a specified program from a diskette into memory and then transfers control to the Intellec monitor for execution and or debugging. |
| Submit | Provides capability for executing a series of ISIS-II commands previously written to a diskette file. |

Additional commands are provided for support of the hard disk.

**Table 7-4. ISIS-II System Commands**

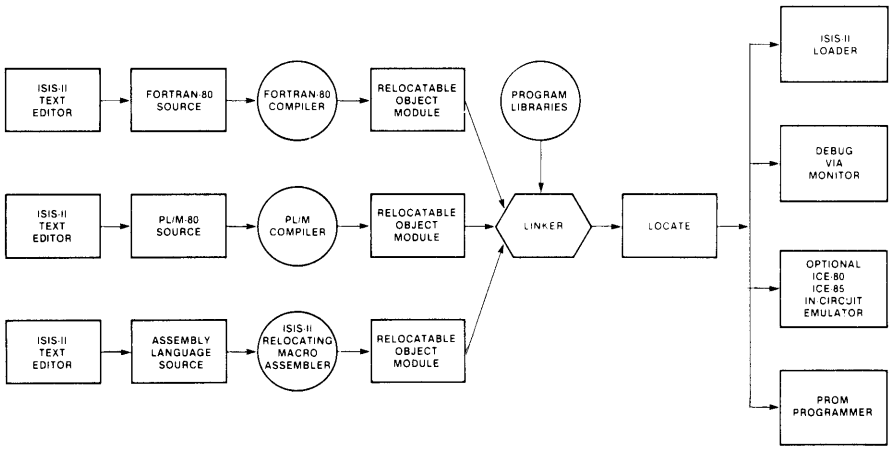



**Figure 7-6. Program Development Flow Using ISIS-II Disk Operating System**

## ISIS-II Text Editor

The ISIS-II text editor is a comprehensive tool for assembly language, PL/M, and FORTRAN program entry and correction for Intel microcomputers. Its command set allows either entire lines of text or individual characters to be manipulated within a line.

**Program Entry** — Programs may be entered from the console keyboard or may be loaded directly. Text is stored internally in the editor's workspace, and may be edited with the following commands:

- string insertion or deletion
- string search
- string substitution

**Utility Commands** — To facilitate the use of these editing commands, utility commands are used to change positions in the workspace. These include:

- move pointer by line or by character
- move pointer to start of workspace
- move pointer to end of workspace

**Storage** — The contents of the workspace are stored on diskette and can be immediately accessed by ISIS-II commands or other programs, such as the ISIS-II MCS-80/MCS-85 macroassembler.

For users with 64K of RAM memory, the CREDIT Text Editor is available. See CREDIT data sheet for more information.

## ISIS-II MCS-80/MCS-85™ Relocating Macroassembler

**Address Translation** — The ISIS-II MCS-80/MCS-85 macroassembler translates assembly language mnemonics into relocatable and/or absolute object code modules. In addition to eliminating the errors of hand translation, the ability to refer to program addresses with symbolic names makes it easy to modify programs by adding or deleting instructions. Extended macro capability eliminates the need to rewrite similar sections of code repeatedly, and thus simplifies program documentation. Conditional assembly permits the assembler to include or delete sections of code that may vary from system to system, such as the code required to handle optional external devices. Additionally, the user is allowed complete freedom in assigning the location of code, data, and stack segments.

**List File** — The ISIS-II Assembler accepts disk file input and produces a relocatable object file with corresponding symbol table and assembly listing file, including any error messages. A cross reference listing is also optionoptionally produced. The list file may then be examined from the system console or copied to a specified list device.

**Object File** — The relocatable object file generated by the assembler may be combined with other object programs residing on the disk to form a single relocatable object module or it can be converted to an absolute form for subsequent loading and execution.

## ISIS-II Linker

The ISIS-II linker provides the capability to combine the outputs of several independently compiled or assembled object modules (files) into a single relocatable object module. The linker automatically resolves all external program and data references during the linking process. Object modules produced from previous link operations may be easily linked to a new module. ISIS-II also provides facilities to ease the generation of overlays. An optional link map showing the contents and lengths of each segment in the output module can be requested. All unsatisfied external references are also listed. If requested by the user, the ISIS-II linker can search a specified set of program libraries for routines to be included in the output module.

## ISIS-II Object Locator

The ISIS-II locate program takes output from either the resident FORTRAN or PL/M compilers, the macroassembler, or the linker and transforms that output from relocatable format to an absolute format which may then be loaded via the standard ISIS-II loader, or loaded into an appropriate in-circuit emulator (ICE) module. During the locate process, code, data, and stack segments may be separately relocated, allowing code to be put in areas to be subsequently specified as ROM, while data and the stack are directed to RAM addresses. A locate map showing absolute addresses for each code and data segment and a symbol table dump listing symbols, attributes, and absolute address may also be requested.

## ISIS-II Library Manager™

The ISIS-II Library Manager program provides for the creation and maintenance of a program library containing Intel-provided and user-written programs and subroutines. These library routines may be linked to a program using the ISIS-II linker. Several libraries, each containing its own set of routines, may be created.

# ORDERING INFORMATION

| Part Number | Description |
|---|---|

Included with all Intellec Series II Microcomputer Development Systems.

Not available separately.

# PL/M-80
# HIGH LEVEL PROGRAMMING LANGUAGE
# INTELLEC® RESIDENT COMPILER

**Provides resident operation on Intellec® Microcomputer Development System and Intellec® Series II microcomputer development systems**

**Produces relocatable and linkable object code**

**Sophisticated code optimization reduces application memory requirements**

**Speeds project completion with increased programmer productivity**

**Cuts software development and maintenance costs**

**Improves product reliability with simplified language and consequent error reduction**

**Eases enhancement as system capabilities expand**

The PL/M-80 High Level Programming Language Intellec Resident Compiler is an advanced, high level programming language for Intel 8080 and 8085 microprocessors, iSBC-80 OEM computer systems, and Intellec microcomputer development systems. PL/M has been substantially enhanced since its introduction in 1973 and has become one of the most effective and powerful microprocessor systems implementation tools available. It is easy to learn, facilitates rapid program development and debugging, and significantly reduces maintenance costs. PL/M is an algorithmic language in which program statements naturally express the algorithm to be programmed, thus freeing programmers to concentrate on system development rather than assembly language details (such as register allocation, meanings of assembler mnemonics, etc.). The PL/M compiler efficiently converts free-form PL/M programs into equivalent 8080/8085 instructions. Substantially fewer PL/M statements are necessary for a given application than would be using assembly language or machine code. Since PL/M programs are problem oriented and thus more compact, programming in PL/M results in a high degree of productivity during development efforts, resulting in significant cost reduction in software development and maintenance for the user.

## FUNCTIONAL DESCRIPTION

The PL/M compiler is an efficient multiphase compiler that accepts source programs, translates them into object code, and produces requested listings. After compilation, the object program may be first linked to other modules, then located to a specific area of memory, and finally executed. The diagram shown in Figure 7-7 illustrates a program development cycle where the program consists of three modules: PL/M, FORTRAN, and assembly language. A typical PL/M compiler procedure is shown on the following page.

### Features

Major features of the Intel PL/M-80 compiler and programming language include:

**Resident Operation** — on Intellec microcomputer development systems eliminates the need for a large in-house computer or costly timesharing system.

**Object Code Generation** — of relocatable and linkable object codes permits PL/M program development and debugging in small modules, which may be easily linked with other modules and/or library routines to form a complete application.

**Extensive Code Optimization** — including compile time arithmetic, constant subscript resolution, and common subexpression elimination, results in generation of short, efficient CPU instruction sequences.

**Symbolic Debugging** — fully supported in the PL/M compiler and ICE-85 in-circuit emulators.

**Compile Time Options** — includes general listing format commands, symbol table listing, cross reference listing, and "innerlist" of generated assembly language instructions.

**Block Structure** — aids in utilization of structured programming techniques.

**Access** — provided by high level PL/M statements to hardware resources (interrupt systems, absolute addresses, CPU input/output ports).

**Data Definition** — enables complex data structures to be defined at a high level.

**Re-entrant Procedures** — may be specified as a user option.

### Benefits

PL/M is designed to be an efficient, cost-effective solution to the special requirements of microcomputer software development as illustrated by the following benefits of PL/M use:

**Low Learning Effort** — even for the novice programmer, because PL/M is easy to learn.

**Earlier Project Completion** — on critical projects, because PL/M substantially increases programmmer productivity while reducing program development time.

**Lower Development Cost** — because increased programmer productivity requiring less programming resources for a given function translates into lower software development costs.

**Increased Reliability** — because of PL/M's use of simple statements in the program algorithm, which are easier to correct and thus substantially reduce the risk of costly errors in systems that have already reached full production status.

**Easier Enhancement and Maintenance** — because programs written in PL/M are easier to read and easier to understand than assembly language, and thus are easier to enhance and maintain as system capabilities expand and future products are developed.

