1983 DEVELOPMENT SYSTEMS PRODUCTS DATABOOK National Semiconductor

DEVELOPMENT SYSTEMS PRODUCTS DATABOOK

NATIONAL
SEMICONDUCTOR
CORPORATION

National
Semiconductor
Corporation

Introduction

Welcome and thank you for your interest in National's products. National has been providing and manufacturing microprocessors and support tools since 1972. This databook describes the various development tools available to design and develop microprocessor based products. Support tools include software packages for applications software development; real-time emulators for fast and efficient hardware and software debugging; peripherals packages (such as PROM programmers and printers); and, of course, powerful host systems to bring all of these support packages together.

National offers the STARPLEX II and the SYS16 Development Systems. STARPLEX II allows the designer to develop 8-bit microprocessor-based products, while the SYS16 allows the designer to develop NS16000-based products. Available for each of the host systems are appropriate packages for software development work. Software packages for the STARPLEX II include 8-bit cross-assemblers and compilers (such as the COPS cross-assembler for COP400 family microcontrollers), and PASCAL compilers for 8085 and NSC800 microprocessors. Software for the SYS16 include PASCAL and C to support the NS16000 family. To debug the software in the hardware prototype, powerful real-time In-System Emulators (ISEs) are available for the appropriate microprocessors.

The Service Organization provides technical support and repair for Microcomputer Systems Division products. The designer can use toll-free numbers (800-536-1866 or 800-672-1811 in California) to contact the Response Center for hardware and software technical assistance or for service.

As integrated circuits become more and more complex, the benefit of consistently high quality products becomes increasingly more important to customers, many of whom have long recognized National as *the* outstanding supplier of top quality products. Such recognition is the result of a management-driven Quality Improvement Program that has pervaded every manufacturing operation, from product design through assembly and packaging, for the past several years. Progress has been nothing less than dramatic, and National's commitment to quality will remain unrelenting in the decades to come.

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

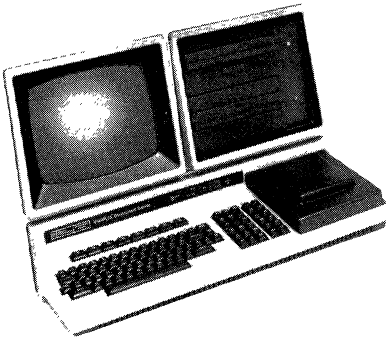
1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National
Semiconductor
Corporation

Table of Contents

National Semiconductor

STARPLEX II™ Development System



■ A Complete Development System

- Dual CPU microprocessor-based system in master/slave configuration
  - 128K bytes of Random Access Memory
  - Dual floppy disk drives
  - Video monitor and keyboard controller
  - Two RS232C interfaces
  - Integral CRT keyboard with eight upper/lower case for a total of sixteen user definable keys
  - PROM programmer interface
- Software
  - Disk Operating System
  - Resident Debugger
  - Text Editor
  - Macro Assembler
  - On-board ROM and RAM diagnostics
  - I/O Spooling
  - FORTRAN
  - BASIC
- Options
  - In-System Emulator (ISE™) packages for NSC800™ INS8048 family, INS8070 family, NS80CX48, 8080, 8085 and Z80 microprocessor devices
  - In-System Emulator package for COP400 microcontroller devices
  - PL/M for 8080/8085, PL/M for NSC800/Z80
  - PASCAL compiler for 8080/8085, PASCAL compiler for NSC800/Z80
  - Optional double-sided/double-density disk drives with 2 megabytes of memory expandable to 4 megabytes
  - Cross assemblers (Included with the emulator packages)
  - STARLINK — Interface to Intellec Development System
  - PAL /PROM programmer personality modules

■ Field-Upgradable from STARPLEX™ 80/41, 80/51 or 80/61 Systems

- Upgrade kit includes:
  - Z80A Master CPU Board
  - Z80A Slave CPU Board with 64K bytes of RAM
  - Internal RS232C cable and connector
  - Keyboard with user-definable keys
  - Disk-Based Operating System for STARPLEX II

■ Easy to Use

- Prompting menus guide operator entries
  - English language explanation of user errors
  - Direct system function keys to PAUSE/CONTINUE/ABORT/DEBUG
  - HELP key for online user assistance
  - Single stroke CRT edit keys

Product Overview

The STARPLEX II Development System is a general-purpose microcomputer and microprocessor development system. New levels of operating simplicity have been designed into the STARPLEX II system to significantly reduce the amount of time spent on product development. By getting the user into actual application work sooner and with fewer mistakes, the STARPLEX II system allows the user to take full advantage of time spent at the console.

A Complete System

The STARPLEX II design combines all the components required for the entire development task in one complete system. The STARPLEX II package includes a Z80A-based system controller board, a Z80A-based user processor/memory board with 64K bytes of RAM, 64K bytes of system RAM, 1M byte of disk storage controlled by a floppy disk controller, a video monitor and keyboard. The standard STARPLEX II software package includes a disk operating system, Z80 assembler, debugger, editor, linker, loader, FORTRAN, BASIC, on-board ROM diagnostics and utilities. Options available are: in-system emulation packages for real-time debugging of customized hardware and software prototype systems, PAL/PROM programmer personality modules for verifying, copying and programming PROMs or PALs, STARLINK for transferring files between STARPLEX II and Intellec Development System, and cross assemblers.

Easy to Use

The STARPLEX Systems reduce the time a user must spend at a terminal by making many complex functions accessible through single easy keystrokes. System commands are initiated by clearly marked function keys which invoke prompting menus to guide the user through each task. These function keys eliminate the need to memorize system commands and various command options. As a result, there is no need to refer to lengthy documentation, and errors or delays caused by incorrectly entered commands are eliminated. With the user-definable keys on the STARPLEX II System keyboard, the amount of time a user must spend at a terminal is further reduced. Eight function keys are provided with upper and lower case capability for a total of sixteen different keys which are user-definable. These keys may be utilized both in command mode (system) and by an application program running on the system. Thus, while system commands are initiated by clearly marked function keys, which invoke prompting menus to guide the user through each task, many non-system complex functions become accessible through these user-definable keys.

Recognizing that a great deal of the user's time is spent on creating and changing source code, the designers of the STARPLEX II system have devoted special attention to the text editing facility.

A set of special function keys direct the STARPLEX II Editor, allowing corrections to be made with single keystrokes. Also, the powerful "string mode" commands allow search and replacement of character strings as well as block moves. An entire file may be quickly and easily reviewed or altered. The number of mistakes is reduced because the data and changes are immediately displayed. Backup files are automatically created, protecting the user from accidental loss of data. Because the STARPLEX II system is easy to use, learning time is considerably shortened. A first-time user can be productive within a half hour. Also, as users make more efficient use of the system, machine availability is maximized.

Spooled Printer Capability

STARPLEX II supports spooled I/O to a user-selected print or another input or output device. Thus, printing long listings of files, compiler output and similar tasks may now be done at the same time as text editing, compiling, emulation, debugging, etc. The net result is a greater utilization of designer resources and subsequent reduction in program development time.

Resident System Debugger

The system debug utility is resident and always available to the user. This program does not occupy any user space in memory and can be invoked by a single keystroke. Unlike many other debug utilities, the STARPLEX II debugger does not have to be specified prior to program execution and may be invoked at any time.

Full Product Line Support

The STARPLEX II system supports development for the NSC800, NS16000, INS8070 family (8070, 8072, 8073 with Tiny Basic Interpreter), INS8048 family (8048, 8049, 8050), NS80CX48, Z80A, Z80B, 8085 microprocessors and COP400 microcontroller devices.

Functional Description

Hardware Modules

STARPLEX II components are packaged into modules which form a unified system when placed together. The modules are durable, with housings constructed of ⅛-inch aluminum and front panels of molded lexan foam.

STARPLEX II is designed for easy maintenance. Snap-down doors on the base module make it easy to access the card cages and circuit boards. Interconnecting cables between all modules and boards are routed to the rear of the system and covered by easily removable cable channels. Thus, cables are out of sight and protected from accidental damage. All cables, including the single AC power distribution system, are plug-detachable at both ends, making it easy to disconnect modules and reconfigure the system as the user chooses.

STARPLEX II Electronics

Five printed circuit boards make up the STARPLEX II electronics: the main Z80A-based CPU board, a Z80A-based user processor board which also has 64K bytes of memory, an 8080A-based video monitor/keyboard controller board, an 8080A-based floppy disk controller board and an additional 64K byte memory board.

The Z80A-based CPU board and user slave processor board are designed in a master/slave configuration to give the user processing power and speed that were unobtainable with previous development systems. The main CPU board with the floppy disk controller board and the video/keyboard controller board all have multi-master bus logic allowing them to share the system bus. The floppy disk controller board and the video/keyboard controller board communicate with the main CPU board and user processor board using Direct Memory Access and programmed I/O.

The optional printers and PAL /PROM programmer personality modules communicate with the main CPU/user processor boards through two programmable parallel I/O ports. A pair of RS232C ports on the main CPU board are available and permit both asynchronous and synchronous communications for use with options such as STARLINK.

Individual circuit boards are built to National's high manufacturing quality standards, utilizing techniques such as computer-aided làyout and auto insertion. All boards are tested dynamically under system load conditions at elevated temperatures as part of a thorough factory burn-in.

Software

User programs are separated from those of the STARPLEX II operating system. This means that users have much more memory space available, and since the operating system resides in its own environment, accidental interface between user programs and the operating system is virtually eliminated.

The STARPLEX II software is completely thought out from a functional standpoint, carefully engineered to be easy to understand and use, and thoroughly integrated into the total system. Every aspect is designed to assist the user in rapidly developing microprocessor-based systems from the ground up.

The elegance of STARPLEX II software lies in its ability to make the complicated process of program development appear simple to the user.

OPERATING SYSTEM

The operating system provides system housekeeping functions and coordinates access to system resources. It includes a nucleus file manager, an I/O control system and a loader.

The nucleus of the STARPLEX II operating system controls and allocates system resources for the higher-level processes. The nucleus:

- Provides synchronization and communication facilities for higher-level asychronous processes.
- Services all hardware interrupts.
- Provides interval timer functions.
- Is completely device-independent.

File Manager

The file manager organizes, stores and retrieves data and programs stored on the diskettes.

- Maintains a directory.
- Allows multiple file attributes.
- Supports random access.

I/O Control System

The I/O control system is designed to eliminate the need for the user to understand the physical I/O characteristic of each individual device and presents a simplified, logical device-independent architecture.

- Provides overlapped I/O commands.
- Allows files to be accessed by name.
- Handles error conditions.
- Supports spooled I/O to a user-selected print or another input or output device.

Loader

The loader brings programs into main memory at specified locations.

- Provides "load and go" mode.
- Allows controlled load mode — starting address returned to calling program, useful for implementing overlay structures.

DEVELOPMENT SERVICES

The "development services" include a linker, a CRT-oriented editor, utilities, a resident debugger, optional PAL/PROM programmer support macro assemblers, BASIC and FORTRAN IV, optional PL/M for NSC800/Z80 or 8080/8085, and optional PASCAL for NSC800/Z80 or 8080/8085.

Linker

The linker combines relocatable object modules created by the assemblers or compilers into an executable run time module.

- Assigns absolute addresses to load modules.
- Produces a memory map of linked components.
- Searches system and user libraries for unresolved external references.

Editor

The STARPLEX II editor is an easy-to-use CRT-oriented text editor.

- String search and replace.
- Forward and backward paging.
- Block moves.
- Automatic source file backup.
- Traps illegal commands.

Utilities

General utilities provide routine maintenance functions.

- Transfer data files between devices.
- Obtain diskette directory listings.
- Format diskettes.

- Modify file attributes.
- Rename files.
- Print screen.

Debugger

The system debug utility is resident and always available to the user. The debugger does not occupy any user space in memory and may be invoked by a single keystroke. The program debugger simplifies program checkout by allowing program execution to be monitored and altered.

- Allows single step control.
- Permits eight breakpoint assignments.
- Displays program counter and registers at breakpoints.
- Memory references are absolute or relative to one of the relocation registers.

PAL/PROM Programmer Support

The PAL/PROM programmer support software manages the optional PAL/PROM personality module functions.

- Allows PROM code to be listed, verified and copied.
- Data stored in a PROM can be transferred to or from another PROM, a diskette file, memory, the video monitor or keyboard.
- Allows for custom programming of programmable array logic devices (PAL).

Macro Assembler

Individual macro assemblers can assemble 8080, 8085, 8048, 8070, NSC800, or Z80 mnemonic code and allow operator definition of useful higher-level instructions called "Macros" which are then expanded into a sequence of machine-level instructions. (Macro assembler for NSC800/Z80 is included with the STARPLEX II system. All other cross assemblers are optional.)

- Generates absolute or relocatable object modules.
- Conditional assembly parameters.
- Allows external references.

FORTRAN IV

The FORTRAN IV compiler on the STARPLEX II system meets the ANSI X3.9-1966 standard and includes the following enhancements:

- PEEK and POKE — allow direct access to memory.
- Supports user-written I/O drivers.
- Random access disk I/O.
- Allows assembly language subroutine calls.

BASIC

The STARPLEX II BASIC compiler/interpreter conforms to the Dartmouth-defined BASIC with extensions:

- PEEK and POKE — allow direct access to memory.
- Complete string operators.
- Multi-dimensional arrays.
- Extensive debugging and programming aids — trace, edit, direct mode, renumber.

PL/M for 8080/8085 and NSC800/Z80 (Optional)

PL/M is compatible with the industry standard PL/M, but offers many enhancements to improve program execution time and memory utilization.

- Available for 8080/8085 object code or NSC800/Z80 object code.
- Hardware access via high-level statements.
- Block structure facilitates structured programming techniques.
- Relocatable and linkable output object code.