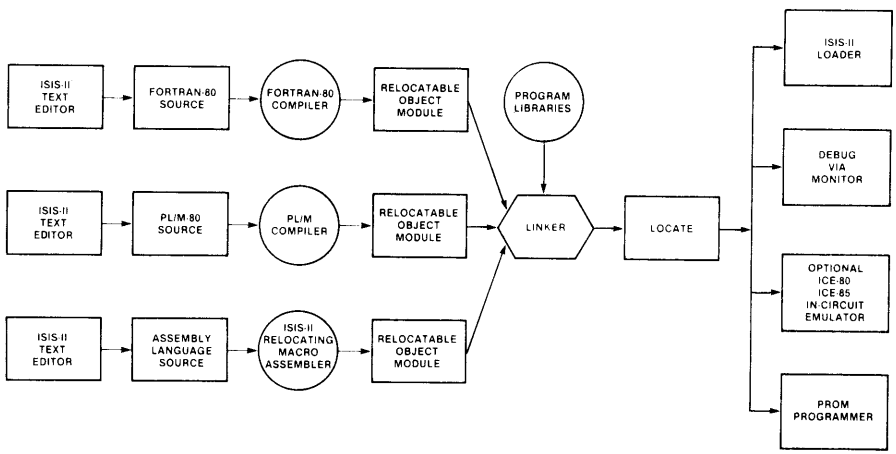



**Figure 7-7. Program Development Cycle Block Diagram**

**Simpler Project Development** — because the Intellec microcomputer development system with resident PL/M-80 is all that is needed for developing and debugging software for 8080 and 8085 microcomputers, and the use of expensive (and remote) timesharing or large computers is consequently not required.

```
                                        $OBJECT(:F1:FACT.OB2)
                                        $DEBUG
                                        $XREF
                                        $TITLE('FACTORIAL GENERATOR — PROCEDURE')
                                        $PAGEWIDTH(80)

 1                                      FACT:
                                        DO;

 2    1                                 DECLARE NUMCH BYTE PUBLIC;

 3    1                                 FACTORIAL: PROCEDURE (NUM,PTR) PUBLIC;
 4    2                                     DECLARE NUM BYTE, PTR ADDRESS;
 5    2                                     DECLARE DIGITS BASED PTR (161) BYTE;
 6    2                                     DECLARE (I,C,M) BYTE;

 7    2                                     NUMCH = 1; DIGITS(1) = 1;
 9    2                                     DO M = 1 TO NUM;
10    3                                         C = 0;
11    3                                         DO I = 1 TO NUMCH;
12    4                                             DIGITS(I) = DIGITS(I)*M + C;
13    4                                             C = DIGITS(I)/10;
14    4                                             DIGITS(I) = DIGITS(I) — 10*C;
15    4                                         END;

16    3                                         IF C<>0 THEN
17    3                                         DO;
18    4                                             NUMCH = NUMCH + 1; DIGITS(NUMCH) = C;
20    4                                             C = DIGITS(NUMCH)/10;
21    4                                             DIGITS(NUMCH) = DIGITS(NUMCH) — 10*C;
22    4                                         END
                                            END;

24    2                                 END FACTORIAL;

25    1                                 END;
```

**PL/M-80 Compiler Sample Factorial Generator Procedure**

## SPECIFICATIONS

### Operating Environment

**Required Hardware**

Intellec microcomputer development system
65K bytes of memory
Dual diskette drives
System console — teletype

**Optional Hardware**

CRT as system console
Line printer

**Required Software** — ISIS-II diskette operating system

### Shipping Media

Diskette

### Reference Manuals

**980026** — PL/M-80 Programming Manual (SUPPLIED)

**9800300** — ISIS-II PL/M-80 Compiler Operator's Manual (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

| Product Code | Description |
|---|---|
| MDS-PLM | High level language compiler |

# ICE-85™
# MCS-85™ IN-CIRCUIT EMULATOR

**Connects the Intellec® System Resources to the user-configured system via a 40-pin adaptor plug**

**Executes user system software in real-time**

**Allows user-configured system to share Intellec® memory and I/O facilities**

**Provides 1023 states of 8085 trace data plus 18 additional logic signals via an External Trace Module**

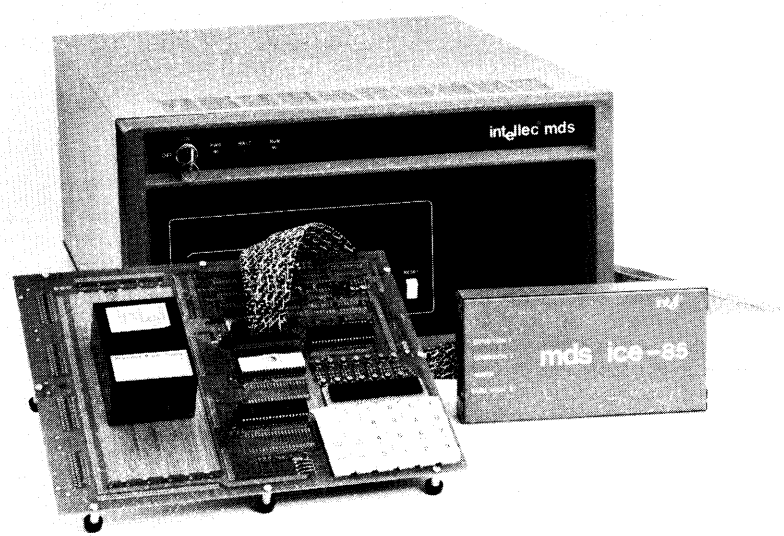
**Offers full symbolic debugging capability for both assembly language and Intel's high-level compiler languages PL/M-80 and FORTRAN-80**

**Displays trace data from the user's 8085 in assembler mnemonics and allows personality groupings of data sampled by the external 18-channel trace module**

**Extends ICE capabilities to the rest of the prototype system peripheral circuitry by allowing the user to execute his own peripheral chip analysis routines**

**Provides ability to examine and alter MCS-85™ registers, memory, flag values, interrupt bits and I/O ports**

The ICE-85 module resides in the Intellec® Microcomputer Development System and interfaces to the user system's 8085. In addition, an external trace module provides access to user system peripheral circuitry via a user-configured DIP clip for peripheral ICs or may be attached to as many as 18 separate prototype signal nodes via individual probe clips. Using the ICE-85 module, the designer can execute prototype software in real-time or single-step mode and can substitute Intellec® system memory and I/O for user system equivalent. ICE capability can be extended to the rest of the user system peripheral circuitry by allowing the user to create and execute a library of user-defined peripheral chip analyzer routines. All user access to the prototype system software may be done symbolically by assigning names to program locations and data, I/O ports and groups of external trace signals. For the first time, in-circuit emulation extends beyond the user's prototype CPU to the entire user's system, allowing In-System Emulation.



## SYMBOLIC DEBUGGING CAPABILITY

ICE-85 allows the user to make symbolic references to I/O ports, memory addresses and data in his program. Symbols and PL/M-80 statement number may be substituted for numeric values in any of the ICE-85 commands. The user is relieved from looking up addresses of variables or program subroutines.

The user symbol table generated along with the object file during a PL/M-80 or FORTRAN-80 compilation or by the ISIS-II 8080/8085 Macro Assembler is loaded into the Intellec® System 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. By referring to symbol memory addresses, the user can examine, change or break at the intended location.

ICE-85 provides symbolic definition of all 8085 registers, interrupt bits and flags. The following symbolic references are also provided for user convenience: TIMER, the low-order 16 bits of a register containing the number of 2 MHz clock pulses elapsed during emulation; HTIMER, the high-order 16 bits of the timer counter; PPC, the address of the last instruction emulated; BUFFERSIZE, the number of frames of valid trace data (between 0 and 1022).

## PERSONALITY GROUPED DISPLAYS

Trace data in the 1023 by 42-channel real-time trace memory buffer is displayed in easy to read format. The user has the option to specify trace data displays in actual 8085 assembler instruction mnemonics. The data collected from the External Trace Module can be grouped and symbolically named according to user specifications and displayed in the appropriate number base designation. Simple ICE-85 commands allow the user to select any portion of the 42K-bit trace buffer for immediate display.

## MEMORY AND I/O MAPPING

Memory and I/O for the user system can be resident in the user system or "borrowed" from the Intellec® System through ICE-85's mapping capability.

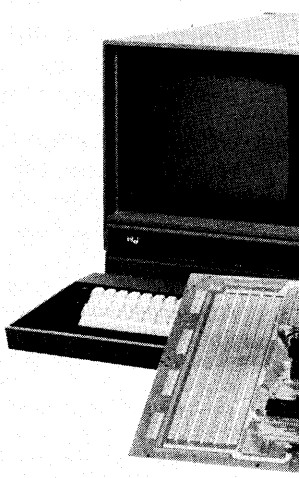
ICE-85 separates user memory into 32 2K blocks. Each block of memory can be defined independently. The user may assign Intellec® System equivalents to take the place of devices not yet designed for the user system during prototyping. In addition, Intellec® System memory or I/O can be accessed in place of suspect user system devices during prototyping or production checkout.

The user can also designate a block of memory or I/O as nonexistent. ICE-85 issues error messages when memory or I/O designated as nonexistent is accessed by the user program.

## INTEGRATED HARDWARE/SOFTWARE DEVELOPMENT

The user prototype need consist of no more than an 8085 CPU socket and a user bus to begin integration of software and hardware development efforts. Through ICE-85 mapping capabilities, Intellec® System equivalents can be accessed for missing prototype hardware. Hardware designs can be tested using the system software which will drive the final product.

The system integration phase, which can be so costly when attempting to mesh completed hardware and software products, becomes a convenient two-way debug tool when begun early in the design cycle.



## INTERROGATION AND UTILITY COMMANDS

| | |
|---|---|
| DISPLAY/ CHANGE | Display/Changes the values of symbols and the contents of 8085 registers, pseudo-registers, status flags, interrupt bits, I/O ports and memory. |
| EVALUATE | Displays the value of an expression in the binary, octal, decimal or hexadecimal. |
| SEARCH | Searches user memory between locations in a user program for specified contents. |
| CALL | *Emulates* a procedure starting at a specified memory address in user memory. |
| ICALL | *Executes* a user-supplied procedure starting at a specified memory address in the Intellec® System memory. |
| EXECUTE | Saves emulated program registers and emulates a user-supplied subroutine to access peripheral chips in the user's system. |

## REAL TIME TRACE

ICE-85 captures valuable trace information from the emulating CPU and the External Trace Module while the user is executing programs in real time. The 8085 status, the user memory or port addressed, the data read or written, the serial data lines and data from 18 external signals, is stored for the last 1023 machine states executed (511 machine cycles). This provides ample data for determining how the user system was reacting prior to emulation break. It is available whether the break was user-initiated or the result of an error condition.

For detailed information on the actions of CPU registers, flags, or other system operations, the user may operate in single or multi-step sequences tailored to system debug needs.

## EXTERNAL TRACE MODULE

TTL level signals from 18 points in the user system may be synchronously sampled by the External Trace Module and collected in ICE-85's trace buffer. The signals can be collected from a single peripheral chip via the supplied 40-pin DIP clip or may be placed by the user on up to 18 separate signal nodes using the supplied 18 individual probe clips. These signals are included in the 42-channel breakpoint comparisons and clock qualifiers. Also, data from these 18 channels may be displayed in each to read, user-defined groupings.

## SYNCHRONOUS OPERATION WITH OTHER DESIGN AIDS

ICE-85 can be synchronized with other Intellec® design aids by means of two external synchronization lines. These lines are used to enable and disable ICE-85 trace data collection and to cause break conditions based on an external signal which may not be included in the ICE-85 breakpoint registers. In addition, ICE-85 can generate signals on these lines which may be used to control other design aids.

## EMULATION CONTROLS AND COMMANDS

| | |
|---|---|
| GROUP | Defines into a symbolically named group, a channel or combination of channels from the 8085 Microprocessor and/or the External Trace Module. |
| GO | Initiates real-time emulation and controls emulation break conditions. |
| STEP | Initiates emulation in single instruction steps. User may specify the type and amount of information displayed following each step, and define conditions under which stepping should continue. |
| PRINT | Prints the user-specified portion of the trace memory to the selected list device. |

## BREAK REGISTERS/TRACE MEMORY

ICE-85 has two breakpoint registers which are used to break emulation, and two trace qualifier registers which are used to control the collection of trace data during emulation. Each register is 42 entries wide, one entry for each channel and each entry can take any one of the three values 0, 1 or "don't care."

The trace buffer, also 42 entries wide, collects data sampled from 24 8085 processor channels and 18 external channels sampled by the External Trace Module. The signals collected from the 8085 include address lines, data lines, status lines and series input and output lines. The 18 channels extending from the External Trace Module synchronously sample and collect into the trace buffer any user-specified TTL compatible signal from the rest of the prototype system. "Break" and "trace qualification" may therefore occur as a result of a match of any combination of up to 42 channels of CPU and external circuitry signals.

# SPECIFICATIONS

## ICE-85 Operating Environment

Required Hardware:
- Intellec® Microcomputer Development System
- System Console
- Intellec® Diskette Operating System
- ICE-85 Module

Required Software:
- System Monitor
- ISIS-II
- ICE-85 Software

## Equipment Supplied

18-Channel External Trace Module
Printed Circuit Boards (2)
Interface Cable and Emulation Buffer Module
Operator's Manual
ICE-85 Software, Diskette-Based Version

## Emulation Clock

User's system clock or ICE-85 adaptor socket (6.144 MHz Crystal)

## 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: | 6.00 lb (2.73 kg) |

## Electrical Characteristics

DC Power:
- $V_{CC} = +5V \pm 5\%$
- $I_{CC} = 12A$ maximum; 10A typical
- $V_{DD} = +12V \pm 5\%$
- $I_{DD} = 80$ mA maximum; 60 mA typical
- $V_{BB} = -10V \pm 5\%$
- $I_{BB} = 1$ mA maximum; 10 $\mu$A typical

## Environmental Characteristics

Operating Temperature: 0° to 40°C
Operating Humidity: Up to 95% relative humidity without condensation.

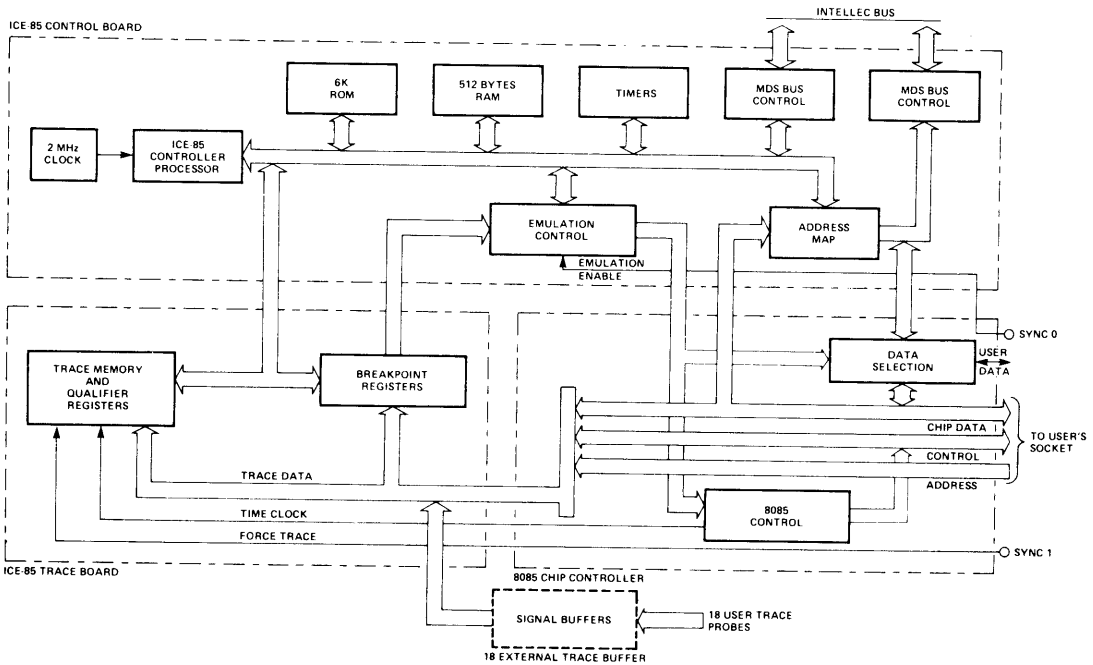



**Figure 7-8. ICE-85 Block Diagram**

## Ordering Information

| Part Number | Description |
|---|---|
| MDS-85-ICE | 8085 CPU In-Circuit Emulator and 18-Channel External Trace Module |

# SDK-85
# MCS-85™ SYSTEM DESIGN KIT

**Complete single board microcomputer system including CPU, memory, and I/O**

**Easy to assemble, low cost, kit form**

**Extensive system monitor software in ROM**

**Interactive LED display and keyboard**

**Large wire-wrap area for custom interfaces**

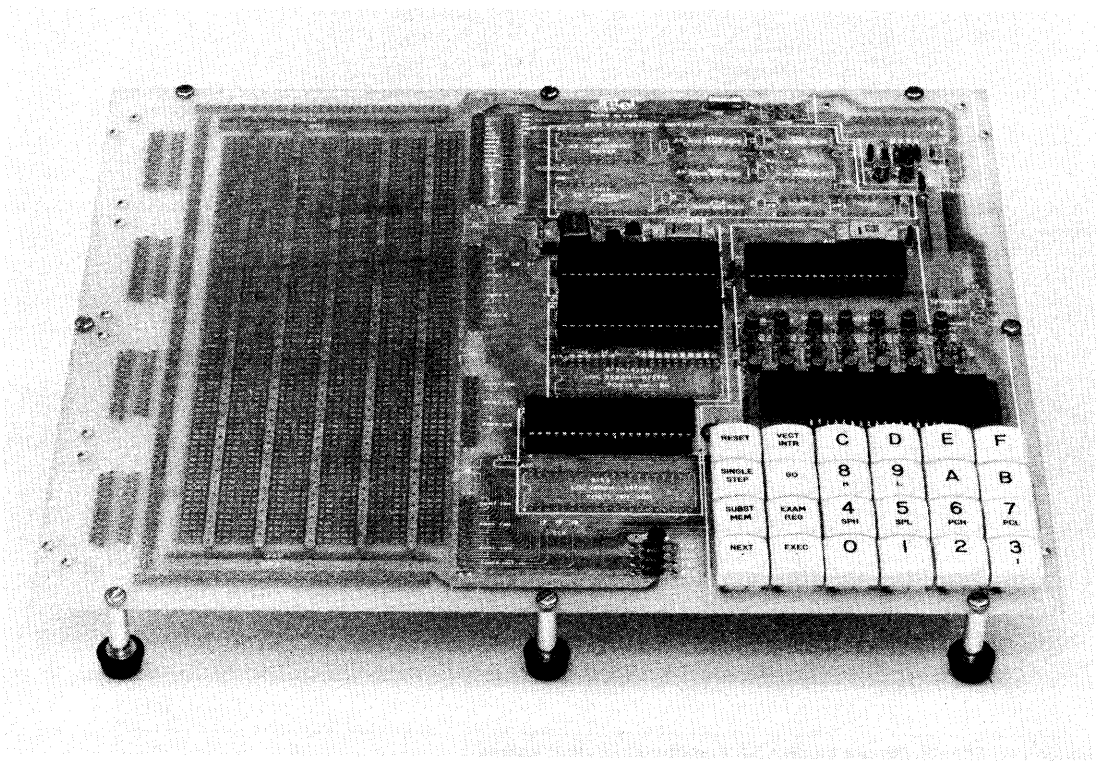
**Popular 8080A instruction set**

**Interfaces directly with TTY**

**High performance 3 MHz 8085A CPU (1.3 $\mu$s instruction cycle)**

**Comprehensive design library included**

The SDK-85 MCS-85 System Design Kit is a complete single board microcomputer system in kit form. It contains all components required to complete construction of the kit, including LED display, keyboard, resistors, caps, crystal, and miscellaneous hardware. Included is a preprogrammed ROM containing a system monitor for general software utilities and system diagnostics. The complete kit includes a 6-digit LED display and a 24-key keyboard for a direct insertion, examination, and execution of a user's program. In addition, it can be directly interfaced with a teletype terminal. The SDK-85 is an inexpensive, high performance prototype system that has designed-in flexibility for simple interface to the user's application.



## FUNCTIONAL DESCRIPTION

The SDK-85 is a complete 8085A microcomputer system on a single board, in kit form. It contains all necessary components to build a useful, functional system. Such items as resistors, capacitors, and sockets are included. Assembly time varies from three to five hours, depending on the skill of the user. The SDK-85 functional block diagram is shown in Figure 7-9.