PASCAL for 8080/8085 and NSC800/Z80 (Optional)

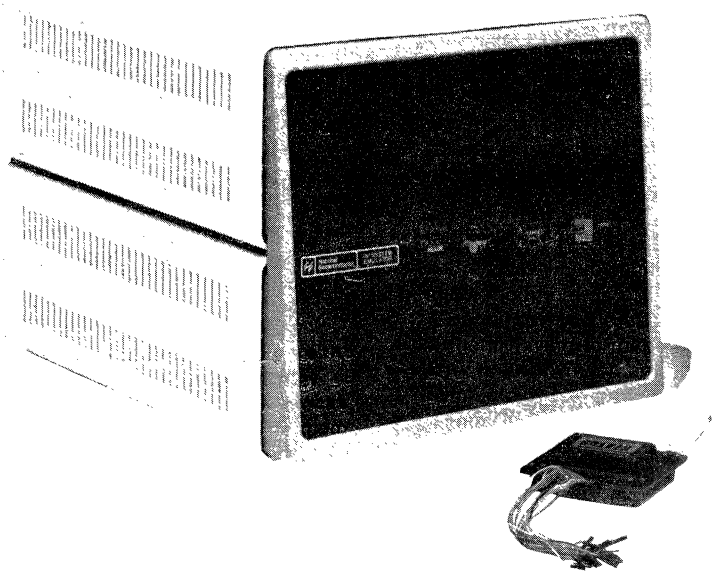
Specifications

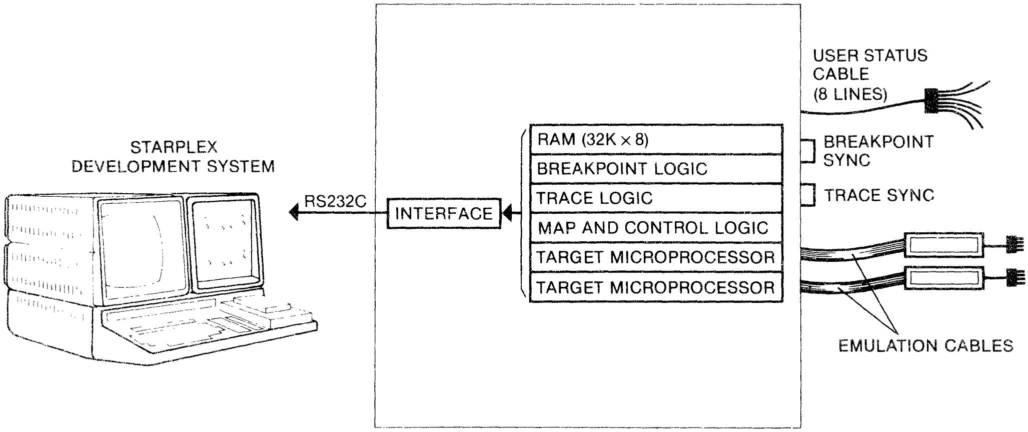
Processor Subsystem:	Z80A-based CPU board Z80A-based user processor/memory with 64K bytes RAM Video monitor/keyboard controller Double-density floppy disk controller Memory board with 64K bytes RAM (128K bytes total RAM)
Floppy Disk Subsystem:	
Configuration	Dual disk drives
Format	IBM-compatible, soft-sectored
Capacity	Double-density, single-sided 512K bytes/drives
Maximum Capacity	Expanded to 4 double-density, double-sided drives with 4 megabyte storage capacity
Keyboard Subsystem:	
System Function	8 single-stroke system control keys
ASCII	58 alphanumeric keys
Programmable	8 user-definable keys with upper/lower case
CRT Subsystem:	
Matrix	7×9 dot
Display Array	80 columns by 25 lines
Phosphor	P2 green
Other	Screen tilted 10° for comfortable viewing
Printers:	
Type	Impact
Speed	120 characters per second
Width	132 columns
Character Type	7×9 dot matrix
Power:	115 VAC, 60 Hz, 10 amps (max) or 230 VAC, 50 Hz, 5 amps (max)
Base Module	644 Watts
Floppy Disk Module	966 Watts
Impact Printers	360 Watts
Video Monitor	34 Watts

Physical:

	Base Module	Floppy Disk Module	Impact Printer	Video Monitor
Height	5.75 in. 14.6 cm	11.5 in. 29.2 cm	8 in. 20.3 cm	11 5 in. 2 92 cm
Width	26 in. 66 cm	13 in. 33 cm	24.5 in. 62.2 cm	13 in. 33 cm
Depth	26 in. 66 cm	19 in. 48.3 cm	18 in. 45.7 cm	19 in. 48.3 cm
Weight	68 lb. 30.8 kg	50 lb. 22.7 kg	60 lb. 27 kg	29 lb. 13.2 kg

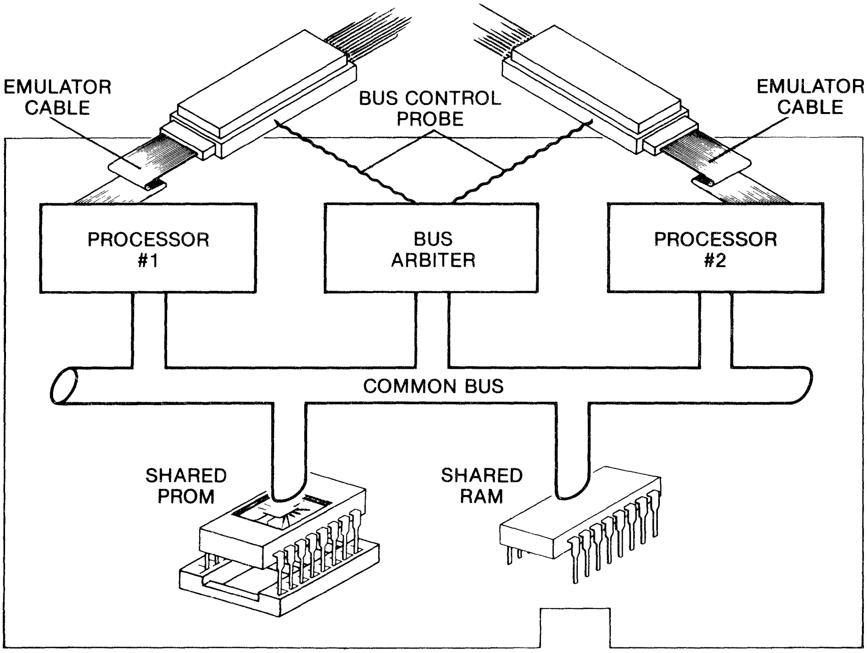
In-System Emulator Module



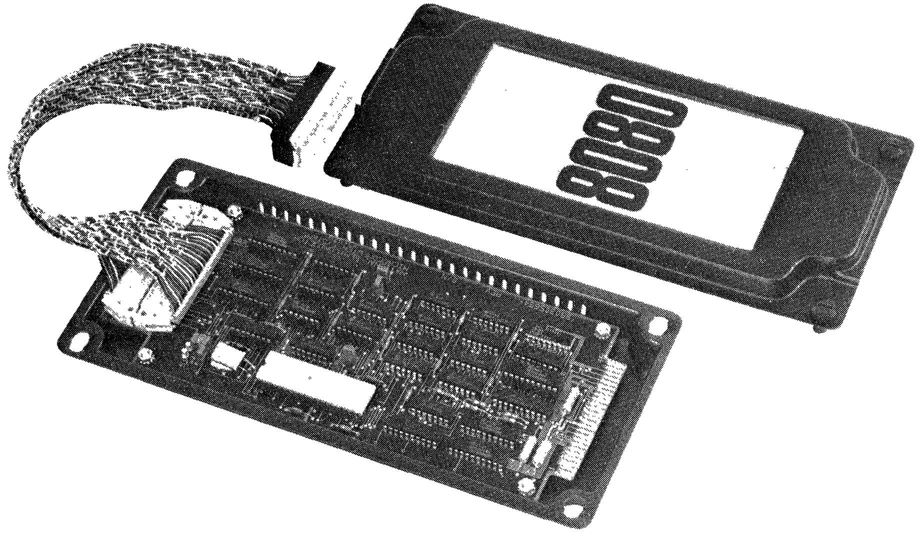


In-System Emulator System Configuration

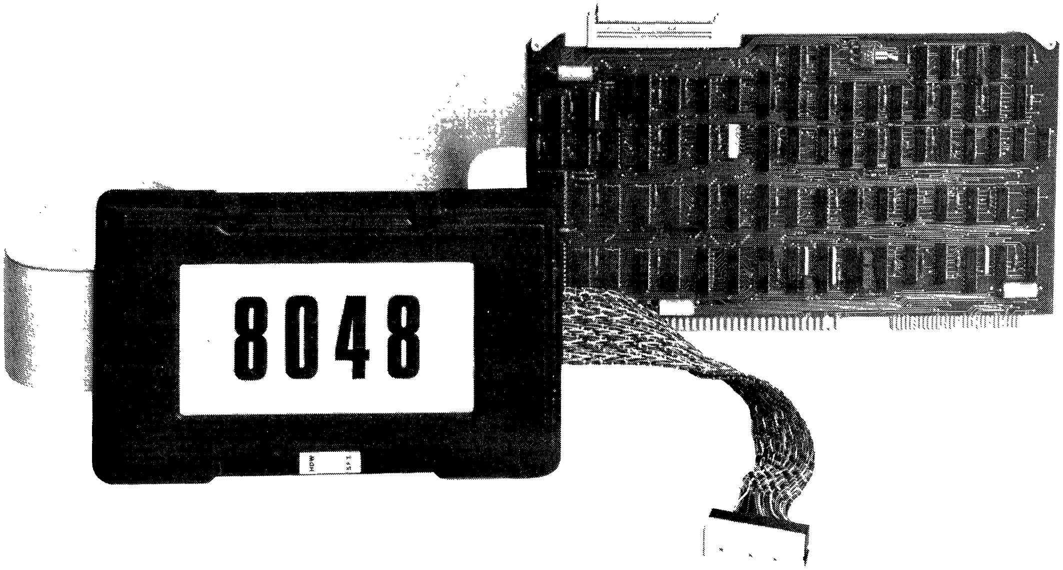
Application Multiprocessor System Configuration