### 8085A Processor

The SDK-85 is designed around Intel's 8085A microprocessor. The Intel 8085A is a new generation, complete 8-bit parallel central processing unit (CPU). Its instruction set is 100% software upward compatible with the 8080A microprocessor, and it is designed to improve the present 8080A's performance by higher system speed. Its high level of system integration allows a minimum system of three IC's: 8085A (CPU), 8156 (RAM), and 8355/8755 (ROM/PROM). A block diagram of the 8085A microprocessor is shown in Figure 7-10.

**System Integration** — The 8085A incorporates all of the features that the 8224 (clock generator) and 8228 (system controller) provided for the 8080A, thereby offering a high level of system integration.

**Addressing** — The 8085A uses a multiplexed data bus. The 16-bit address is split between the 8-bit address bus and the 8-bit address/data bus. The on-chip address latches of 8155/8156/8355/8755 memory products allows a direct interface with the 8085A.

### System Monitor

A compact but powerful system monitor is supplied with the SDK-85 to provide general software utilities and system diagnostics. It comes in a pre-programmed ROM.

### Communications Interface

The SDK-85 communicates with the outside world through either the on-board LED display/keyboard combination, or the user's TTY terminal (jumper selectable).

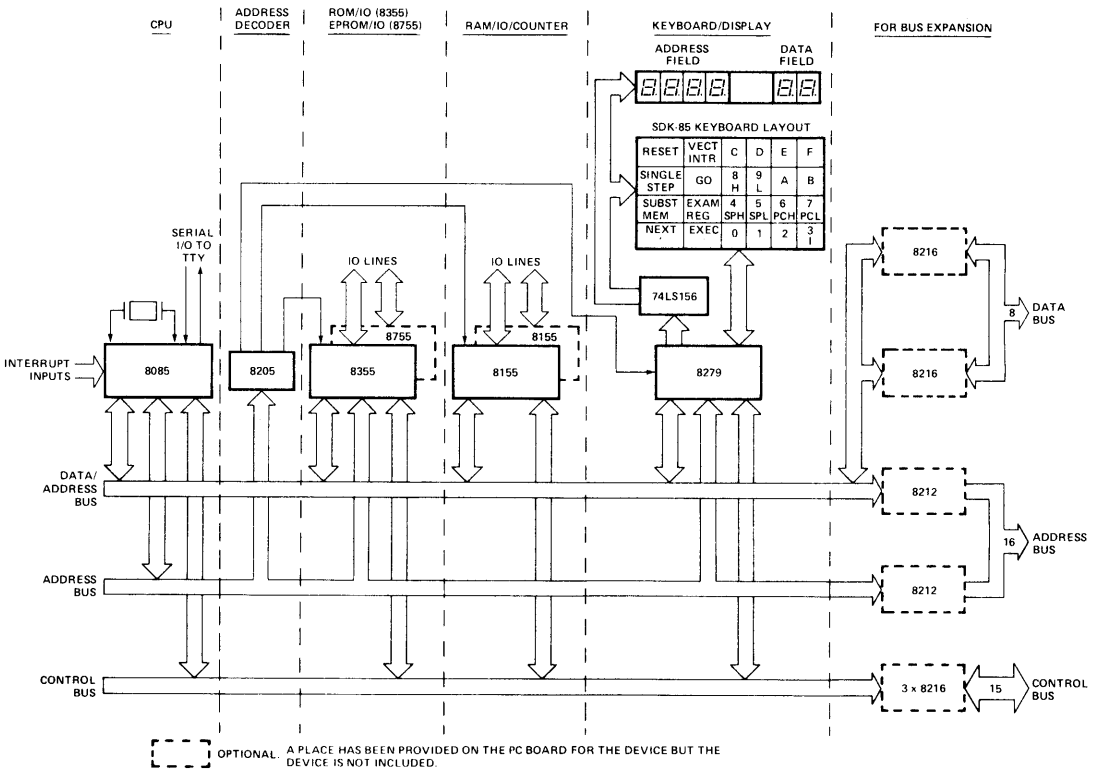
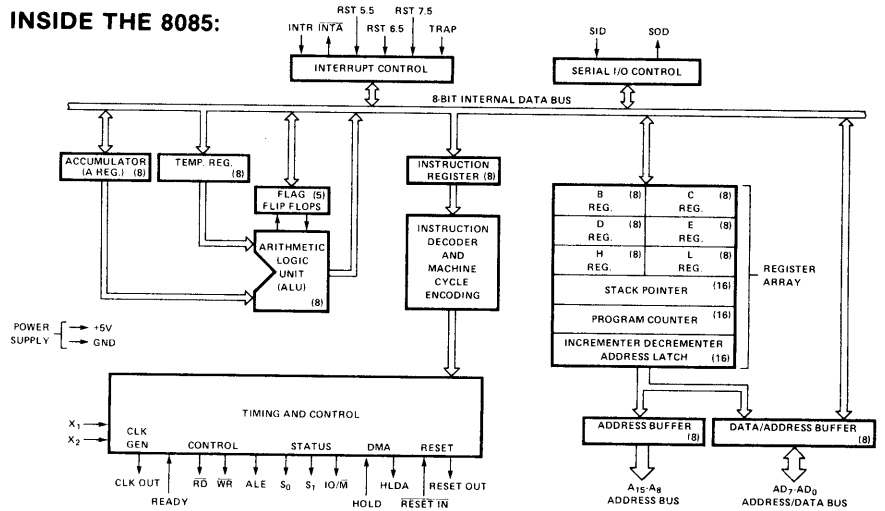



**Figure 7-9. SDK-85 System Design Kit Functional Block Diagram**




- SEVEN 8-BIT REGISTERS. SIX OF THEM CAN BE LINKED IN REGISTER PAIRS FOR CERTAIN OPERATIONS.
- 8-BIT ALU.
- 16-BIT STACK POINTER (STACK IS MAINTAINED OFFBOARD IN SYSTEM RAM MEMORY).
- 16-BIT PROGRAM COUNTER.

**Figure 7-10. 8085A Microprocessor Block Diagram**

Both memory and I/O can be easily expanded by simply soldering in additional devices in locations provided for this purpose. A large area of the board (45 sq. in.) is laid out as general purpose wire-wrap for the user's custom interfaces.

## Assembly

Only a few simple tools are required for assembly; soldering iron, cutters, screwdriver, etc. The SDK-85 user's manual contains step-by-step instructions for easy assembly without mistakes. Once construction is complete, the user connects his kit to a power supply and the SDK-85 is ready to go. The monitor starts immediately upon power-on or reset.

| Command | Operation |
|---|---|
| Reset | Starts monitor. |
| Go | Allows user to execute user program. |
| Single step | Allows user to execute user program one instruction at a time—useful for debugging. |
| Substitute memory | Allows user to examine and modify memory locations. |
| Examine register | Allows user to examine and modify 8085A's register contents. |
| Vector interrupt | Serves as user interrupt button. |

**Table 7-5. Keyboard Monitor Commands**

**Commands** — Keyboard monitor commands and teletype monitor commands are provided in Table 7-5 and Table 7-6, respectively.

| Command | Operation |
|---|---|
| Display memory | Displays multiple memory locations. |
| Substitute memory | Allows user to examine and modify memory locations one at a time. |
| Insert instructions | Allows user to store multiple bytes in memory. |
| Move memory | Allows user to move blocks of data in memory. |
| Examine register | Allows user to examine and modify the 8085A's register contents. |
| Go | Allows user to execute user programs. |

**Table 7-6. Teletype Monitor Commands**

## Documentation

In addition to detailed information on using the monitors, the SDK-85 user's manual provides circuit diagrams, a monitor listing, and a description of how the system works. The complete design library for the SDK-85 is shown in Figure 7-11 and listed in the Specifications section under Reference Manuals.


**Figure 7-11. SDK-85 Design Library**

## 8085A INSTRUCTION SET

Table 7-7 contains a summary of processor instructions used for the 8085A microprocessor.

| Mnemonic[1] | Description | Instruction Code[2] $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | Clock[3] Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| **MOVE, LOAD, AND STORE** | | | | | | | | | | |
| MOVr1r2 | Move register to register | 0 | 1 | D | D | D | S | S | S | 4 |
| MOV M,r | Move register to memory | 0 | 1 | 1 | 1 | 0 | S | S | S | 7 |
| MOV r,M | Move memory to register | 0 | 1 | D | D | D | 1 | 1 | 0 | 7 |
| MVI r | Move immediate register | 0 | 0 | D | D | D | 1 | 1 | 0 | 7 |
| MVI M | Move immediate memory | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 10 |
| LXI B | Load immediate register Pair B & C | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| LXI D | Load immediate register Pair D & E | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| LXI H | Load immediate register Pair H & L | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| **STAX B** | **Store A indirect** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | 7 |
| **STAX D** | **Store A indirect** | **0** | **0** | **0** | **1** | **0** | **0** | **1** | **0** | 7 |
| LDAX B | Load A indirect | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 7 |
| LDAX D | Load A indirect | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 7 |
| STA | Store A direct | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 13 |
| LDA | Load A direct | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 13 |
| SHLD | Store H & L direct | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 16 |
| LHLD | Load H & L direct | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 16 |
| XCHG | Exchange D & E, H & L registers | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 4 |
| **STACK OPS** | | | | | | | | | | |
| PUSH B | Push register pair B & C on stack | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 12 |
| PUSH D | Push register pair D & E on stack | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 12 |
| PUSH H | Push register pair H & L on stack | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 12 |
| PUSH PSW | Push A and flags on stack | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 12 |
| POP B | Pop register pair B & C off stack | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP D | Pop register pair D & E off stack | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 10 |
| POP H | Pop register pair H & L off stack | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 10 |
| POP PSW | Pop A and flags off stack | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| XTHL | Exchange top of stack, H & L | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 16 |
| SPHL | H & L to stack pointer | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 6 |
| LXI SP | Load immediate stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 10 |
| INX SP | Increment stack pointer | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 6 |
| DCX SP | Decrement stack pointer | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 6 |
| **JUMP** | | | | | | | | | | |
| JMP | Jump unconditional | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 10 |
| JC | Jump on carry | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 7/10 |
| JNC | Jump on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 7/10 |
| JZ | Jump on zero | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 7/10 |
| JNZ | Jump on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 7/10 |
| JP | Jump on positive | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 7/10 |
| JM | Jump on minus | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 7/10 |
| JPE | Jump on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 7/10 |
| JPO | Jump on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 7/10 |
| PCHL | H & L to program counter | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 6 |
| **CALL** | | | | | | | | | | |
| CALL | Call unconditional | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 18 |
| CC | Call on carry | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 9/18 |
| CNC | Call on no carry | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 9/18 |
| CZ | Call on zero | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 9/18 |
| CNZ | Call on no zero | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 9/18 |
| CP | Call on positive | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 9/18 |
| CM | Call on minus | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 9/18 |
| CPE | Call on parity even | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 9/18 |
| CPO | Call on parity odd | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 9/18 |
| **RETURN** | | | | | | | | | | |
| RET | Return | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| RC | Return on carry | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 6/12 |
| RNC | Return on no carry | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 6/12 |
| RZ | Return on zero | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 6/12 |
| RNZ | Return on no zero | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 6/12 |
| RP | Return on positive | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 6/12 |
| RM | Return on minus | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 6/12 |

continued

| Mnemonic[1] | Description | Instruction Code[2] $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | Clock[3] Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| RPE | Return on parity even | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 6/12 |
| RPO | Return on parity odd | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 6/12 |
| **RESTART** | | | | | | | | | | |
| RST | Restart | 1 | 1 | A | A | A | 1 | 1 | 1 | 12 |
| **INCREMENT AND DECREMENT** | | | | | | | | | | |
| INR r | Increment register | 0 | 0 | D | D | D | 1 | 0 | 0 | 4 |
| DCR r | Decrement register | 0 | 0 | D | D | D | 1 | 0 | 1 | 4 |
| INR M | Increment memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 10 |
| DCR M | Decrement memory | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 10 |
| INX B | Increment B & C registers | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 6 |
| INX D | Increment D & E registers | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 6 |
| INX H | Increment H & L registers | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 6 |
| DCX B | Decrement B & C | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 6 |
| DCX D | Decrement D & E | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 6 |
| DCX H | Decrement H & L | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 6 |
| **ADD** | | | | | | | | | | |
| ADD r | Add register to A | 1 | 0 | 0 | 0 | 0 | S | S | S | 4 |
| ADC r | Add register to A with carry | 1 | 0 | 0 | 0 | 1 | S | S | S | 4 |
| ADD M | Add memory to A | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ADC M | Add memory to A with carry | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| ADI | Add immediate to A | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 7 |
| ACI | Add immediate to A with carry | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 7 |
| DAD B | Add B & C to H & L | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD D | Add D & E to H & L | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 10 |
| DAD H | Add H & L to H & L | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 10 |
| DAD SP | Add stack pointer to H & L | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 10 |
| **SUBTRACT** | | | | | | | | | | |
| SUB r | Subtract register from A | 1 | 0 | 0 | 1 | 0 | S | S | S | 4 |
| SBB r | Subtract register from A with borrow | 1 | 0 | 0 | 1 | 1 | S | S | S | 4 |
| SUB M | Subtract memory from A | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| SBB M | Subtract memory from A with borrow | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |
| SUI | Subtract immediate from A | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 7 |
| SBI | Subtract immediate from A with borrow | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 7 |

| Mnemonic[1] | Description | Instruction Code[2] $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | Clock[3] Cycles |
|---|---|---|---|---|---|---|---|---|---|---|
| **LOGICAL** | | | | | | | | | | |
| ANA r | And register with A | 1 | 0 | 1 | 0 | 0 | S | S | S | 4 |
| XRA r | Exclusive Or register with A | 1 | 0 | 1 | 0 | 1 | S | S | S | 4 |
| ORA r | Or register with A | 1 | 0 | 1 | 1 | 0 | S | S | S | 4 |
| CMP r | Compare register with A | 1 | 0 | 1 | 1 | 1 | S | S | S | 4 |
| ANA M | And memory with A | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| XRA M | Exclusive Or memory with A | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| ORA M | Or memory with A | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| CMP M | Compare memory with A | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| ANI | And immediate with A | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 7 |
| XRI | Exclusive Or immediate with A | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 7 |
| ORI | Or immediate with A | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 7 |
| CPI | Compare immediate with A | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 7 |
| **ROTATE** | | | | | | | | | | |
| RLC | Rotate A left | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 4 |
| RRC | Rotate A right | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 4 |
| RAL | Rotate A left through carry | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 4 |
| RAR | Rotate A right through carry | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 4 |
| **SPECIALS** | | | | | | | | | | |
| CMA | Complement A | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 4 |
| STC | Set carry | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 4 |
| CMC | Complement carry | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 4 |
| DAA | Decimal adjust A | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 4 |
| **INPUT/OUTPUT** | | | | | | | | | | |
| IN | Input | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 10 |
| OUT | Output | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 10 |
| **CONTROL** | | | | | | | | | | |
| EI | Enable interrupts | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 4 |
| DI | Disable interrupts | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 4 |
| NOP | No-operation | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 4 |
| HLT | Halt | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 5 |
| **NEW 8085 INSTRUCTIONS** | | | | | | | | | | |
| RIM | Read interrupt mask | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 4 |
| SIM | Set interrupt mask | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 4 |

**Notes**

1. All mnemonics copyright © Intel Corporation 1977.
2. DDD or SSS: B = 000, C = 001, D = 010, E = 011, H = 100, L = 101, Memory = 110, A = 111.
3. Two possible cycle times. (6/12) indicates instruction cycles dependent on condition flags.

**Table 7-7. Summary of 8085A Processor Instructions**

# SPECIFICATIONS

## Central Processor

**CPU** — 8085A

**Instruction Cycle** — 1.3 μs

**Tcy** — 330 ns

## Memory

**ROM** — 2K bytes (expandable to 4K bytes) 8355/8755A

**RAM** — 256 bytes (expandable to 512 bytes) 8155

## Addressing

**ROM** — 0000-07FF (expendable to 0FFF with an additional 8355/8755A)

**RAM** — 2000-20FF (2800-28FF available with an additional 8155)

**Note**

The wire-wrap area of the SDK-85 PC board may be used for additional custom memory expansion up to the 64K-byte addressing limit of the 8085A.

## Input/Output

**Parallel** — 38 lines (expandable to 76 lines)

**Serial** — Through SID/SOD ports of 8085A. Software generated baud rate.

**Baud Rate** — 110

## Interfaces

**Bus** — All signals TTL compatible
**Parallel I/O** — All signals TTL compatible
**Serial I/O** — 20 mA current loop TTY

**Note**
By populating the buffer area of the board, the user has access to all bus signals that enable him to design custom system expansions into the kit's wire-wrap area.

## Interrupts

**Three Levels**
(RST 7.5) — Keyboard interrupt
(RST 6.5) — TTL input
(INTR) — TTL input

## DMA

**Hold Request** — Jumper selectable. TTL compatible input.

## Software

**System Monitor** — Pre-programmed 8755A or 8355 ROM
**Addresses** — 0000-07FF
**Monitor I/O** — Keyboard/display or TTY (serial I/O)

## Physical Characteristics

**Width** — 12.0 in. (30.5 cm)
**Height** — 10 in. (25.4 cm)
**Depth** — 0.50 in. (1.27 cm)
**Weight** — approx. 12 oz

## Electrical Characteristics

**DC Power Requirement** (power supply not included in kit)

| Voltage | Current |
|---|---|
| $V_{CC}$ 5V ± 5% | 1.3A |
| $V_{TTY}$ – 10V ± 10% | 0.3A ($V_{TTY}$ required only if teletype is connected) |

## Environmental Characteristics

**Operating Temperature** — 0-55 °C

## Reference Manuals

**9800451** — SDK-85 User's Manual (SUPPLIED)

**9800366** — MCS-85 User's Manual (SUPPLIED)

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

8085/8080 Assembly Language Reference Card (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 |
|---|---|
| SDK-85 | MCS-85 system design kit |

# FORTRAN-80
# 8080/8085 ANS FORTRAN 77
# INTELLEC® RESIDENT COMPILER

**Meets and exceeds ANS FORTRAN 77 Subset Language Specification**

**Supports Intel Floating Point Standard with either the floating point support library or the iSBC-310 High Speed Mathematics Board**

**Resident operation on Intellec® Microcomputer Development System and Intellec® Series II Microcomputer Development System**

**Supports full symbolic debugging with ICE-80™ and ICE-85™**

**Produces relocatable and linkable object code compatible with resident PL/M-80 and 8080/8085 Macro Assembler**

**Full FORTRAN 77 language I/O support or optional RMX-80 run-time library**

**Well defined I/O interface for configuration with user-supplied drivers**

**Sophisticated code optimization insures efficient program implementation**

FORTRAN-80 is a computer industry-standard, high-level programming language and compiler that translates FORTRAN statements into relocatable object modules. When the object modules are linked together and located into absolute program modules, they are suitable for execution on Intel® 8080/8085 Microprocessors, iSBC-80 OEM Computer Systems, and Intellec® Microcomputer Development Systems. FORTRAN-80 meets and exceeds the ANS FORTRAN 77 Language Subset Specification[1]. The compiler operates on the Intellec Microcomputer Development System under the ISIS-II Disk Operating Systems and produces efficient relocatable object modules that are compatible for linkage with PL/M-80 and 8080/8085 Macro Assembler modules.