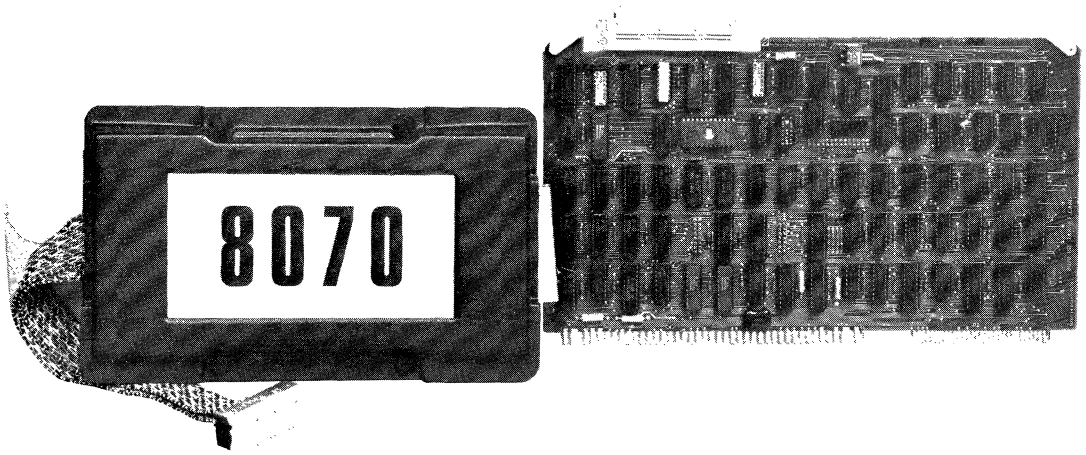
8080 Emulator Package



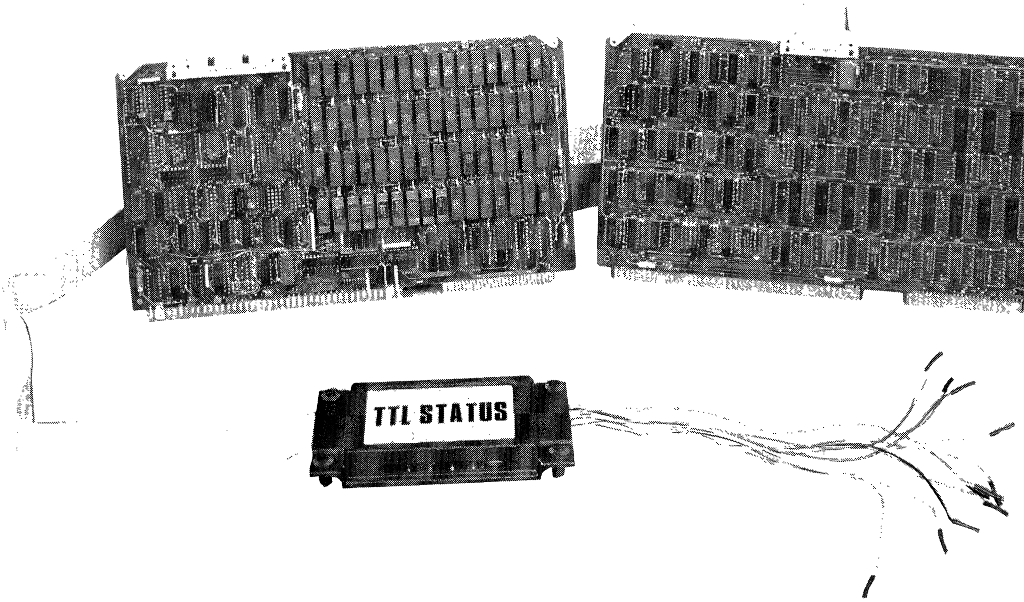
8048 Family Emulator Package



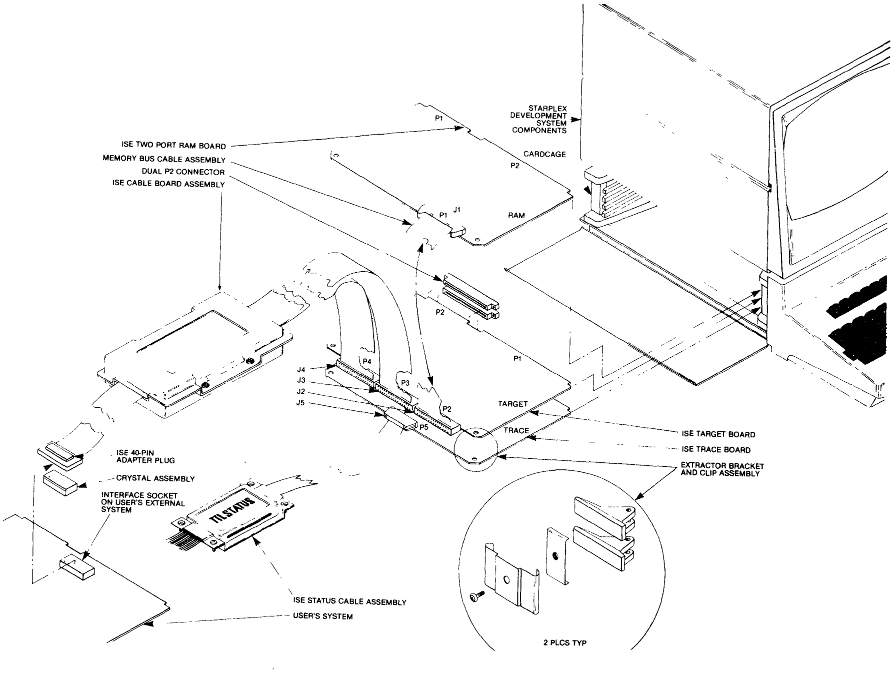
8070 Series Emulator Package



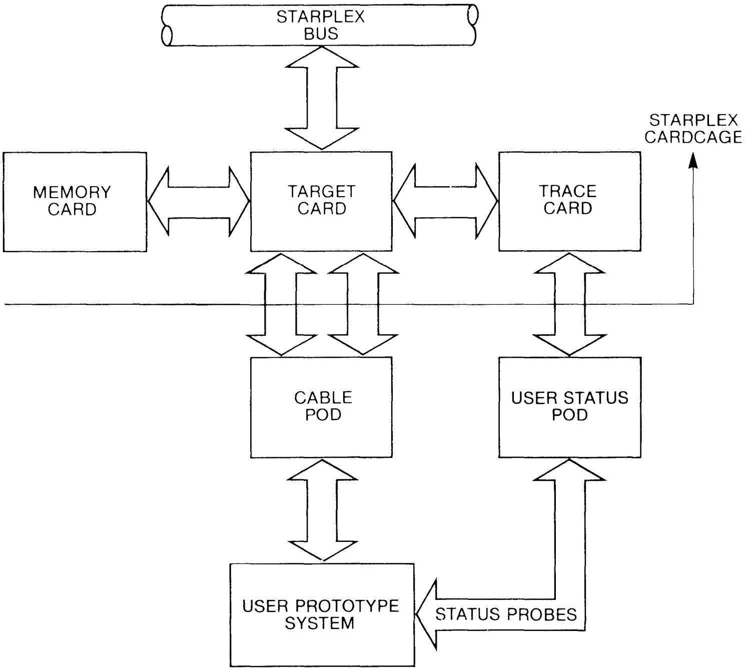
Integral In-System Emulator



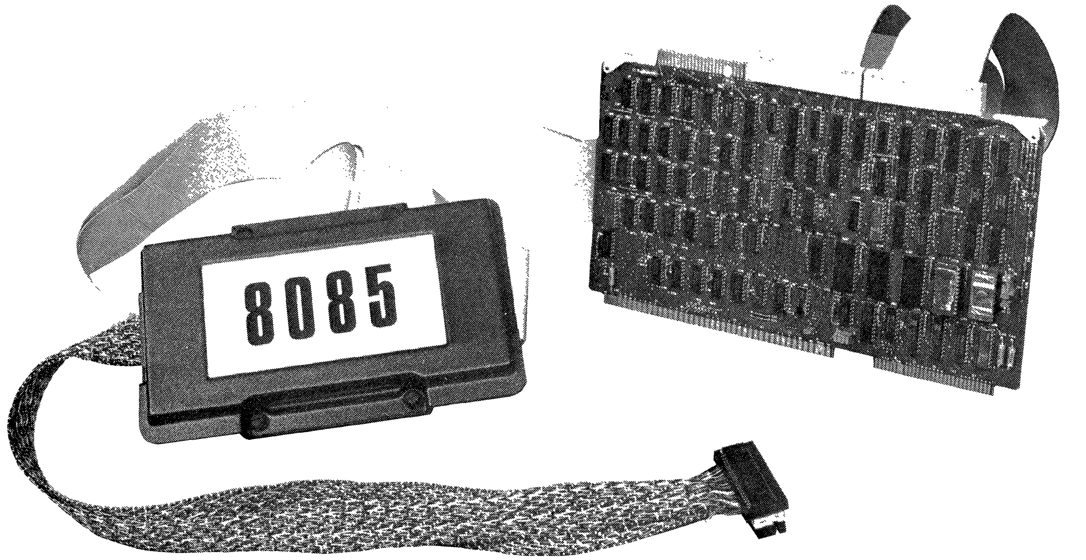
Integral ISE Components Installation



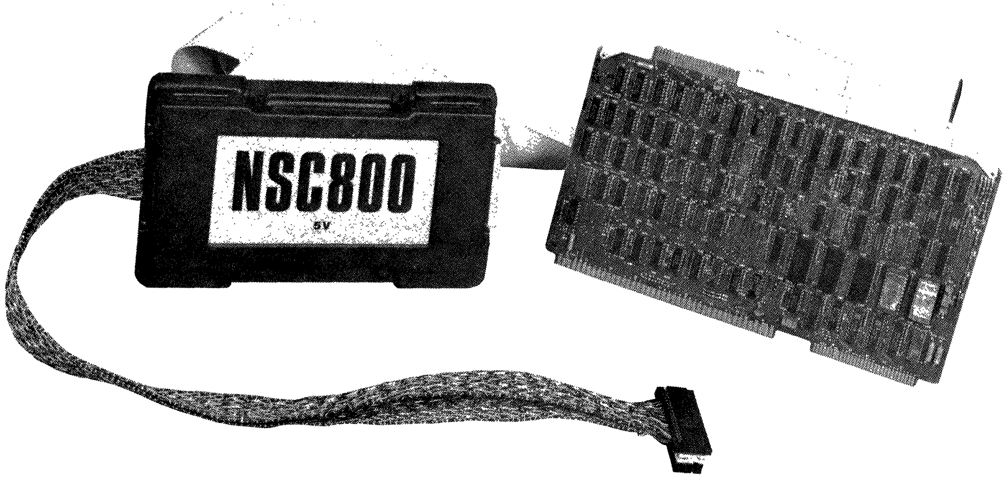
Integral ISE System Configuration (Total: 3 Boards and 2 Pods)

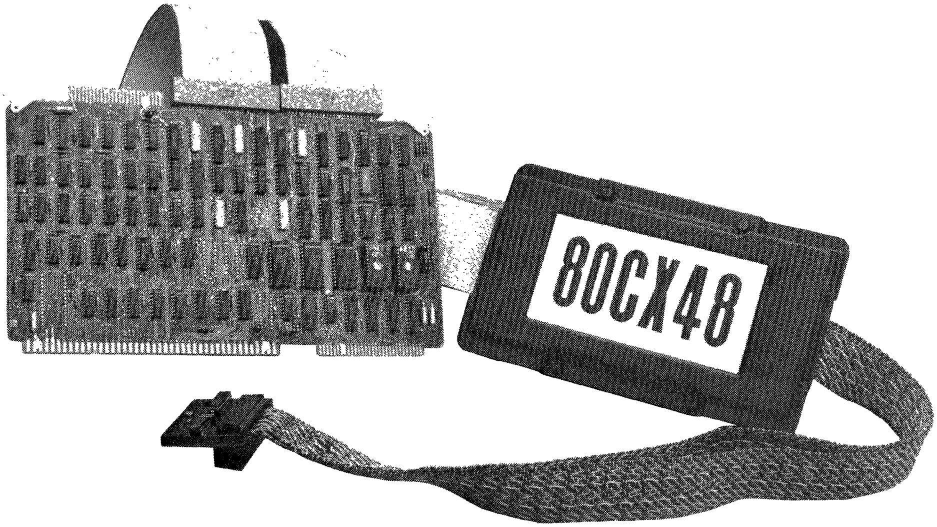


8085 Emulator Package

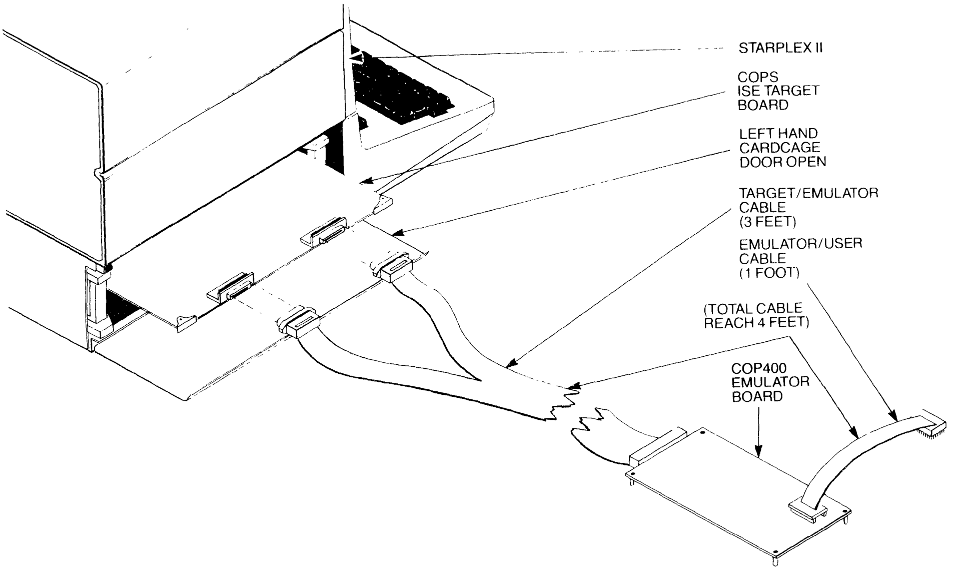


NSC800 (5V) Emulator Package

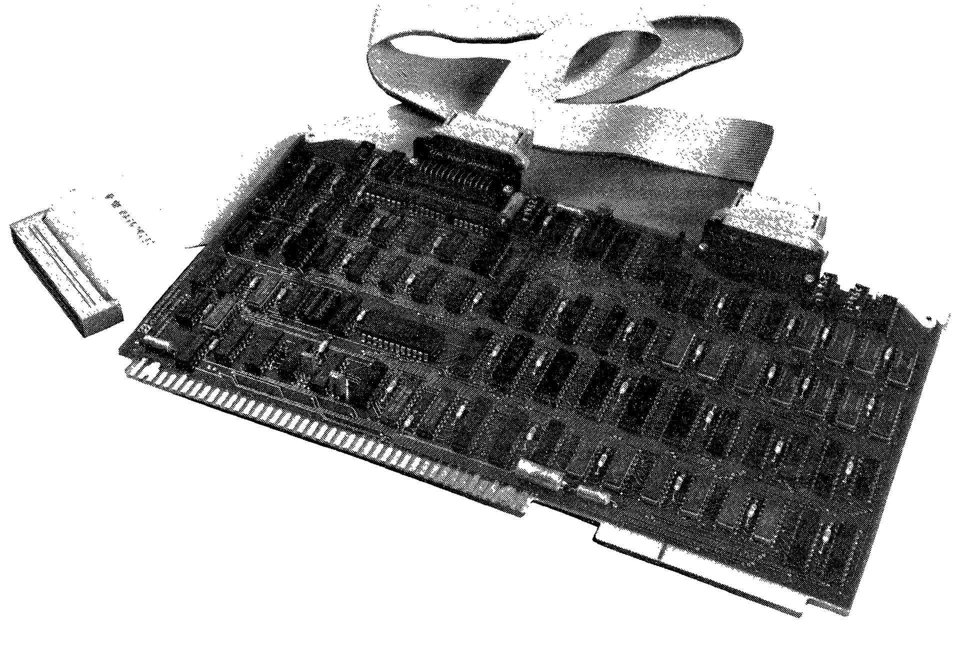




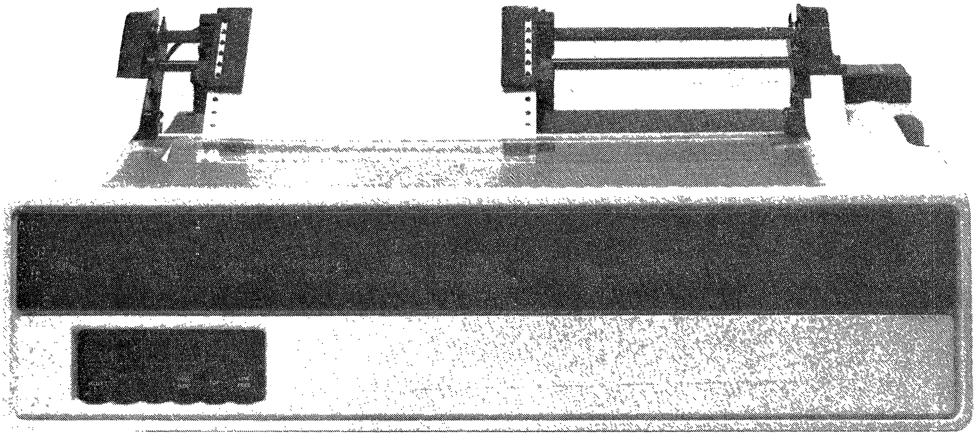
Installation of the COPS ISE Target Board and an Emulator Board


STARPLEX II
COPS ISE TARGET BOARD
LEFT HAND CARDCAGE DOOR OPEN
TARGET/EMULATOR CABLE (3 FEET)
EMULATOR/USER CABLE (1 FOOT)
(TOTAL CABLE REACH 4 FEET)
COP400 EMULATOR BOARD

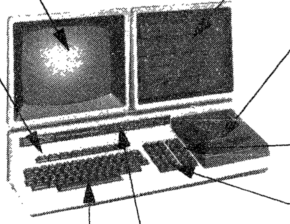
COPS™ In-System Emulator Package



Impact Printer



STARPLEX II Development System



Video Monitor Subsystem
Large screen — measures 12" diagonally
Legible characters — 7×9 dot matrix
24 lines × 80 characters
Soft green phosphor
Variable screen intensity
10° tilted screen for comfortable viewing
Extensive screen control; scrolling, blink, blank, inverse video or alternate characters

User Definable Function Keys
Eight function keys are provided with upper and lower case capability for a total of sixteen different keys which are user definable

Processors Subsystem
Z80-based CPU
Z80-based user processor/memory with 64K byte RAM
Floppy disk controller/formatter
64K byte RAM
Dual 4-slot chassis provides three expansion slots

Disk Subsystem
Dual standard floppy drives give 512K bytes per drive capacity
Uses IBM soft-sectored format
Expanadable to four drives (two million bytes)

PROM Programmer (Optional)
Plug in PROM personality modules — standard PRO-LOG compatible
Programs bipolar PROMs, 2708, 2716 EPROMs, PALs