The ANS FORTRAN 77 language specification offers many powerful extensions to the FORTRAN language that are especially well suited to Intel® 8080/8085 Microprocessor software development. Because FORTRAN-80 conforms to the ANS FORTRAN 77 standard, the user is assured of compatibility with existing FORTRAN software that meets the standard as well as a guarantee of upward compatibility to other computer systems supporting an ANS FORTRAN 77 Compiler.

[1] ANSI X3J3/90

## FORTRAN-80 LANGUAGE FEATURES

Major ANS FORTRAN 77 features supported by the Intel® FORTRAN-80 Programming Language include:

- Structured Programming is supported with the IF ... THEN ... ELSE IF ... ELSE ... END IF constructs.
- CHARACTER data type permits alphanumeric data to be handled as strings rather than characters stored in array elements.
- Full I/O capabilities include:
  - — Sequential and Direct Access files
  - — Error handling facilities
  - — Formatted, Free-formatted, and Unformatted data representation
  - — Internal (in-memory) file units provide capability to format and reformat data in internal memory buffers
  - — List Directed Formatting
- Supports arrays of up to seven dimensions.
- Supports logical operators

  .EQV. — Logical equivalence

  .NEQV. — Logical nonequivalence

Major extensions to FORTRAN 77 in Intel FORTRAN-80 include:

- Direct 8080/8085 port I/O supported by intrinsic subroutines.
- Binary and Hexadecimal integer constants.
- Well defined interface to FORTRAN-80 I/O statements (READ, OPEN, etc.), allowing easy use of user-supplied I/O drivers.
- User-defined INTEGER storage lengths of 1, 2 or 4 bytes.
- User-defined LOGICAL storage lengths of 1, 2 or 4 bytes.
- REAL STORAGE lengths of 4 bytes.
- Bitwise Boolean operations using logical operators on integer values.
- Hollerith data constants.
- Implicit extension of the length of an integer or logical expression to the length of the left-hand side in an assignment statement.
- A format descriptor to suppress carriage return on a terminal output device at the end of the record.

## FORTRAN-80 COMPILER FEATURES

- Supports multiple compilation units in single source file.
- Optional Assembly Language code listing.
- Comprehensive cross-reference, symbol attribute and error listing.
- Compiler controls and directives are compatible with other Intel language translators.
- Optional Reentrancy.
- User-defined default storage lengths.
- Optional FORTRAN 66 Do Loop semantics.
- Source files may be prepared in free format.
- The INCLUDE control permits specified source files to be combined into a compilation unit at compile time.
- Transparent interface for software and hardware floating point support, allowing either to be chosen at time of linking.

## FORTRAN-80 BENEFITS

FORTRAN-80 provides a means of developing application software for Intel® MCS-80/85 products in a familiar, widely accepted, and computer industry-standardized programming language. FORTRAN-80 will greatly enhance the user's ability to provide cost-effective solutions to software development for Intel microprocessors as illustrated by the following:

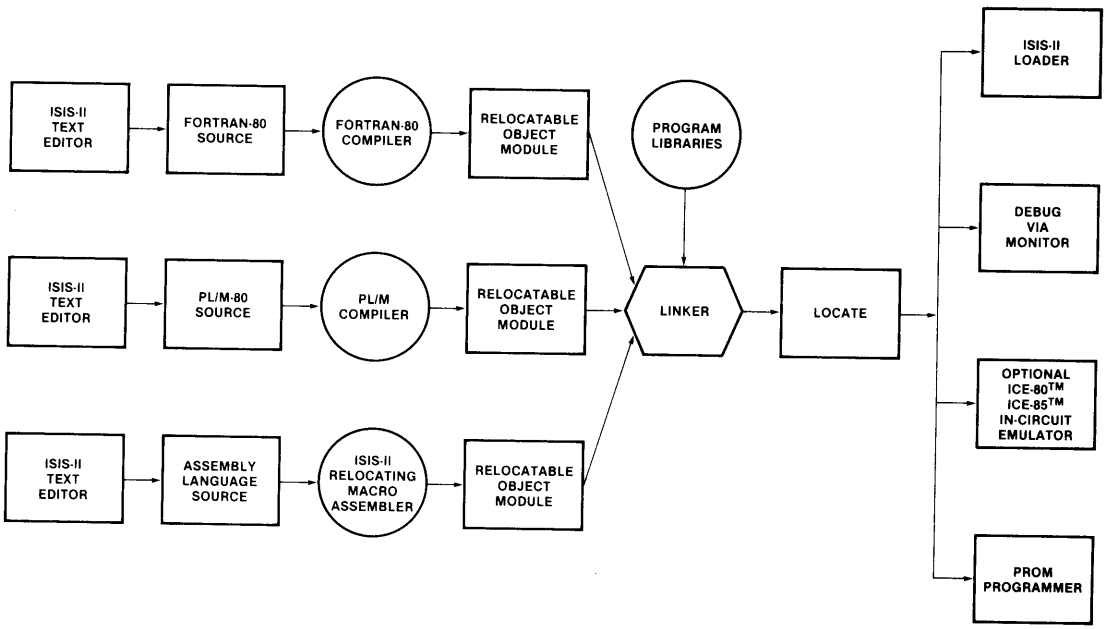
- *Completely Complementary to Existing Intel Software Design Tools* — Object modules are linkable with new or existing Assembly Language and PL/M Modules.
- *Incremental Runtime Library Support* — Runtime overhead is limited only to facilities required by the program.
- *Low Learning Effort* — FORTRAN-80, like PL/M, is easy to learn and use. Existing FORTRAN software can be ported to FORTRAN-80, and programs developed in FORTRAN-80 can be run on any other computer with ANS FORTRAN 77.
- *Earlier Project Completion* — Critical projects are completed earlier than otherwise possible because FORTRAN-80 will substantially increase programmer productivity, and is complementary to PL/M Modules by providing comprehensive arithmetic, I/O formatting, and data management support in the language.
- *Lower Development Cost* — Increases in programmer productivity translates into lower software development costs because less programming resources are required for a given function.
- *Increased Reliability* — The nature of high-level languages, including FORTAN-80, is that they lend themselves to simple statements of the program algorithm. This substantially reduces the risk of costly errors in systems that have already reached production status.
- *Easier Enhancements and Maintenance* — Like PL/M, program modules written in FORTRAN-80 are easier to read and understand than assembly language. This means it is easier to enhance and maintain FORTRAN-80 programs as system capabilities expand and future products are developed.
- *Comprehensive, Yet Simple Project Development* — The Intellec Microcomputer Development System, with the 8080/8085 Macro Assembler, PL/M-80 and FORTRAN-80 is the most comprehensive software design facility available for the Intel MCS-80/85 Microprocessor family. This reduces development time and cost because expensive (and remote) timesharing or large computers are not required.

## SAMPLE FORTRAN-80 SOURCE PROGRAM LISTING

```
*    **  THIS PROGRAM IS AN EXAMPLE OF ISIS-II FORTRAN-80 THAT
*    **  CONVERTS TEMPERATURE BETWEEN CELSIUS AND FARENHEIT

      PROGRAM CONVRT

      CHARACTER*1 CHOICE, SCALE

      PRINT 100
*     **  ENTER CONVERSION SCALE (C OR F)
10    PRINT 200
      READ (5,300) SCALE

        IF (SCALE .EQ. 'C')
     +     THEN
              PRINT 400
*             **  ENTER THE NUMBER OF DEGREES FARENHEIT
              READ (5,*) DEGF
              DEGC = 5./9.*(DEGF-32)
*             **  PRINT THE ANSWER
              WRITE (6,500) DEGF,DEGC
*             **  RUN AGAIN?
20            PRINT 600
              READ (5,300) CHOICE
                 IF (CHOICE .EQ. 'Y')
     +              THEN
                       GOTO 10
                 ELSE IF (CHOICE .EQ. 'N')
     +              THEN
                       CALL EXIT
                 ELSE
                    GOTO 20
                 END IF
        ELSE IF (SCALE .EQ. 'F')
     +     THEN
*             **  CONVERT FROM FARENHEIT TO CELSIUS
              PRINT 700
              READ (5,*) DEGC
              DEGF = 9./5.*DEGC+32.
*             **  PRINT THE ANSWER
              WRITE (6,800) DEGC,DEGF
              GOTO 20
        ELSE
*          **  NOT A VALID ENTRY FOR THE SCALE
           WRITE (6,900) SCALE
           GOTO 10
        END IF
100   FORMAT(' TEMPERATURE CONVERSION PROGRAM',//,
     +' TYPE C FOR FARENHEIT TO CELSIUS OR',/,
     +' TYPE F FOR CELSIUS TO FARENHEIT',//)
200   FORMAT(/,' CONVERSION? ',$)
300   FORMAT(A1)
400   FORMAT(/,'ENTER DEGREES FARENHEIT: ',$)
500   FORMAT(/,F7.2,  DEGREES FARENHEIT = ',F7.2,' DEGREES CELSIUS')
600   FORMAT(/,' AGAIN (Y OR N)? ',$)
700   FORMAT(/,' ENTER DEGREES CELSIUS: ',$)
800   FORMAT(/,F7.2,' DEGREES CELSIUS = ',F7.2,' DEGREES FARENHEIT',/)
900   FORMAT(/,1H ,A1,' NOT A VALID CHOICE - TRY AGAIN!',/)
      END
```

# FORTRAN-80

The FORTRAN-80 Compiler is an efficient, multiphase compiler that accepts source programs, translates them into relocatable object code, and produces requested listings. After compilation, the object program may be linked to other modules, located to a specific area of memory, then executed. The diagram shown below illustrates a program development cycle where the program consists of modules created by FORTRAN-80, PL/M-80 and the 8080/8085 Macro Assembler.