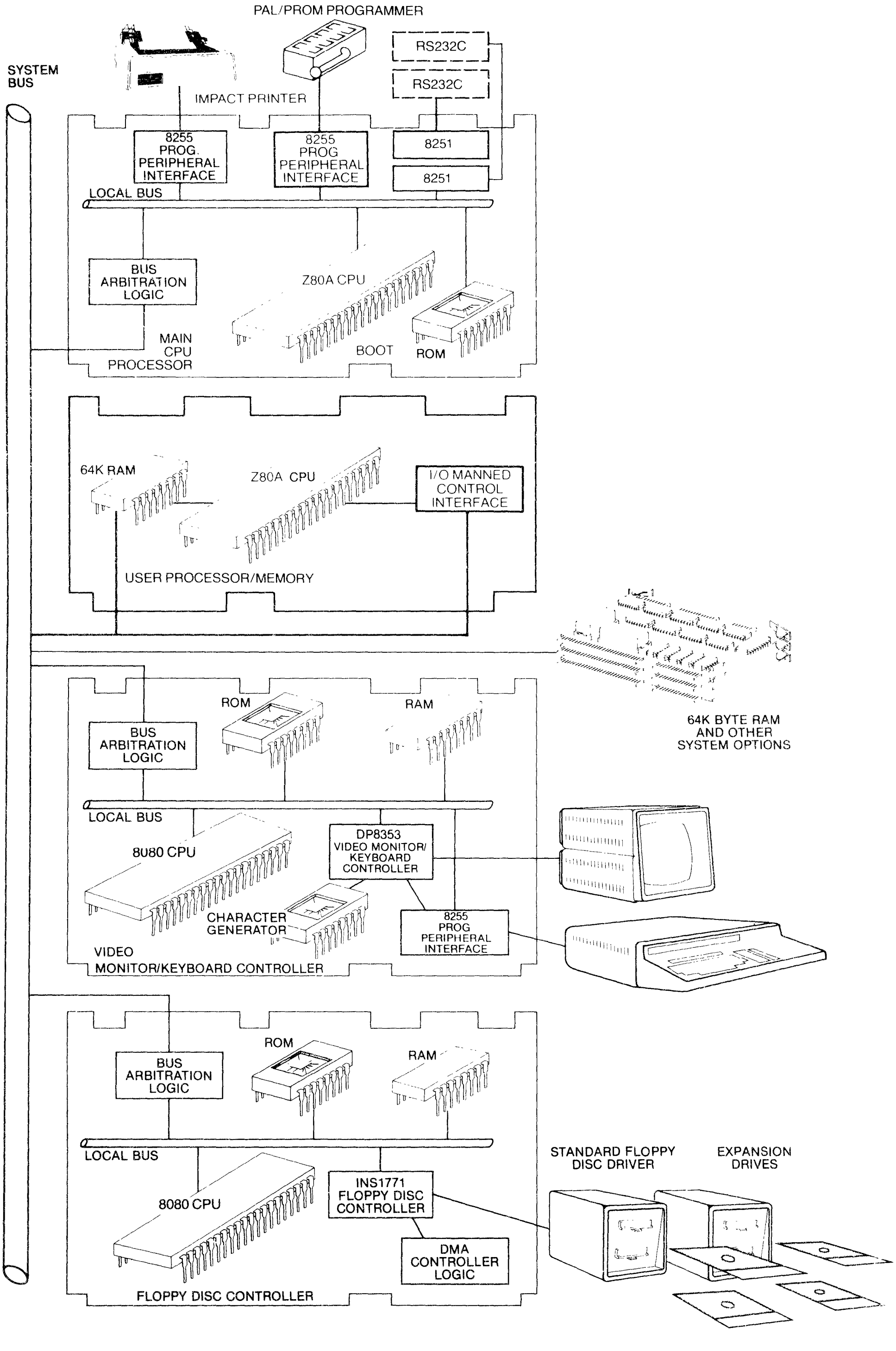
System Function Keypad
9 system control keys
Control program execution

Editor Keypad
5 cursor keys
13 special edit keys

ASCII Keypad
58 alphanumeric keys

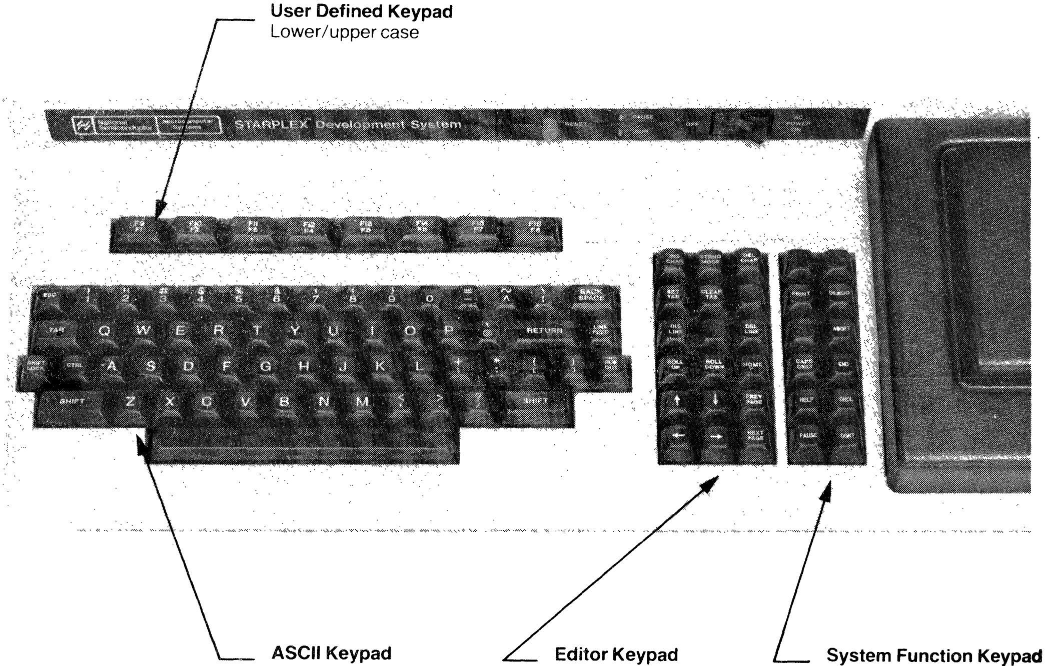
System Reset Boot Load Button
Powerful resident bootstrap has built-in micro-diagnostics to check all system facilities on initialization, then automatically switch out of user memory space

STARPLEX II Multiprocessor System



STARPLEX II Keyboard

User Defined Keypad
Lower/upper case



ASCII Keypad
58 alphanumeric keys

Editor Keypad
5 cursor keys
13 special edit keys

System Function Keypad
9 system control keys
Control program execution

Standard Configuration

IN THE STANDARD CONFIGURATION, STARPLEX II provides a fully functioning turnkey system including the following features:

- CPU Master
- CPU Slave
- Bootstrap and diagnostic utility
- Two RS232C serial I/O ports
- Real time clock/calendar
- 128K bytes of mappable RAM
- Keyboard base
- Video monitor with 7 × 9 dot matrix and 1920 character display
- Dual floppy disk subsystem with double-density (1 mb) or double-sided double-density (2 mb) disk drives
- Debugger for diagnosing program execution
- Additional utilities for system maintenance
- Expansion slots for Integral ISE™ capability
- BASIC interpreter
- FORTRAN compiler
- Modular construction for versatility in operation
- Expansion capabilities to meet your growing requirements
- Complete operating system including an input/output system with an independent interface to user tasks
- File manager for comprehensive data storage and retrieval file creation, protection, deletion and attribute assignment with use of unique keyboard utility keys
- Screen oriented text editor for creating and editing source statements
- Macro assembler for assembling Z80 mnemonics and user-defined macros
- Linker for linking independent program modules into executable files
- PROM programming capability including interface board and universal software with PAL support

Order Information

SPX-90/51	STARPLEX II Development System with 1 Megabyte Disk Storage (single-sided, double-density drives) (60 Hz)
SPX-90/61	STARPLEX II Development System with 2 Megabyte Disk Storage (double-sided, double-density drives) (60 Hz)

Options

SPM-90-A02-2	PROM programming module for programming 2716 EPROMs
SPM-90-A06-1	STARPLEX II Dual Single Sided, Disk Expansion
SPM-90-A06-2	STARPLEX II Dual Double Sided, Disk Expansion
SPM-90-A08	In-System Emulator Module
SPM-90-A09-1	8080 Emulator Package
SPM-90-A09-2	8048 Emulator Package (includes upgrade kits that convert ISE 8048 to emulate 8049 and 8050)
SPM-90-A09-3	8070 Emulator Package (includes upgrade kits that convert ISE 8070 to emulate 8070 and 8073)
SPM-90-A13	Integral In-System Emulator Package
SPM-90-A13-2	80CX48 Emulator Package
SPM-90-A13-3	8085 Emulator Package
SPM-90-A13-4	NSC800 Emulator Package
SPM-90-A13-7	Z80 (4 MHz) Emulator Package
SPM-90-A15	COPS™ Emulator Package
SPM-90-A20	Z80 (6 MHz) Complete ISE Package
SPM-90-A25	80CX48 Complete ISE Package
SPM-90-A55	Impact Printer
SFW-90-A001	8048 Cross-Assembler
SFW-90-A002	8070 Cross-Assembler
SFW-90-A003	NSC800 Cross-Assembler
SFW-90-A006	COPS Cross-Assembler
SFW-90-A009	8080 Cross-Assembler
SFW-90-A50	PL/M Compiler for 8080/8085
SFW-90-A60	PL/M Compiler for NSC800/Z80
SFW-90-A200	CP/M Operating System Software Package
SFW-90-A300	PASCAL Compiler for 8080/8085
SFW-90-A320	PASCAL Compiler for NSC800/Z80
AEE-90-A001	STARLINK — SPX/MDS220, 230 Link
AEE-90-A002	STARLINK — SPX/MDS800, 888 Link

Note: To order 50 Hz add the letter "E" to the order.

Documentation

STARPLEX II Development System

420306465-001	STARPLEX II System Hardware Reference Manual
420306383-001	STARPLEX II System Software Reference Manual
420305788-001	STARPLEX II Macro Assembler Software User's Manual
420305790-001	STARPLEX II FORTRAN Compiler Software User's Manual
420305791-001	STARPLEX II BASIC Interpreter Software User's Manual
420305804-001	BLC-8222 Double-Density Floppy Disk Controller Hardware Reference Manual
420305587-001	BLC-8228/8229 Video Monitor/Keyboard Controller Hardware Reference Manual
420305529-001	BLC-032/048/064 32/48/64K RAM Board Hardware Reference Manual
420306183-001	Universal PAL/PROM Programmer User's Manual

STARPLEX II Development System Options

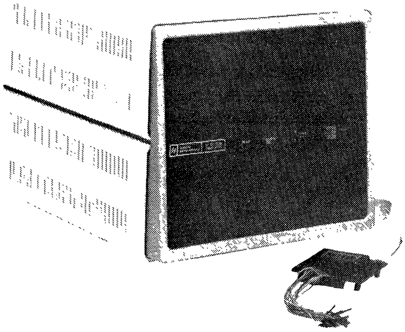
420305653-001	SPM-90-A09-1 8080 ISE Target Board User's Manual
420305869-001	SPM-90-A08 In-System Emulator Reference Manual
420306065-001	SPM-90-A09-2 8048 ISE Target Board User's Manual
420306132-001	SPM-90-A09-3 8070 ISE Target Board User's Manual
420306240-001	SPM-90-A13-3 8085 Integral ISE-User's Manual
420306241-001	SPM-90-A13-4 NSC800 Integral ISE User's Manual
420306692-001	SPM-90-A13-7, Z80 Integral ISE User's Manual
420306254-001	SPM-90-A15 COPS ISE User's Manual
420308101-001	SPM-90-A13-2, 80CX48 Integral ISE User's Manual

STARPLEX II Development System Software

420305789-001	SFW-90-A009 8085/8085 Cross Assember Software User's Manual
422306050-001	STARLINK, STARPLEX II/ MDS800/888 or MDS220/230 Reference Guide
420306064-001	SFW-90-A001 8048 Family Cross-Assembler User's Manual
420306123-001	SFW-90-A002 8070 Family Cross-Assembler User's Manual
420306198-001	SFW-90-A003 NSC800 Cross-Assembler User's Manual
420306253-001	SFW-90-A006 COPS Cross-Assembler User's Manual
420306371-001	SFW-90-A50, SFW-90-A60 PLM80 Software Reference Manual
420306680-001	SFW-90-A300, SFW-90-A320 Pascal Compiler Reference Manual

National Semiconductor

In-System Emulator (ISE™) Module



- **Real-Time Emulation of 8-Bit Microprocessors**
  - Full Support for 8080 Microprocessors
  - Full Support for 8048 Family Microprocessors (i.e., 8035, 8039, 8040, 8048, 8049, and 8050)
  - Full Support for 8070 Series Microprocessors (i.e., 8070, 8072 and 8073. 8073 is preprogrammed with National's Tiny BASIC™.)
- **Communicates to Any STARPLEX™/ STARPLEX II™ Development System Via a High-Speed RS232C Serial Port**
- **A Total Emulation System**
  - Hardware
    - — Processor Independent
    - — Breakpoint, Trace, Interface, Memory Mapping and Control Logic
    - — 32K-Bytes Dedicated Mapped Memory
    - — RS232C Serial Port
    - — 128 x 40-Bit Trace Memory
    - — Trace Load Sync Pulses
    - — 8-Bit User Status Cable
    - — Microsecond Timer
  - Software
    - — Host Resident Symbolic Debugger and Driver Software
    - — Control Firmware
    - — In-Line Assembler and Disassembler
    - — Coast After Breakpoint
- **Easy to Use**
  - Consistent and Easy-to-Use Commands
  - Infile Facility for Automatic Test
  - Full Access to STARPLEX/STARPLEX II Development Systems Facilities
- **Versatile**
  - Single or Double Microprocessor Emulation
  - Optional Emulator Package to Handle Conversions from One Target Processor to Another

Product Overview

National Semiconductor's In-System Emulator goes beyond the single-card approach to emulation and qualifies as a genuine innovation in the development of microprocessor-based systems.

ISE is a complete stand-alone unit housing 32K bytes of user programmable memory and all the necessary logic for breakpoints, tracing and memory mapping. Microprocessor emulation is isolated on a single target card containing all the logic needed to emulate the particular microprocessor. ISE is capable of supporting two of these target cards concurrently to achieve emulation in a multiprocessor environment. ISE can support either two target cards for the same microprocessor or two different microprocessors.

There are three important advantages to a stand-alone emulation system over the emulation card approach:

Performance is the primary advantage. An emulation card must share the host system bus and memory. The card not only shares these resources; it also must compete for them in a priority scheme designed into the host system. This creates an unpredictable environment, making real-time emulation impossible.

In contrast, ISE as a stand-alone system has its own special bus designed for high speed emulation. It also has memory dedicated to the user's program, thus eliminating any conflicts and allowing real-time emulation.

Economy is another advantage of the system approach to emulation. The only difference between one emulator card and another is the microprocessor under emulation. The expensive trace memory, breakpoint logic, memory mapping logic, etc., are the same for all microprocessor emulations. The ISE module contains all the logic common to the emulation process while individual target cards are dedicated to the emulation of particular microprocessors. Each target card supported by ISE shares the total system resources, thus eliminating the unnecessary cost of supplying separate logic and memory on each emulator card.

Convenience is the most obvious advantage. The user needs to master only one software package — a single STARPLEX software driver program — which supports all features of ISE and a variety of target cards. Specific characteristics of the emulated microprocessor which must be known by the driver program (register complement, word size, status bits, etc.) are recorded in an architecture ROM located on the target card. The driver program simply reads the contents of the architecture ROM when the system is initialized. It then knows which microprocessor it is emulating and the characteristics of that microprocessor.

The ISE software package is totally integrated into the STARPLEX Development Systems. All of the ease-of-use concepts that set STARPLEX above other development systems are designed into the ISE system.

ISE is called with a single keystroke on the STARPLEX keyboard, as are all other STARPLEX system resources. A fill-in-the-blank menu appears on the CRT and prompts the user to select the microprocessor to be emulated. During the emulation process a portion of the CRT screen is reserved to inform the user of emulation status. This status information includes the type of microprocessor(s) selected for emulation, the state of the emulated microprocessor(s) — running, selected, present — breakpoint condition masks and whether or not breakpoints are enabled.

Should the user wish to review the full range of ISE commands available he can call for "HELP"; the "HELP" key on the STARPLEX keyboard allows the user to display information describing the ISE software functions.

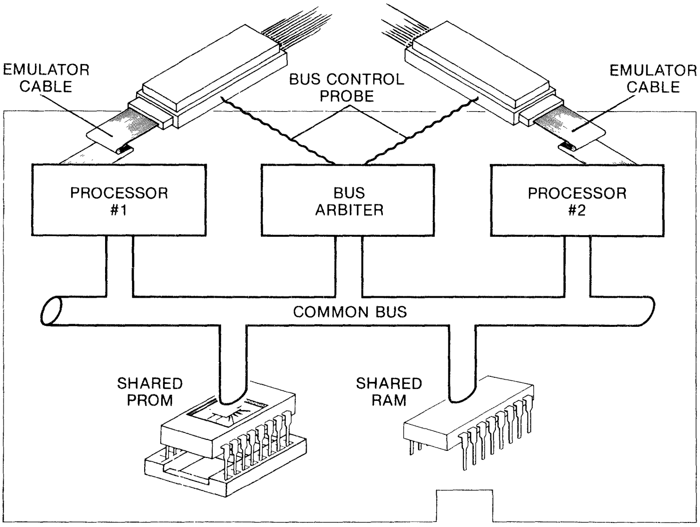


FIGURE 1. Application Multiprocessor System Configuration

Functional Description

Supports Various Microprocessors

The In-System Emulator is a one system solution for users who wish to prototype systems involving one or more types of microprocessors. By changing a target CPU card, ISE can be used to emulate various different microprocessors such as the 8080, 8048 family and 8070 family.

Multiprocessor Support

Many complex microprocessor-based systems use two or more microprocessors in a distributed or multiprocessor configuration. ISE will accommodate two target cards, and will support two microprocessors operating on a common on-board bus, such as National's MICROBUS .

All memory mapping, trace and breakpoint features are available for multiprocessor emulation whenever both processors are on a common bus. When the two microprocessors are emulated in this system, breakpoints can be set on either of the microprocessors, and the trace memory will record all activities on the common bus, including which processor is on the bus during each cycle.

Multiprocessing emulation is accomplished by connecting two bus control probes from the target card cable assemblies to the application system. When attached to the user system's bus arbitration circuitry, these probes enable the user to direct ISE to dynamically monitor the target card on the common bus.

Powerful Debugging Capability

National's ISE provides all the usual features of a powerful In-System Emulator, plus many more that make it the most powerful unit available today. The usual features include: program loading from the host mass storage unit to the ISE program memory; saving programs in the ISE memory on the host system's mass storage medium; memory examination and modification; register examination and modification. Some of the additional and more powerful characteristics include:

- Real-Time Emulation of the Target Microprocessor

  Real-time emulation means that the target microprocessor is emulated in an applications system with the same hardware and software timing characteristics that the microprocessor chip will exhibit when it is plugged into the system. Real-time emulation has been designed into ISE. Some design characteristics contributing to real-time emulation are:

  — Separation of the Host Development System Function. Separating ISE from the host development system is a major contribution to real-time emulation. ISE uses a separate internal bus from the host system, thus eliminating bus access conflicts between the emulation function and the host control functions. Its internal structure is optimized for microprocessor emulation, and is not compromised by some predefined architecture. However, via the RS232C link and the driver program in the host, ISE is able to u$e all of the host system peripherals.

  — System Clock Selection. In the early stages of the applications system checkout, where minor timing variations are more easily tolerated, the applications system designer may choose to run the emulation using the ISE system clock. In the final checkout stages, where real-time emulation is much more critical, the designer may choose to run the emulation using the applications system clock. ISE will support either mode of operation.

  — Positioning of the Emulator Processor. Propagation delays in cables and buffers can contribute significant timing errors to the emulation process. For this reason, the emulation processor is located on a cable card only eight inches from the emulation plug to the applications system microprocessor socket. High speed buffers are used to transmit signals between the emulation processor and the applications system.

  — Emulation Processor Selection. Wherever possible an exact copy of the microprocessor being emulated is used as an emulation processor. For example, when an 8080 microprocessor is being emulated, an 8080 is used as the emulation processor. Instruction execution times and control signal timing are therefore identical to the timing that will be experienced in the final system.

- Thirty-Five Breakpoint Conditions

  Two breakpoint registers (BPC) can be defined on a 32-bit maskable word. Each breakpoint register is specified by:
  — 16 bits of address
  — 8 bits of target CPU status
  — 8 bits of user hardware status

  Each bit of the 32-bit breakpoint register mask may be specified to compare on "1" or "0," or "don't care."

  The user can then specify a breakpoint to occur when any one of the following conditions is met:
  — If BPC #1 is met
  — If BPC #2 is met
  — If BPC #1 or BPC #2 is met
  — If BPC #1 is met after BPC #2 is met
  — If BPC #2 is met after BPC #1 is met

ISE can also be told to "coast" after the breakpoint combination has been satisfied before suspending operation:
- Coast until n more BPs are encountered
- Coast until n more BPC #1s are encountered
- Coast until n more BPC #2s are encountered
- Coast until n more read/write cycles are encountered
- *Coast until n more instruction fetches are encountered
- *Coast until n more memory read/write cycles are encountered
- *Coast until n more I/O read/write cycles are encountered

Note: 0 < n < 256.

There are five Breakpoint (BP) combinations and seven "Coast" combinations, making a total of thirty-five total combinations.

- Program Trace

ISE maintains a constant record, in real-time, of the last 128 cycles performed by the target microprocessor. Forty bits of information are recorded for each cycle:
- 16 bits of address
- 8 bits of data
- 8 bits of CPU status
- 8 bits of user-defined status, via the 8-bit status cable

The type of information recorded in the trace memory is selectable in four ways:
- All read/write cycles
- *Instruction fetches only
- *Memory read/write cycles only
- *I/O read/write cycles only

ISE generates a Sync Pulse each time data is recorded in the trace memory. In addition, the user may specify that the applications program be halted after 64 words are recorded in the trace memory.

- Target Card Control Features

The target microprocessor will be placed in an inactive state at the end of the current instruction when one of the following conditions occurs:
- The user gives a halt-command to the given target
- A breakpoint is encountered
- A memory protect violation is encountered
- Trace memory is filled with 64 words of new data, when specified
- In single step mode

When a target is halted, the user may take any one or all of the following actions:
- Examine and change the target's internal registers
- Examine and change program memory
- Dump trace memory for examination
- Change trace specifications
- Change memory map

- Flexible Memory Mapping

A 32K address space is available in ISE. A maximum of 32K bytes of the applications program may be mapped into ISE memory in 512-byte blocks. These blocks need not be contiguous. The memory map may be specified and altered under program control, and any segment may be write protected. In addition, data may be copied from the applications system memory.

- Microsecond Timer

National's ISE has a 16 second timer which counts in one-microsecond increments. The user may use this timer to measure the time elapsed between any two points of his program. The two points in the program must be defined through breakpoint conditions; the clock starts counting as breakpoint condition #1 is encountered and stops when breakpoint condition #2 is encountered.

- User Status Cable

ISE provides the user with a four foot cable carrying eight probes. The user may hook these probes anywhere in his system and treat the status of these points as part of his breakpoint word and trace word.

Convenient Software

Several tools are provided to make ISE a very convenient emulation system to use. Many of the debugging features available for software development, like symbolic debugging, are now available for system development.

- Symbolic Debugging

Programmers use symbols to reference program and data memory when writing programs, but they are usually required to use absolute hexadecimal addresses when referencing those locations during program debug. ISE allows the designer to use those same symbols to reference program and data memory during program debug. A symbol table is generated when the program is first assembled or compiled in the host development system. That symbol table is passed to the driver program in the host system for use during the debugging operations. During debugging operations, symbols may be added or deleted, and symbol values may be redefined.

*If the target microprocessor puts out necessary status information in one form or another.

- In-Line Assembler

  A one-pass in-line assembler is provided to allow modification of object code in ISE memory or the applications system memory without having to manually convert symbolic instructions to machine language. The in-line assembler accepts program modifications in the assembly language of the target microprocessor, assembles them, and inserts them into the object program at the locations specified by the system programmer.

- Disassembler

  The disassembler examines specified segments of ISE or applications system memory, disassembles them, and displays their contents in the assembly language mnemonics of the target microprocessor. This feature eliminates many of the tedious manual steps normally involved in applications system debug.

- Automatic Testing

  The application system designer often wishes to perform a predefined sequence of tests on the system over a relatively long period of time. ISE has an automatic testing mode whereby the designer may write a sequence of test steps in a language similar to BASIC, store those tests in the memory of the host system, and initiate the test sequence. ISE will perform the tests in the specified sequence and record the results on a disc or a hard copy device of the host system. Branching and conditional branching are also permitted in the test program. This feature is especially useful for rigorous proof that all parts of the applications system are in fact working, for detecting and documenting infrequent failures, and for performing "life" tests.

  The list of predefined test sequences resides in a file created by using the ISE software or the STARPLEX Text Editor. Once the file is resident on the STARPLEX disc, it can be retrieved, deleted, edited, etc., by the ISE Software Package.

  The following commands allow the user to perform automatic testing functions:

DELETE	Deletes a range of lines from test program.
EXECUTE	Executes the test program.
LIST	Prints the test program to a selected device.
LOAD INFILE	Loads the specified test program fom disc.
SAVE INFILE	Saves the test program on disc.
SCRATCH	Deletes the entire test program.
END	Directive to end test program and return control to command mode.
GOTO	Unconditional branch to another statement in test program.
IF	Conditional branch to another statement in test program.
INPUT	Enables user to interact with test program at run time to specify data values.
PRINT	Prints number and string data on console.

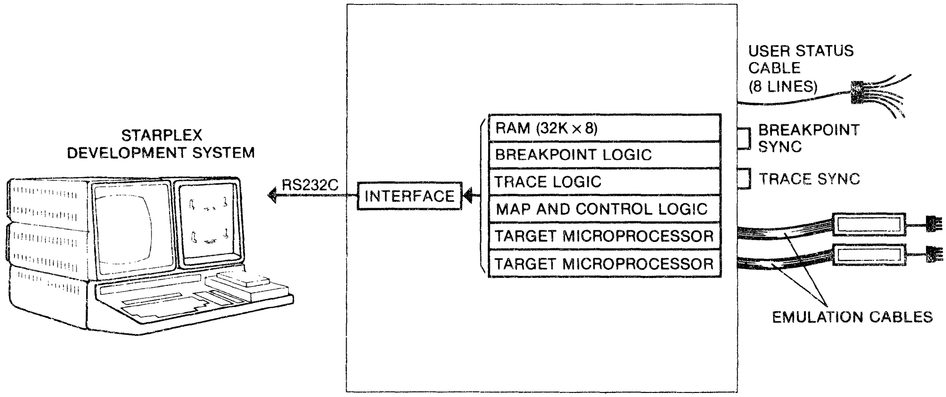


FIGURE 2. In-System Emulator System Configuration

ISE Commands

Initialization and Setup Commands

SELECT or TARGET	Select target processor
LATCH	Control input from user status cable.
RESET	Set flag to reset selected target processor prior to resumption of emulation.
NORESET	Rescind RESET command.
PRINT	Select host processor's output device.
RADIX or BASE	Establish default input and display mode (binary, decimal, hexadecimal, or octal).
ARCHIVE	Save system's status on nonvolatile storage medium for later retrieval.
RESTORE	Restore system status saved by ARCHIVE command.

Emulation Commands

RUN	Continue system emulation until break condition is encountered.
STEP	Continue system emulation in single-step mode.

Memory/Register Modification and Display Commands

CHANGE	Alter contents of memory locations and registers with new data values.
DISPLAY or DUMP	Display portions of target processor memory, register or trace data.
MOVE	Transfer a region of memory into another region.
SEARCH	Search a range of memory locations for a specified value and display the location.

Breakpoint Control Commands

BREAK	Suspend emulation when specified break conditions are met in target system.
TIME	Display time between breakpoints.

Trace Control Commands

TRACE	Select target activity to be recorded into trace memory.

Memory Mapping/Demapping Control

MAP	Copy a specified memory range to or from target memory and ISE memory.
DEMAP or NOMAP	Restore a memory range previously mapped by MAP command.

Symbol Table and File Manipulation

DELETE	Delete specified symbol(s) from symbol table.
LOAD	Fetch symbol table or load file from storage medium.
SAVE	Create symbol table or load file on storage medium.

Specifications

Memory	Mappable — 32K bytes Trace — 128 × 40 bits
Power	115 VAC, 60 Hz 230 VAC, 50 Hz 345 Watts
Physical	Height — 11.31 in. (28.7 cm) Width — 13.0 in. (33.0 cm) Depth — 15.95 in. (40.5 cm) Weight — 35 lbs. (15.9 kg)

Prerequisite

Any STARPLEX/STARPLEX II Development System.

Order Information

For STARPLEX Development Systems:

SPM-A08	In-System Emulator Module
SPM-A09-1	8080 Emulator Package
SPM-A09-2	8048 Family Emulator Package
SPM-A09-3	8070 Series Emulator Package

For STARPLEX II Development Systems:

SPM-90-A08	In-System Emulator Module
SPM-90-A09-1	8080 Emulator Package
SPM-90-A09-2	8048 Family Emulator Package
SPM-90-A09-3	8070 Series Emulator Package

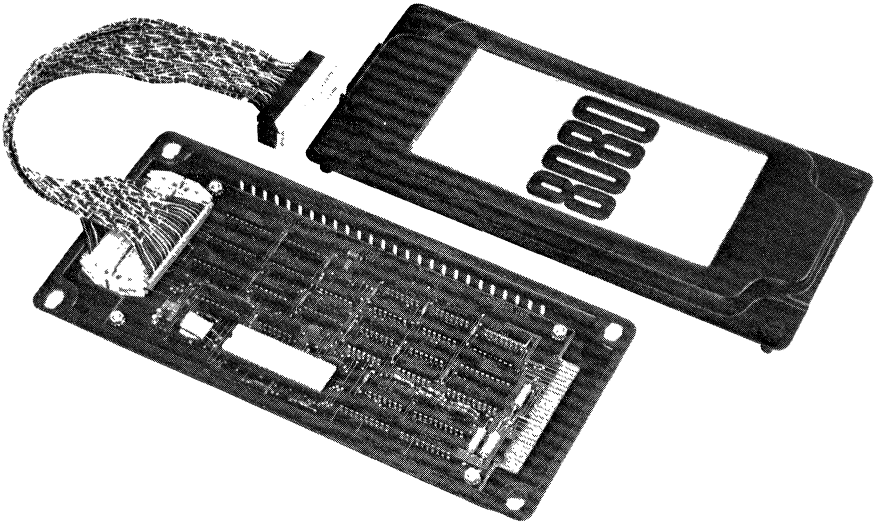
Note: With the exception of the SPM-A09-1, 8080 Emulator Package, all the other emulator packages listed above include appropriate cross-assemblers and manuals. The 8080/8085 macroassembler for the SPM-A09-1 emulator package is included with the STARPLEX Development System operating system diskettes set.