## SPECIFICATIONS

### OPERATING ENVIRONMENT

Required Hardware:

Intellec* Microcomputer Development System
- — MDS-800, MDS-888
- — Series II Model 220, Model 230

64K bytes of RAM memory

Dual diskette drives
- — Single or Double Density

System console
- — CRT or hardcopy interactive device

Optional Hardware:
Line Printer
ICE-80™, ICE-85™

Required Software:
ISIS-II Diskette Operating System
- — Single or Double Density

Optional Software:
iSBC-801 FORTRAN-80 Run-Time Software Package for RMX-80

### DOCUMENTATION PACKAGE

FORTRAN-80 Programming Manual (9800481)

ISIS-II FORTRAN-80 Compiler Operator's Manual (9800480)

FORTRAN-80 Programming Reference Card (9800547)

### SHIPPING MEDIA

Flexible Diskettes
- — Single and Double Density

## ORDERING INFORMATION

| PRODUCT CODE | DESCRIPTION |
|---|---|
| MDS-301 | FORTRAN-80 Compiler for Intellec Microcomputer Development Systems |

# BASIC-80
# EXTENDED ANS 1978 BASIC
# INTELLEC® RESIDENT INTERPRETER

**Meets ANS 1978 standard for minimal BASIC and adds many powerful extensions**

**Operates under the ISIS-II operating system on Intellec and Intellec® Series-II Microcomputer Development Systems**

**Full sequential and random disk file I/O with ISIS-II**

**Applications range from prototyping microcomputer software to inexpensive engineering and management problem solving on the Intellec® systems**

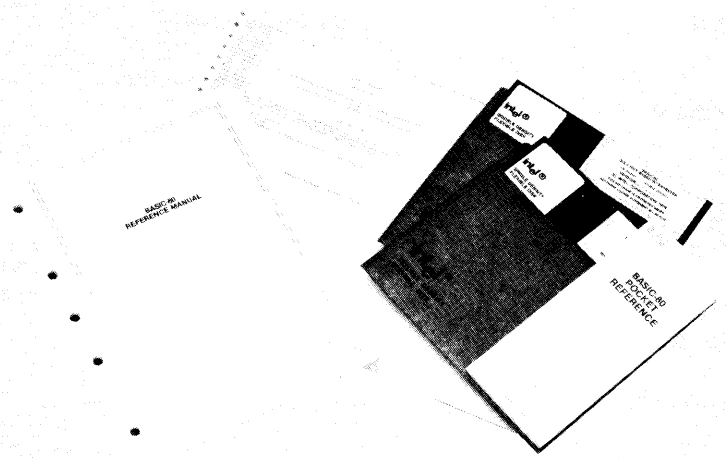
**Supports the Intel floating point standard and provides integer and string data types**

**Can call user subroutines written in FORTRAN-80, PL/M-80, and 8080/85 macro assembler that are resident in the Intellec® memory**

**Easily learned language and interactive environment combine to provide a flexible and powerful facility for developing programs to run on the Intellec® Microcomputer Development Systems**

BASIC is an industry standard, high-level programming language which is designed to be easily learned and used by novices and experienced programmers alike. The interpreter provides an interactive environment which allows fast and easy program development, testing, and debugging. BASIC is widely used for problem solving in engineering and management; extensive software exists for business applications such as order entry, accounts receivable, accounts payable, and inventory control, and engineering applications such as numeric and statistical analysis.

Intel's BASIC-80 meets the standards of ANS 1978 BASIC and extends them to take advantage of the software development capabilities of the Intellec Microcomputer Development Systems. The matching of these resources with the ease of programming in BASIC-80 provides a very effective tool for both microprocessor systems development and inexpensive applications programming and problem solving on the Intellec systems.



## BASIC-80 LANGUAGE FEATURES

Standard ANS 78 BASIC features, all supported by BASIC-80, include:

- String and numeric constants, variables, and arrays.
- FOR . . . TO . . . STEP . . . NEXT statements for loop execution.
- IF . . . THEN statements for conditional execution.
- ON . . . GOTO statements for computed branching.
- GOSUB/RETURN subroutine calls and returns.
- Built in scientific functions:

| | | |
|---|---|---|
| ABS | RND | TAN |
| EXP | SGN | COS |
| INT | SQR | SIN |
| LOG | ATN | |

- User defined single statement functions.

Major extensions to ANS 78 BASIC which BASIC-80 provides include:

- Support for the Intel single and double precision floating point standard.
- Disk file I/O, supporting both random access and sequential access files.
- Direct read and write to CPU I/O ports through the INP and OUT functions.
- Direct memory read and write through the PEEK and POKE functions.
- Calls to user-supplied external subroutines, which may have been written in FORTRAN-80, PL/M-80, or 8080/8085 Assembly Language and have been located at absolute memory locations using the ISIS-II facilities.
- User directed error trapping and handling functions.
- Program execution trace command.
- Formatted print statement with the PRINT USING function.
- ELSE clause for IF . . . THEN statements.
- Matrices with up to 110 dimensions.
- Extensive string manipulation functions.
- Boolean operators.
- Type conversion functions—integer, floating point, and character.

## BENEFITS OF BASIC-80

- Added Value to the Intellec Systems—with BASIC-80 the Intellec Microcomputer Development Systems can be effectively used in many engineering and management applications.
- Inexpensive and Accessible Computational Facility—the ease of use and flexibility inherent in BASIC-80 and its interpretive environment fit well with the "at hand" computational resources of the Intellec systems. The combination is a particularly useful tool for obtaining fast and accurate results.
- Easy to Learn—the language is designed to be easily understood and learned. Results are obtained faster and people who may benefit from using the system can do so easily.
- Aid in Microcomputer Software Design—microcomputer software can be prototyped in BASIC-80 to inexpensively develop and test program logic.
- Complemented by Existing Software—subroutines written in PL/M-80, FORTRAN-80, and ASM 8080/85 can be called from BASIC-80 programs.
- Easy to Enhance and Maintain—BASIC-80, being straightforward and easily understood, provides for programs that are easy to maintain and modify in the future.

## SPECIFICATIONS

### Operating Environment

**Required Hardware:**

Intellec Microcomputer Development System
— MDS-800, MDS-888
— Series-II Model 220, Model 230

48K bytes of RAM memory

Diskette drive
— Single or double density

System console
— CRT or hard copy interactive device

**Optional Hardware:**

Line printer
Additional diskette drive

**Required Software:**

ISIS-II Diskette Operating System
— Single or double density

**Documentation Package:**

Basic-80 Reference Manual (9800758A)
Basic-80 Programming Reference Card (9800774)

**Shipping Media:**

Flexible diskettes
— Single and double density

## EXAMPLE BASIC-80 PROGRAM

```
list
10 PRINT "THIS PROGRAM CALCULATES THE MEAN AND STANDARD"
20 PRINT " DEVIATION OF INPUT DATA"
30 S=0:V=0
40 INPUT "NUMBER OF VALUES";N
50 FOR I=1 TO N
60 INPUT A(I)
70 S=S+A(I)
80 NEXT
90 S=S/N
100 REM CALCULATION OF VARIANCE
110 FOR I=1 TO N
120 V=V+(A(I)-S)^2/N
130 NEXT
140 SD=SQR(V)
150 PRINT "MEAN=";S
160 PRINT "STANDARD DEVIATION IS=";SD
Ok

run
THIS PROGRAM CALCULATES THE MEAN AND STANDARD
 DEVIATION OF INPUT DATA
NUMBER OF VALUES? 6
? 34.7
? 32.9
? 38.2
? 35
? 37.6
? 40.9
MEAN= 36.55
STANDARD DEVIATION IS= 2.642442
Ok
```

## ORDERING INFORMATION

| Product Code | Description |
|---|---|
| MDS-320 | ISIS-II BASIC-80<br>Disk-Based Interpreter |

# iCIS-COBOL™ SOFTWARE PACKAGE

**Meets and exceeds minimum ANSI Level 1 standard for COBOL (X3.23-1974).**

**Runs under ISIS-II on Intellec or Intellec® Series II Microcomputer Development Systems.**

**Compiler compiles COBOL source programs into an intermediate code which is optimized for speed and memory space.**

**Includes execution run-time interpreter and an interactive debugger.**

**Powerful extensions for interactive programming.**

**Can Link/Call routines written in PL/M-80, FORTRAN-80 and 8080/8085 Assembly Language.**

**FORMS utility program allows the user to design and test CRT screen format input by generating COBOL source code for the data descriptions defining that CRT screen format.**

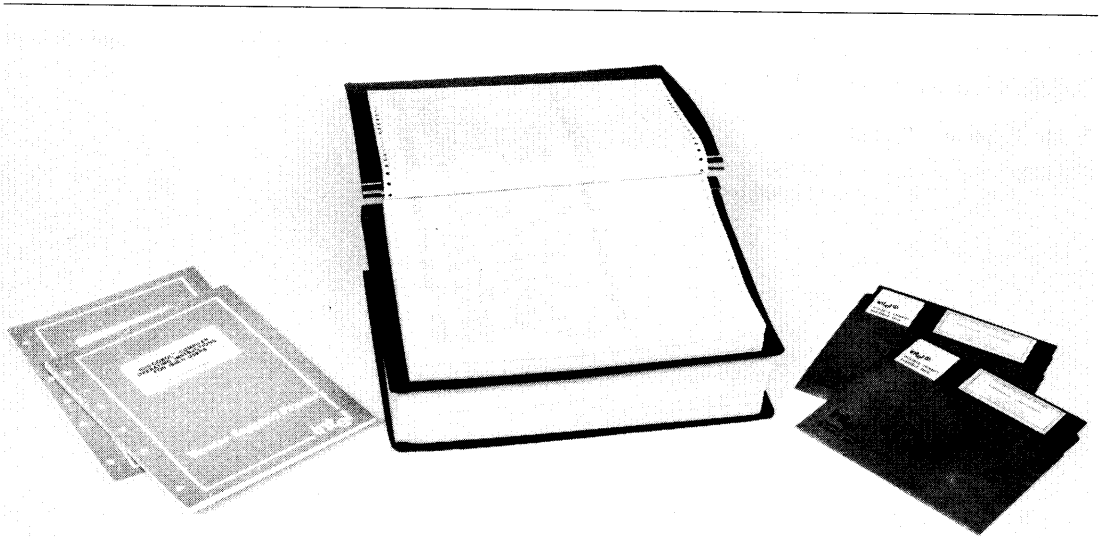
**Compile-time option available to flag any non-ANSI standard features for portability.**

**Tested using U.S. Dept. of Navy COBOL validation system.**

iCIS-COBOL, an acronym for Intel's Compact Interactive Standard COBOL, is a package designed to provide a powerful interactive business language to users of Intel's Intellec and Intellec Series II Microcomputer Development Systems. iCIS-COBOL contains the most relevant parts of the ANSI 74 standard plus extra extensions to make this product especially useful to Intellec users. The compiler provides a feature to optionally disallow the iCIS-COBOL extensions and rigidly enforce the ANSI 74 specification. This will prove beneficial to users who may need to port COBOL programs from the Intellec system to any other ANS Level 1 COBOL compiler.

iCIS-COBOL Compiler generates object code for a COBOL "virtual machine." This code is designed for optimum representation of COBOL verbs and data types. The code generated is interpreted by a Run-Time System. This consists of an interpreter which emulates the COBOL virtual machine and interfaces to the ISIS-II operating system and the CRT.