Documentation

420305869-001	ISE System Reference Manual
420305789-001	8080/8085 Macroassembler Software User's Manual (Part of STARPLEX System Reference Manual, Vol II, 950305891-002)
420305653-001	8080 In-System Emulator Target Board User's Manual
420306064-001	8048 Family Cross-Assembler Software User's Manual
420306065-001	8048 In-System Emulator Target Board User's Manual
420306123-001	8070 Family Cross-Assembler Software User's Manual
420306132-001	8070 In-System Emulator Target Board User's Manual

National Semiconductor

8080 Emulator Package



- **True Real-Time Emulation of 8080 Microprocessor (2 MHz)**
- **Includes Target Card, Cable Card with Cables and Complete Software**
- **Supports Three Modes of Operation**
  - Program development
  - Single processor emulation
  - Multiprocessor emulation
- **Contains 8080 Architectural Firmware, Status Decode Logic and CPU Control Circuitry**
- **Used with In-System Emulator (ISE™) Module**

Product Overview

The 8080 Emulator Package, in conjunction with the In-System Emulator Module, provides emulation capabilities for the user that are not available in other emulators. National's In-System Emulator Module is a complete unit housing 32K bytes of real time map memory and all the necessary logic for breakpoints, tracing and memory mapping. These resources are available for the emulation of other processors since the individual emulation packages are the only components dedicated to particular processors. This approach simplifies changing processors since the user needn't learn a new machine each time he changes.

The 8080 Emulator Package provides the physical and electrical interface between the In-System Emulator Module and an 8080 based system undergoing development. A target board resides in the Emulator Module and connects to the system under development by way of a cable board with associated cables. One of these cables connects the target board to the cable board containing the 8080 microprocessor; the other connects the cable board to the user system. This arrangement allows the user to take full advantage of ISE module features.

The target board has three modes of operation: program development, single processor emulation, and multiprocessor emulation. The program development mode permits the user to develop and debug his software even though he has no hardware available. During the emulation of a single processor, the user may select the clock signals for the 8080 from either his hardware or from the emulator. Multiprocessor emulation enables the user to emulate both the 8080 and another processor operating in the target system.

Specifications

Environmental

Operating Temperature 10°C (50°F) to 30°C (90°F) in a controlled environment

Storage Temperature −40°C to 75°C

Power

$+5\,V_{DC}$ @ 1.94 A (Typical)
$-5\,V_{DC}$ @ 1.00 A (Typical)
$-12\,V_{DC}$ @ 0.08 A (Typical)
$+12\,V_{DC}$ @ 0.05 A (Typical)

Physical

Target Board Dimensions — 6.75 × 12.00 inches

Cable Board Dimensions — 3.8 × 8.9 inches

Cable-Board Box Dimensions — 4.7 × 9.8 inches

Target/Cable-Board Cables —
Internal: 14 inches
External: 48 inches

Cable-Board/User Cable Length — 10 inches

Prerequisites

Any STARPLEX^{TM}/STARPLEX II^{TM} Development Systems and In-System Emulator Module (SPM-A08, SPM-90-A08)

Bus Enable Lines S1 and S2

	INPUT LOAD		OUTPUT DRIVE		
PIN	$I_{ih}\mu A$	$I_{il}\mu A$	$I_{oh}mA$	$I_{ol}mA$	Mnemonic
1			−2 6	16	A10
2	—	—	—	—	GND
3	80	−200	−5	20	D4
4	80	−200	−5	20	D5
5	80	−200	−5	20	D6
6	80	−200	−5	20	D7
7	80	−200	−5	20	D8
8	80	−200	−5	20	D2
9	80	−200	−5	20	D1
10	80	−200	−5	20	D0
11	—	—	—	—	−5V
12	20	−360			RESET
13	20	−360			HOLD
14	20	−360			INT
15	30	−370	70 A	2.1	02
16			−400 μA	8	INTE
17			−400 μA	8	DBIN
18			−400 μA	8	$\overline{\text{WR}}$
19			−400 μA	8	SYNC
20	—	—	—	—	+5V
21			−400 μA	8	HLDA
22	10	−10	90 μA	2 5	01
23	20	−360			READY
24			−400 μA	8	WAIT
25			−2.6	16	A0
26			−2.6	16	A1
27			−2.6	16	A2
28	—	—	—	—	+12V
29			−2 6	16	A3
30			−2.6	16	A4
31			−2 6	16	A5
32			−2.6	16	A6
33			−2.6	16	A7
34			−2.6	16	A8
35			−2.6	16	A9
36			−2.6	16	A15
37			−2.6	16	A12
38			−2.6	16	A13
39			−2.6	16	A14
40			−2.6	16	A11
S1	+40 μA	−720 μA			BUS ENABLE1
S2	40 μA	−720 μA			BUS ENABLE2

Order Information

(Includes Target Board, Lightweight Plastic Cable Pod, Cables, Software for 8080 Display Charge for Mnemonic Assembly and Disassembly. SPM-90-A09-1 also includes 8080 Cross-Assembler.)

For STARPLEX Development Systems:

SPM-A08	In-System Emulator Module
SPM-A09-1	8080 Emulator Package

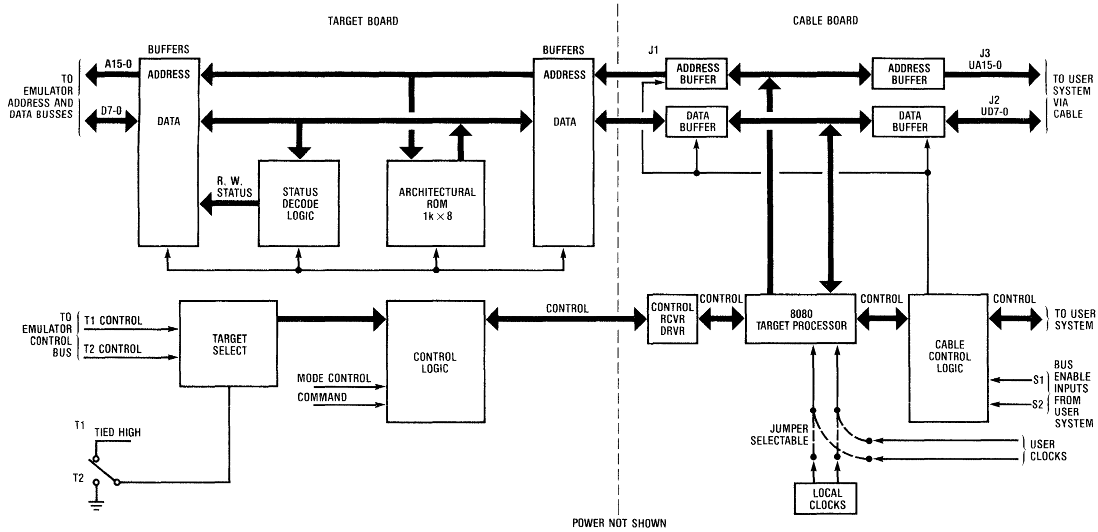
For STARPLEX II Development Systems:

SPM-90-A08	In-System Emulator Module
SPM-90-A09-1	8080 Emulator Package

Documentation

420305869-001	ISE Module Reference Manual
420305653-001	8080 In-System Emulator Target Board User's Manual*
420305789-001	8080/8085 Macroassembler Software User's Manual (Part of STARPLEX System Reference Manual, Vol. II, 420306383-001)

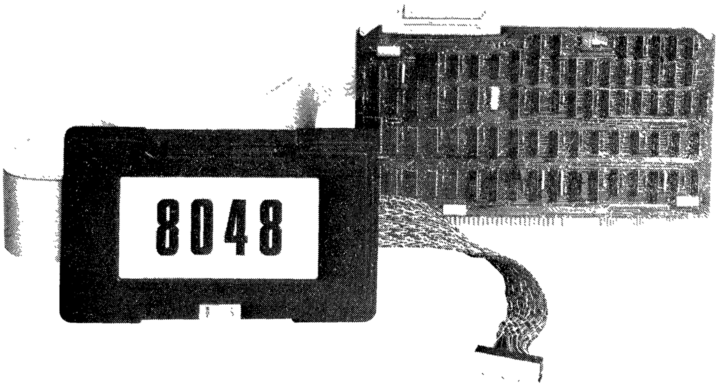
*Included with 8080 Emulator Package (SPM-A09-1, SPM-90-A09-1).



Target and Cable Boards, Simplified Block Diagram

National Semiconductor

8048 Family Emulator Package



- **Real Time Emulation of 8048 Microcomputer Family Devices (8035, 8039, 8040, 8048, 8049, 8050)**
- **Supports Both 6 and 11 MHz Versions of 8049 and 8050 with Crystal Change**
- **4K Bytes of On-Board RAM allows disassembly and debug of program ROM**
- **Used with In-System Emulator (ISE™) Module**
- **Supports Three Modes of Operation**
  - Program Development
  - Single Processor Emulation
  - Multiprocessor Emulation
- **Contains 8048, 8049, 8050 Architectural Firmware, Status Decode Logic and CPU Control Circuitry**

Product Overview

National's 8048 Emulator Package gives the designer of 8048 based systems the kind of sophisticated tool required for efficient microcomputer development. The 8048 Emulator Package, in conjunction with the In-System Emulator, provides capabilities that have not been available in this type of instrument.

National's In-System Emulator Module is a complete unit housing 32K bytes of real time map memory and all the necessary logic for break points, tracing and memory mapping. These resources are available for the emulation of other processors since the individual emulation packages are the only components dedicated to particular processors. This approach simplifies changing processors since the user needn't learn a new machine each time he changes. The 8048 Emulator Package provides capabilities which focus on the problems of designing with single-chip microcomputers. The target card has its own 4K bytes of RAM dedicated to the purpose of emulating the processor's program ROM. The designer has complete access to this memory during emulation. He may examine and disassemble existing ROM contents into the dedicated RAM, make changes and execute the altered code. This gives the user considerable flexibility in new product design as well as in debugging existing systems containing a previously masked 8048. Also, the unit supports the 6 or 11 MHz versions of the 8048.

The 8048 Emulator Package provides the physical and electrical interface between the In-System Emulator module and an 8048 based system undergoing development. A target board resides in the module and connects by way of the cable board and 40-pin plug to the system under development. The system supports three modes of operation. These modes are program development, single processor emulation and multiprocessor emulation.

The program development mode permits the user to develop and debug his software even though he has no prototype hardware available. The emulator package provides the clocks and memory necessary for this task. During emulation of a single processor, the user may select clock signals for the 8048 from the prototype hardware or from the emulator. Multiprocessor emulation enables the user to emulate both the 8048 and another processor operating in the same target system.

Specifications

Environmental Operating Temperature 10°C (50°F) to 30°C (90°F) in a controlled environment

Storage Temperature −40°C to 75°C

Power +5 V_{DC} @ 2.6 A (Typical)
−5 V_{DC} @ 4.00 mA (Typical)
+12 V_{DC} @ 50 mA (Typical)

Physical Target Board Dimensions — 6.75 x 12.00 inches

Cable Board Dimensions — 4.5 x 9.0 inches

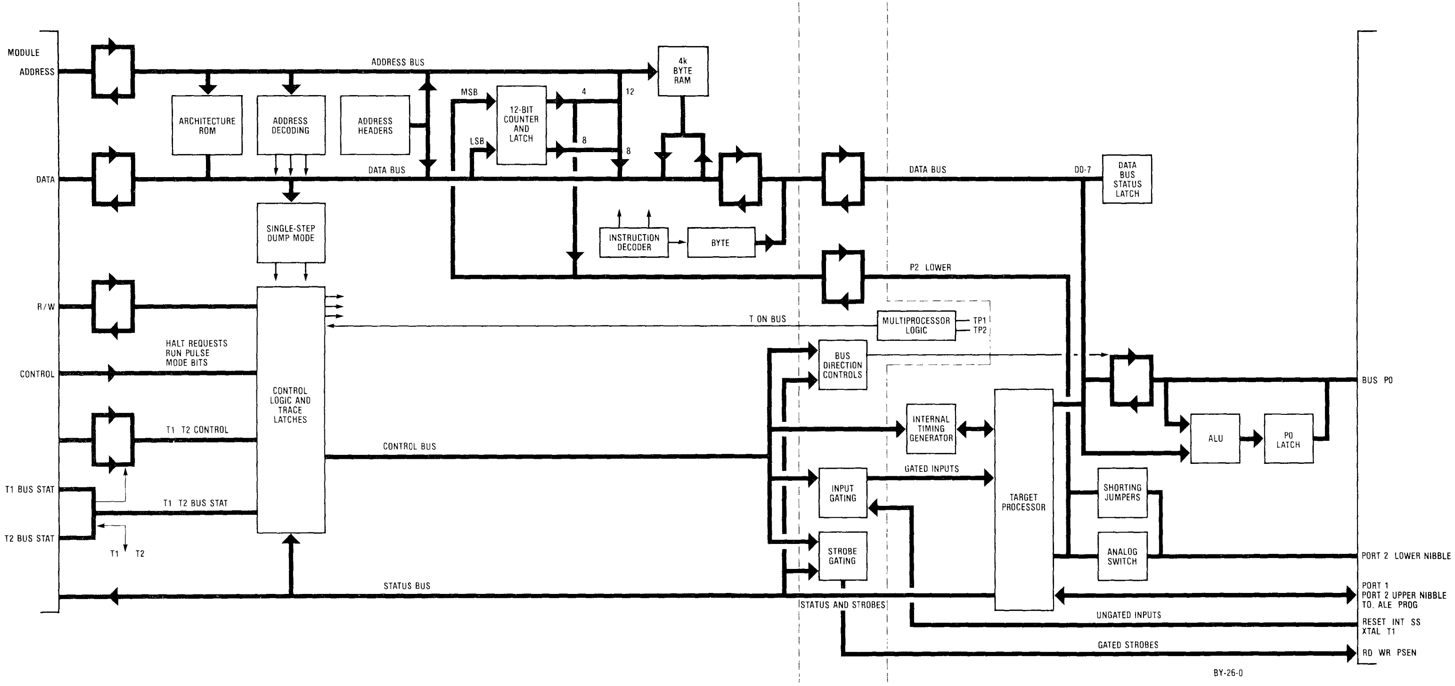
Cable-Board Housing — 4.8 x 10.0 inches

Target Board Cables —
Internal: 14 inches
External: 48 inches

Cable-Board/User Cable Length — 10 inches

Loading Information for 40-Pin Connector and Bus Enable Lines

	INPUT LOAD		OUTPUT DRIVE		
PIN	I_{IH}	I_{IL}	I_{OH}	I_{OL}	Mnemonic
1	—	10 μA	—	—	T0
2	—	10 μA	—	—	XTAL$_1$
3	—	10 μA	—	—	XTAL$_2$
4	—	400 μA	10 μA	—	$\overline{RESET}$
5	20 μA	360 μA	—	—	$\overline{SS}$
6	20 μA	360 μA	—	—	$\overline{INT}$
7	20 μA	360 μA	—	—	EA
8	—	—	400 μA	8 mA	$\overline{RD}$
9	—	—	400 μA	8 mA	$\overline{PSEN}$
10	—	—	400 μA	8 mA	$\overline{WR}$
11	—	—	400 μA	8 mA	ALE
12*	200 μA	5 mA	10 mA	43 mA	D_{B0}
13*	200 μA	5 mA	10 mA	43 mA	D_{B1}
14*	200 μA	5 mA	10 mA	43 mA	D_{B2}
15*	200 μA	5 mA	10 mA	43 mA	D_{B3}
16*	200 μA	5 mA	10 mA	43 mA	D_{B4}
17*	200 μA	5 mA	10 mA	43 mA	D_{B5}
18*	200 μA	5 mA	10 mA	43 mA	D_{B6}
19*	200 μA	5 mA	10 mA	43 mA	D_{B7}
20	—	—	—	—	V_{SS}
21	—	250 μA	125 μA	1.8 mA	P20
22	—	250 μA	125 μA	1.8 mA	P21
23	—	250 μA	125 μA	1.8 mA	P22
24	—	250 μA	125 μA	1.8 mA	P23
25	—	—	100 μA	2 mA	PROG
26*	100 mA	—	10 μA	—	V_{DD}
27	—	125 μA	125 μA	2 mA	P10
28	—	125 μA	125 μA	2 mA	P11
29	—	125 μA	125 μA	2 mA	P12
30	—	125 μA	125 μA	2 mA	P13
31	—	125 μA	125 μA	2 mA	P14
32	—	125 μA	125 μA	2 mA	P15
33	—	125 μA	125 μA	2 mA	P16
34	—	125 μA	125 μA	2 mA	P17
35	—	125 μA	125 μA	2 mA	P24
36	—	125 μA	125 μA	2 mA	P25
37	—	125 μA	125 μA	2 mA	P26
38	—	125 μA	125 μA	2 mA	P27
39	20 μA	350 μA	—	—	T1
40	—	—	—	—	V_{CC}
TP1	20 μA	720 μA			Bus Enable 1
TP2	20 μA	720 μA			Bus Enable 2



8048 Family Emulator Package Block Diagram

Prerequisites

Any STARPLEX™/STARPLEX II™ Development Systems and In-System Emulator™ Module (SPM-A08, SPM-90-A08)

Order Information

(Includes Target Board, Lightweight Plastic Cable Pod, Cables, Software for 8048 Display Charge for Mnemonic Assembly and Disassembly, 8048 Cross-Assembler, Upgrades fro 8049 and 8050 ISE.)

For STARPLEX Development Systems:

SPM-A08	In-System Emulator Module
SPM-A09-2	8048 Family Emulator Package

For STARPLEX II Development Systems:

SPM-90-A08	In-System Emulator Module
SPM-90-A09-2	8048 Family Emulator Package

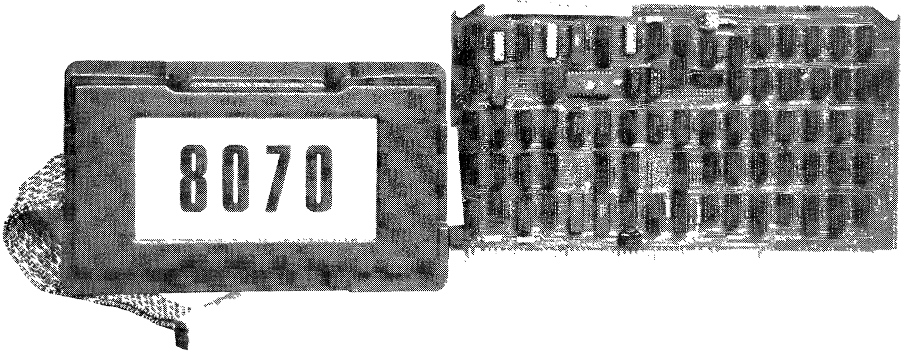
Documentation

420305869–001	ISE Module Reference Manual
420306064–001	8048 Family Cross-Assembler Software User's Manual*
420306065–001	8048 In-System Emulator Target Board User's Manual*

*Included with 8048 Family Emulator Package (SPM-A09-1, SPM-90-A09-1).

National Semiconductor

8070 Series Emulator Package



- **Real-Time Emulation of 8070 Series Microcomputer Devices (8070, 8072, 8073)**
- **2.5K Bytes of On-Board RAM to Simulate On-Board ROM**
- **Used with In-System Emulator (ISE™) Module**
- **Supports Three Modes of Operation**
  - Program development
  - Single processor emulation
  - Multiprocessor emulation
- **Contains 8070, 8072, 8073 Architectural Firmware, Status Decode Logic and CPU Control Circuitry**

Product Overview

National's 8070 Series Emulator Package gives the designer of 807x-based systems the kind of sophisticated tool required for efficient microcomputer development. The 8070 Series Emulator Package, in conjunction with the In-System Emulator, provides capabilities that have not been available in this type of instrument.

National Semiconductor's In-System Emulator (ISE) Module is a complete unit housing 32K bytes of real-time map memory and all the necessary logic for breakpoints, tracing and memory mapping. These resources are available for the emulation of other processors since the individual emulation packages are the only components dedicated to particular processors. This approach simplifies changing processors since the user needn't learn a new set of ISE commands each time he changes.

The 8070 Series Emulator Package provides capabilities which focus on the problems of designing with single-chip microcomputers. The target board has its own 2.5K bytes of RAM dedicated to the purpose of emulating the processor's program ROM. The designer has complete access to this memory during emulation.

The 8070 Series Emulator Package provides the physical and electrical interface between the In-System Emulator module and an 807x-based system undergoing development. A target board resides in the module and connects by way of the cable board and a 40-pin plug to the system under development. The system supports three modes of operation. These modes are program development, single processor emulation and multiprocessor emulation.

The program development mode permits the user to develop and debug his software even though he has no prototype hardware available. The emulator package provides the clocks and memory necessary for this task. During emulation of a single processor, the user may select clock signals for the 807x from the prototype hardware or from the emulator. Multiprocessor emulation enables the user to emulate both the 807x and another processor operating in the same target system.

The 807x Series microprocessors are presently available in the following configurations:

μP	ROM Size	RAM Size
8070	None	64 bytes
8072	2.5 K bytes	64 bytes
8073	2.5 K* bytes	64 bytes

*Pre-programmed with National Semiconductor's "Tiny BASIC™".

Specifications

Environmental Operating Temperature 10°C (50°F) to 30°C (90°F) in a controlled environment

Storage Temperature −40°C (−40°F) to 75°C (167°F)

Power +5 V_{DC} @ 3.22 A (Max.)
−5 V_{DC} @ 45 mA (Max.)
+12 V_{DC} @ 64 mA (Max.)

Physical Target Board Dimensions—6.75 x 12.00 inches

Cable Board Dimensions—4.5 x 9.0 inches

Cable-Board Housing Dimensions—4.8 x 10.0 inches

Target Board Cables—
Internal: 14 inches
External: 48 inches

Cable-Board/User Cable Length—10 inches

Approximate overall length from Emulator Module to the 40-pin connector—5.5 ft.

Loading Information for 40-Pin Connector and Bus Enable Lines

	INPUT LOAD		OUTPUT DRIVE		
PIN	I_{ih} mA	I_{il} mA	I_{oh} mA	I_{ol} mA	Mnemonic
1	—	—	−2.6	16	NENOUT
2	0.02	−0.36	—	—	NENIN
3	0.02	−0.36	0	15.6	NBREQ
4	—	—	−5.2	32	NRDS
5	0.02	−0.36	—	—	NHOLD
6	—	—	−5.2	32	NWDS
7	—	—	−2.6	16	XOUT
8	0.02	−0.36	—	—	XIN
9	—	—	−2.6	16	A15
10	—	—	−2.6	16	A14
11	—	—	−2.6	16	A13
12	—	—	−2.6	16	A12
13	—	—	−2.6	16	A11
14	—	—	−2.6	16	A10
15	—	—	−2.6	16	A9
16	—	—	−2.6	16	A8
17	—	—	−2.6	16	A7
18	—	—	−2.6	16	A6
19	—	—	−2.6	16	A5
20	—	—	—	—	GND
21	—	—	−2.6	16	A4
22	—	—	−2.6	16	A3
23	—	—	−2.6	16	A2
24	—	—	−2.6	16	A1
25	—	—	−2.6	16	A0
26	0.08	−0.2	−3	8	D7
27	0.08	−0.2	−3	8	D6
28	0.08	−0.2	−3	8	D5
29	0.08	−0.2	−3	8	D4
30	0.08	−0.2	−3	8	D3
31	0.08	−0.2	−3	8	D2
32	0.08	−0.2	−3	8	D1
33	0.08	−0.2	−3	8	D0
34	—	—	−2.6	16	F1
35	—	—	−2.6	16	F2
36	—	—	−2.6	16	F3
37	0.001	−0.001	—	—	NRST
38	0.02	−0.4	—	—	SA
39	0.02	−0.4	—	—	SB
40	—	—	—	—	VCC

Note: The above loading information generally provides higher drive capability than the actual 8070, 8072 or 8073 devices. The input loading is nominally equal to the same devices.

Prerequisites

Any STARPLEX™/STARPLEX II™ Development Systems and In-System Emulator Module (SPM-A08, SPM-90-A08)

Order Information

(Includes Target Board, Lightweight Plastic Cable Pod, Cables, Software for 8070 Display Charge for Mnemonic Assembly and Disassembly, 8070 Cross-Assembler Upgrades for 8072 and 8073 ISE.)

For STARPLEX Development Systems:

SPM-A08	In-System Emulator Module
SPM-A09-3	8070 Series Emulator Package

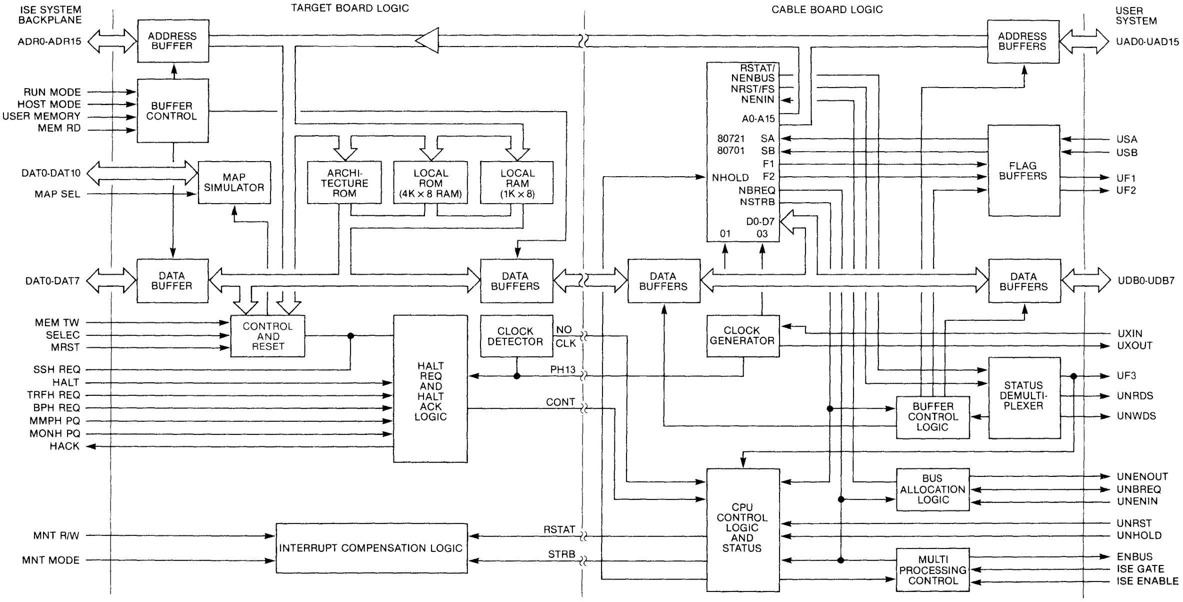
For STARPLEX II Development Systems:

SPM-90-A08	In-System Emulator Module
SPM-90-A09-3	8070 Series Emulator Package

Documentation

420305869-001	ISE Module Reference Manual
420306123-001	8070 Series Cross-Assembler Software User's Manual*
420306132-001	8070 In-System Emulator Target Board User's Manual*

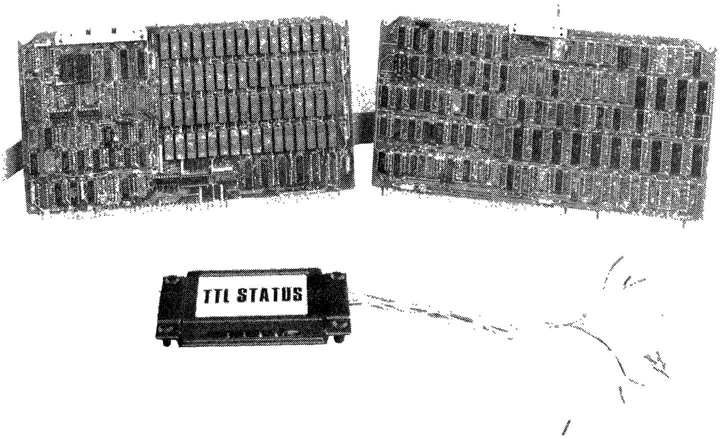
*Included with 8070 Series Emulator Package (SPM-A09-3, SPM-90-A09-3).



Block Diagram, 8070 Emulator

National Semiconductor

Integral In-System Emulator (ISE™) Package



- **Real-time emulation of 8-Bit microprocessors**
  - Full Support for 80CX48 Family, 8085 Series, NSC800™ and Z80 Microprocessors
- **Combined with an emulator board, will operate in any STARPLEX™/ STARPLEX II™ Development System**
- **An integral emulation system**
  - Hardware
    - — Processor independent
    - — 32K byte mapped memory
    - — Two 32-bit breakpoint registers, each bit programmable
    - — 256 × 40-bit trace memory
    - — Memory mapping in 1K byte increments
    - — 8-bit user status cable for custom breakpoint and trace operations
    - — Real-time counter in microseconds up to 16 seconds.
  - Software
    - — Host system resident command driver
    - — Host system resident mnemonic assembler and disassembler
    - — Coast after breakpoint provided with variable length and user-defined qualifications
- **Easy to use**
  - In-File for Automatic Test
  - Consistent Commands
  - Symbolic Debugger
  - Full Access to STARPLEX/ STARPLEX II Development Systems Facilities (e.g., access to STARPLEX/ STARPLEX II Editor and other utility programs)
- **Optional emulator packages to handle conversion from one target processor to another**

Product Overview

The Integral ISE consists of two logic boards in standard STARPLEX/STARPLEX II configuration, one bus connector and a cable. In addition to this are manuals and user software. The two logic boards provide all the necessary logic for breakpoints, tracing, and real-time memory mapping. Microprocessor emulation is isolated on a required single optional target board containing all the logic needed to emulate the particular microprocessor. Together with any specified target processor, which is not part of the Integral ISE, the three boards can be installed in any STARPLEX/STARPLEX II Development System. When installed directly in a STARPLEX/STARPLEX II Development System, the Integral ISE supports only single processor emulation.