After an application program has been tested and is ready for production use, it is possible to link it permanently to the Run-Time System to form a free-standing ISIS-II loadable program.



## LANGUAGE FEATURES

COBOL consists of twelve different modules implemented either to Level 1 or Level 2 as defined in the ANSI specification X3.23. iCIS-COBOL includes the following modules implemented to Level 1:

Nucleus
Table Handling
Sequential I/O
Relative I/O
Indexed I/O
Library
Interprogram Communication

Extensions to ANSI Specification:

- *Advanced screen formatting and data entry facilities.* These include protected and unprotected data, cursor manipulation, and numeric vet.
- *Run time input of filenames.* The actual value of the external filename may be moved to a file identifier location prior to OPENing the file, avoiding the need for an external linking mechanism.
- *Line sequential files.* Variable length records separated by carriage return/line feed saves space on disk and allows iCIS-COBOL programs to process files output by a text editor.
- *Hexadecimal literals.* These may be used to define control characters to output to special peripheral devices.
- *Rapid development facilities.* During development, compiled programs may be loaded directly by the Run-Time System "fast load" facility, thus avoiding the time otherwise spent in linking.
- *Interactive debugging.* Interactive debugging permits the setting of breakpoints, examination and modification of store, etc., at run time. Each COBOL statement is identified by a four-digit hexadecimal number.
- *Lower case.* This is permitted in COBOL words and comments, thus helping to produce easy to read documentation in the program.

## INTERACTIVE CRT HANDLING

Intel has taken COBOL — traditionally a batch processing language — and extended it to become interactive. iCIS-COBOL offers many facilities for automatically formatting a CRT screen and facilitating input keying.

The user can format the screen of any system console (CRT) into protected and unprotected fields by using standard COBOL statements. The screen layout may be defined in the DATA DIVISION. An ACCEPT statement nominates a record description which permits input to the character positions corresponding to variables identified by data-names. These may be separated by FILLERs to position them on the screen. Conversely, a DISPLAY outputs only from non-FILLER fields in the record description which it nominates. The programmer can easily build up complex conversations for data entry and transaction processing.

When data is being keyed in, the operator has full cursor manipulation facilities, each variable acting as a tab stop. Non-numeric digits may not be keyed into fields defined as PIC 9. Finally, when the operator has checked that the data is correct, the RETURN key is pressed and processing continues.

| SCREEN LAYOUT AND FORMAT FACILITIES | CURSOR CONTROL FACILITIES |
|---|---|
| Screen as a record description | HOME to the start of the first data field on the screen |
| FILLER | → Forward space |
| REDEFINES | ← Backward space |
| AT line:column | ↓ Forward field |
| Character highlighting | ↑ Backward field |
| Clear screen | C/R Release the screen of data |
| Numeric vet for PIC fields | L/F Left Fill numeric field |
| | (The actual keys used vary according to CRT keyboard) |

## FORMS UTILITY

A majority (up to 80%) of debugging time can be spent in designing, coding and testing the screen form input/output of a COBOL program. The FORMS utility included in the iCIS-COBOL package significantly reduces this debugging time.

Using the FORMS program, the user may:

- Store an image copy on disk of the form he has defined for subsequent use.
- Generate iCIS-COBOL source code for the data descriptions required to define the form just created. This may then be included in an iCIS-COBOL program using COPY.
- Choose to generate a checkout program which allows duplication of the many machine conversations which would take place during a run of the application which is being designed.

## COMPILE TIME DIRECTIVES

- ANS

  If specified, the Compiler will accept only those iCIS-COBOL language statements that conform to the ANS 74 standard.

- RESEQ

  If specified, the Compiler generates COBOL sequence numbers, renumbering each line in increments of 10.

- NOINT

  No intermediate code file is output. The Compiler is, in effect, used for syntax checking only.

- NOLIST

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

- COPYLIST

  The contents of the file(s) nominated in COPY statements are listed.

- NOFORM

  No form feed or page headings are to be output by the Compiler in the list file.

- ERRLIST

  The listing is limited to those COBOL lines containing syntax errors together with the associated error message(s).

- INT (external-file-name)

  Specifies the file to which the intermediate code is to be directed.

- LIST (external-file-name)

  Specifies the file to which the listing is to be directed.

- FORM (integer)

  Specifies the number of COBOL lines per page of listing.

- NOECHO

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

## BENEFITS

- Brings COBOL to Intellec Microcomputer Development Systems.
  - — COBOL is the industry standard high-level language for business-oriented applications.
  - — Most COBOL programs are self-documenting.
  - — Conversational verbs and phrases and common business terminology make COBOL easy to learn, use and maintain.
  - — More business and application programs are written in COBOL than any other language.
- Meets and exceeds ANS Level 1 COBOL standard.
  - — Assures portability to and from all computers supporting ANS Level 1 COBOL.
  - — Extensive testing and validation using U.S. NAVY COBOL VALIDATION SYSTEM assures functionality for all Level 1 features.
- iCIS-COBOL software package provides an easy to use, efficient and friendly environment for COBOL program development.
  - — CONFIGURATOR program allows the user to reconfigure the software for any non-standard, non-Intel CRT.
  - — Interactive debugger provides features aimed at a CRT based system (rather than batch-oriented).
  - — FORMS utility program reduces total program development time by 30%.
  - — All iCIS-COBOL utilities make use of CRT cursor control.
- Adds value to an Intellec development system.
  - — COBOL applications programs developed using iCIS-COBOL software package will increase utilization of Intellec development systems.

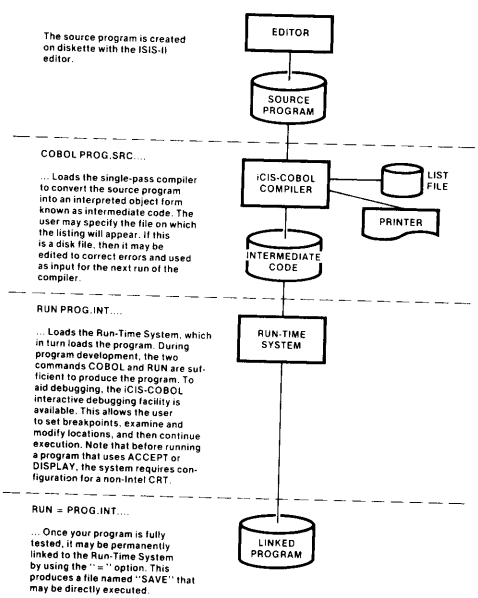



**Figure 7-12. Program Development Cycle**

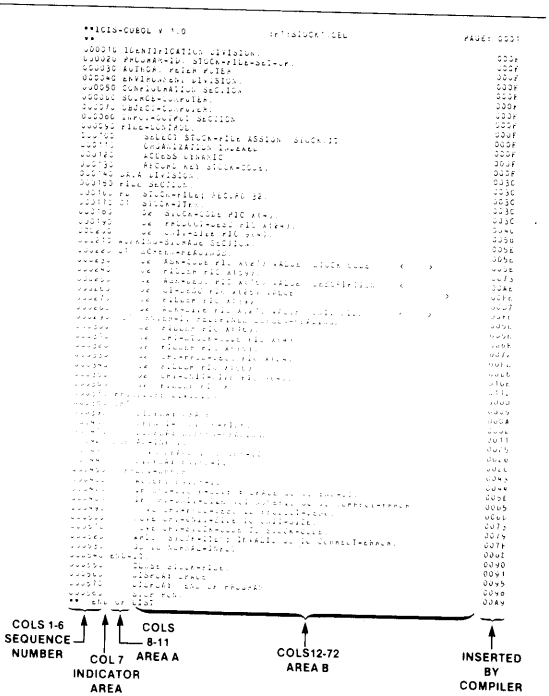



**Figure 7-13. Sample Program Listing Showing Source Format**

## SPECIFICATIONS

### Operating Environment

### Required Hardware:

Intellec Microcomputer Development System
— Model 800
— Series II Model 220, Model 230

48KB of Memory

Dual Diskette Drives
— Single or Double Density

System Console
— Intel or non-Intel CRT

**Recommended Hardware:**

64KB of Memory

Double Density Dual Diskette Drives

**Optional Hardware:**

Line Printer

**Required Software:**

ISIS-II Diskette Operating System
— Single or Double Density

**Optional Software:**

ISIS-II CREDIT (CRT-Based Text Editor)

### Documentation Package

iCIS-COBOL Language Reference Manual (9800927-01)

iCIS-COBOL Compiler Operating Instructions for ISIS-II Users (9800928-01)

iCIS-COBOL Pocket Reference (9800929-01)

### Shipping Media

Flexible Diskettes
— Single and Double Density

## ORDERING INFORMATION:

| Product Code | Description |
|---|---|
| MDS-380 | iCIS-COBOL Software Package |