(From this point on "STARPLEX" will signify "STARPLEX/STARPLEX II".)

There are three very important advantages to this approach to system emulation:

Economy is the prime advantage. The customer needs to purchase exactly what his application requires. For simple single processor applications, the user can install the Integral ISE directly into any STARPLEX Development System without being required to purchase an entire emulation chassis. Since the Trace and Mapped Memory boards are standard logic modules, the customer will require only one set of these boards in most applications, whereas he might have several different types of target modules. In this manner, the user would be allowed to change his target module set-ups for one processor to another quickly and conveniently without changes of any kind to the Trace and Mapped Memory boards.

Convenience is an obvious advantage. The user need only master one software package—a single host software driver program—which supports all the features of the Integral ISE and its entire set of compatible target boards. Specific characteristics of the emulated microprocessor which must be known by the driver program (e.g., register complement, word size, status bits, etc.) are recorded on a "target specific" diskette which is supplied with each different target board. The driver program upon initialization reads the target board status which identifies the target processor device type. This information together with the data contained on the "target specific" diskette allows the software driver to display data to the user in a syntax consistent with each processor type.

The Integral ISE software package is totally integrated into a STARPLEX Development System. All of the ease-of-use concepts that set the STARPLEX above other development systems are designed into the Integral ISE system.

The software is invoked with a single keystroke on a STARPLEX keyboard, as are all other STARPLEX system resources. A fill-in-the-blank menu appears on the CRT and prompts the user to select the microprocessor to be emulated. During the emulation process a portion of the CRT screen is reserved to inform the user of emulation status. This status information includes the type of microprocessor(s) selected for emulation, the state of the emulated microprocessor(s), breakpoint condition mask, and whether or not breakpoints are enabled.

Should the user wish to review the full range of the Integral ISE commands available he can call for "HELP"; the "HELP" key on a STARPLEX keyboard allows the user to display information describing the Integral ISE software functions.

Performance is the final advantage. Unlike other in-system emulators which are installed directly into a development system, National's Integral ISE does not have to compete with the system bus in order to attain real-time emulation, either mapped or unmapped. Even though the Trace and Mapped Memory boards are physically within the development system, they do not interface directly with the system bus. They interface only with the Target Board through a specialized high-speed emulation bus connector. Only the Target Board has the capability of interfacing to the system bus. The Mapped Memory board is dedicated to Integral ISE and does not occupy any STARPLEX Development System address space.

Functional Description

Support Various Microprocessors

The Integral ISE is a flexible solution for users who wish to prototype systems involving one or more types of microprocessors. By changing a target CPU board, Integral ISE can be used to emulate various different microprocessors such as the 8085 NSC800, 80CX48 and Z80 microprocessors.

Powerful Debugging Capability

National Semiconductor's Integral ISE provides all the usual features of a powerful in-system emulator, plus many more that make it the most powerful unit available today. The usual features include: program loading from the host mass storage unit to the Integral ISE program memory; saving programs in the Integral ISE on the host system's mass storage med ium; memory examination and modification; register examination and modification. Some of the additional and more powerful characteristics include:

- Real-Time Emulation of the Target Microprocessor

  Real-time emulation means that the target microprocessor is emulated in an applications system with the same hardware and software timing characteristics that the microprocessor chip will exhibit when it is plugged into the application system. Real-time emulation has been designed into the Integral ISE. Some design characteristics contributing to real-time emulation are:

  — Separation of the Host Development System Function. Separation of Integral ISE from the host development is a major contribution to real-time emulation. Integral ISE uses a separate internal bus from the host system, thus eliminating bus access conflicts between the emulation function and the host control functions. Its internal structure is optimized for microprocessor emulation, and is not compromised by some predefined architecture.

—System Clock Selection. In the early stages of the applications system checkout, where minor timing variations are more easily tolerated, the applications system designer may choose to run the emulator using the Integral ISE system clock. In the final checkout stages, where real-time emulation is much more critical, the designer may choose to run the emulator using the applicaton system's own clock. Integral ISE will support either mode of operation.

—Positioning of the Emulator Processor. Propagation delays in cables and buffers can contribute significant timing errors to the emulation process. For this reason, the emulation processor is located on a cable board only eight inches from the emulation plug to the applications system microprocessor socket. High-speed buffers are used to transmit signals between the emulation processor and the applications system.

—Emulation Processor Selection. Wherever possible an exact copy of the microprocessor being emulated is used as an emulation processor. For example, when an 8085 microprocessor is being emulated, an 8085 is used as the emulation processor. Instruction execution times and control signal timing are therefore identical to the timing that will be experienced in the final system.

- Breakpoint Conditions Provided for

Two breakpoint registers (BPC) can be defined on a 32-bit, maskable word. Each breakpoint register is specified by:

—16 bits of address
—8 bits of target CPU status
—8 bits of user hardware status

Each bit of the 32-bit breakpoint register mask may be specified to compare on “$” or “0”, or “don't care”. The user can then specify a breakpoint to occur when any one of the following conditions is met:

—If BPC #1 is met
—If BPC #2 is met
—If BPC #1 or BPC #2 is met
—If BPC#1 is met after BPC#2 is met
—If BPC #2 is met after BPC #1 is met

Integral ISE can also be told to “coast” after the breakpoint combination has been satisfied before suspending operation:

—Coast until *n* more BPs are encountered
—Coast until *n* more BPC #1s are encountered
—Coast until *n* more BPC #2s are encountered
—Coast until *n* more read cycles are encountered
—Coast until *n* more write cycles are encountered
—Coast until *n* more instruction fetches are encountered
—*Coast until *n* more memory read cycles are encountered
—*Coast until *n* more memory write cycles are encountered
—*Coast until *n* more memory read or write cycles are encountered
—*Coast until *n* more I/O read cycles are encountered
—*Coast until *n* more I/O write cycles are encountered
—*Coast until *n* more I/O read or write cycles are encountered
—*Coast until *n* more interrupt acknowledges
—*Coast until *n* more serial input data
—*Coast until *n* more serial output data
—Coast until *n* more of all the above

Note: $0 < n < 256$; Those coast options preceded by a * are only available if the target microprocessor puts out the necessary status information.

There are five breakpoint (BP) combinations and sixteen “Coast” combinations, making a total of eighty total possible breakpoint conditions.

- Program Trace

Integral ISE maintains a constant record, in real-time, of the last 256 cycles performed by the target microprocessor. Forty bits of information are recorded for each cycle:

—16 bits of address
—8 bits of data
—8 bits of CPU status
—8 bits of user-defined status, via the 8-bit status cable

The type of information recorded in the trace memory is selectable in thirteen ways:

—All write cycles only
—All read cycles only
—Instruction fetches only
—*Memory read cycles only
—*Memory write cycles only
—*Memory read or write cycles
—*I/O read cycles only
—*I/O write cycles only
—*I/O read or write cycles
—*Interrupt acknowledges
—*Serial input data only
—*Serial output data only
—All of the above

Note: Those options preceded by a * are only available if the target microprocessor puts out the necessary status information.

Integral ISE generates a Sync Pulse each time data is recorded in the trace memory.

- Target Board Control Features

  The target microprocessor will be placed in an inactive state at the end of the current instruction when one of the following conditions occurs:

  —The user gives a halt-command to the given target
  —A breakpoint is encountered
  —The Integral ISE is in single-step mode

  When a target is halted, the user may take any one or all of the following actions:

  —Examine and change the target's registers, memory, or port
  —Dump trace memory for examination
  —Change emulation specifications
  —Change memory map

- Flexible Memory Mapping

  A 32K mapped memory space is available for the Integral ISE. The applications program may be mapped into Integral ISE memory in 1K blocks. These blocks need not be contiguous. The memory map may be specified and altered under program control, and any segment may be write protected. In addition, data may be copied from the applications system memory to the Integral ISE memory.

- Microsecond Timer

  National Semiconductor's Integral ISE has a 16-second timer which counts in one-microsecond increments. The user may use this timer to measure the time elapsed between any two points of this program. The two points in the program must be defined through breakpoint conditions; the clock starts counting as breakpoint #1 is encountered and stops when breakpoint condition #2 is encountered.

- User Status Cable

  Integral ISE provides the user with a six-foot cable carrying eight probes. The user may hook these probes anywhere in his system and treat the status of these points as part of his breakpoint word and trace a word.

Convenient Software

Several tools are provided to make the Integral ISE a very convenient emulation system to use. Many of the debugging features available for software development, like symbolic debugging, are now available for system development.

- Symbolic Debugging

  Programmers use symbols to reference program and data memory when writing programs, but they are usually required to use absolute hexadecimal addresses when referencing those locations during program debug. Integral ISE allows the designer to use those same symbols to reference program and data memory during program debug. A symbol table is generated when the program is first assembled or compiled in the host development system. That symbol table is passed to the driver program in the host system for use during the debugging operations. During debugging operations, symbols may be added or deleted, and symbol values may be redefined.

- In-Line Assembler

  A one-pass line-by-line assembler is provided to allow modification of object code in the Integral ISE memory or the applications system memory without having to manually convert symbolic instructions to machine language. The in-line assembler accepts program modifications in the assembly language of the target microprocessor, assembles them, and inserts them into the object program at the locations specified by the system programmer.

- Disassembler

  The disassembler reads specified segments of Integral ISE or applications system memory, disassembles them, and displays their contents in the assembly language mnemonics of the target microprocessor. This feature eliminates many of the tedious manual steps normally involved in application debug.

- Automatic Testing

  The application system designer often wishes to perform a predefined sequence of tests on the system over a relatively long period of time. Integral ISE has an automatic testing mode whereby the designer may write a sequence of test steps in a language similar to BASIC, store those tests in the memory of the host system, and initiate the test sequence. Integral ISE will perform the tests in the specified sequence and, if requested, record the results on a disc or a hard copy device of the host system. Branching and conditional branching are also permitted in the test program. This feature is especially useful for rigorous proof that all parts of the applications system are in fact working, for detecting and documenting infrequent failures, and for performing "life" tests.

The list of predefined test sequences resides in a file created by using the Integral ISE software or the development system's Text Editor. Once the file is resident on disc, it can be retrieved, deleted, edited, etc., by the Integral ISE Software Package.

The following commands allow the user to perform automatic testing functions:

DELETE	Deletes a range of lines from the test program.
EXECUTE	Executes the test program.
LIST	Lists the test program.
LOAD FAST INFILE	Loads the specified test program from disc.
SAVE FAST INFILE	Saves the test program on disc.
SCRATCH	Deletes the entire test program.
END	Directive to end test program and return control to command mode.
GOTO	Unconditional branch to another statement in test program.
IF	Conditional branch to another statement in test program.
INPUT	Enables user to interact with test program at run time to specify data values.
PRINT	Prints number and string data on console.
ERROR	Allows errors to occur during testing without halting the test.
CALL	Passes control to a line in an INFILE subroutine.
RETURN	Returns control from an INFILE subroutine.
SET PARAMETER	Sets a specified INFILE parameter to a value.

Command Summary

Initialization and Setup Commands

CHANGE	Change target-specific system configuration characteristics.
HOLD	Enables or disables hold timeout for the selected target processor.
INITIALIZE	Causes a reset of the target board firmware and clears the work registers of the selected target processor.
LATCH	Selects trigger for input validity from the user status cable.
LOCK	Forces all target processors into "hold" state to allow power-down of user system.
RESET	Indicates that the selected target processor registers are to be reset prior to the resumption of emulation.
NORESET	Rescinds the RESET command.
RADIX	Establishes the default input and display mode (binary, octal, decimal of hexadecimal).
WAIT	Enables or disables wait timeout for the selected target processor.

Memory Mapping/Demapping Control

MAP	This command enables or disables use of ISE memory and allows copying betwen ISE and user memory.
GUARD	Write-protects any block of target I/O ports or memory.
UNGUARD	Write-enables any block of target I/O ports of memory.

Breakpoint Control

BREAK	Suspends emulation when the specified break conditions are met in the target system.
TIME	Displays the time interval between occurence of breakpoints A and B, when B occurs after A.

Emulation Commands

RUN	Continues the system emulation until a break condition is satisfied.
STEP	Continues the system emulation in single-step mode.

Trace Control

TRACE	Selects target activity to be recorded into the trace memory.

Memory/Register/Port Modification and Display Commands

CHANGE	Replaces contents of memory locations with new data values or writes values to I/O ports.
DISPLAY	Displays portions of target processor memory, register, I/O port, or trace data.
MOVE	Transfers a region of memory into another region.
SEARCH	Searches a range of memory locations for a specified value and displays the locations where the value is found.

Symbol Table and File Manipulation Commands

DELETE	Deletes the specified symbol(s) from the symbol table file.
LOAD	Fetches an in-file program or load file from disk medium or opens a symbol table file.

SAVE Creates and saves in-file programs or load files on disk or closes symbol table files.

LIST Display all or part of the INFILE.

Utility Commands

ECHO Selects the host processor's echoing device for hard copy history.

PRINT Displays strings and expressions.

ARCHIVE Saves the system status on disk for later retrieval.

RESTORE Restores the system status saved by ARCHIVE.

DIAGNOSTIC Performs limited testing of Trace and Memory Boards.

Specifications

Environmental

Operating Temperature 10°C (50°F) to 32°C (90°F)

Storage Temperature -40°C (-40°F) to 75°C (167°F)

Power

All boards +5 V_{DC} each

Board Involved	**Worst**
Target Board	4.2A (21.0W)
Cable Board	1.9A (9.5W)
Integral ISE Set:	
Trace Board	3.6A (18.0W)
Status Board	0.3A (1.5W)
Memory Board	5.5A (27.5W)
Total	15.5A (77.5W)

Memory

Mappable — 32K bytes in 1K byte increments

Trace — 256 × 40 bits

Physical

Target Board
Height — 7.15 in. (18.16 cm)
Width — 12.00 in. (30.48 cm)
Depth — 0.50 in. (1.27 cm)

Cable Board
Height — 9.00 in. (22.86 cm)
Width — 4.55 in. (11.56 cm)
Depth — 0.54 in. (1.37 cm)

Cable Board Housing
Height — 9.75 in. (24.76 cm)
Width — 5.75 in. (14.60 cm)
Depth — 1.62 in. (4.11 cm)

Status Board Housing
Height — 6.00 in. (15.24 cm)
Width — 3.00 in. (7.62 cm)
Depth — 0.50 in. (1.27 cm)

Memory Board
Height — 6.75 in. (17.75 cm)
Width — 12.00 in. (30.48 cm)
Depth — 0.50 in. (1.27

Trace Board
Height — 6.75 in. (17.75 cm)
Width — 12.00 in. (30.48 cm)
Depth — 0.50 in. (1.27 cm)

Cables
Target Board Cables — 72.0 in. (6 ft.)
Cable-Board/User Cable — 15.0 in. (1 ft. 3 in.)

Approximate overall length from STARPLEX base module to the 40-pin connector—96 in (8 ft.)

Prerequisites

Any STARPLEX/STARPLEX II Development System.

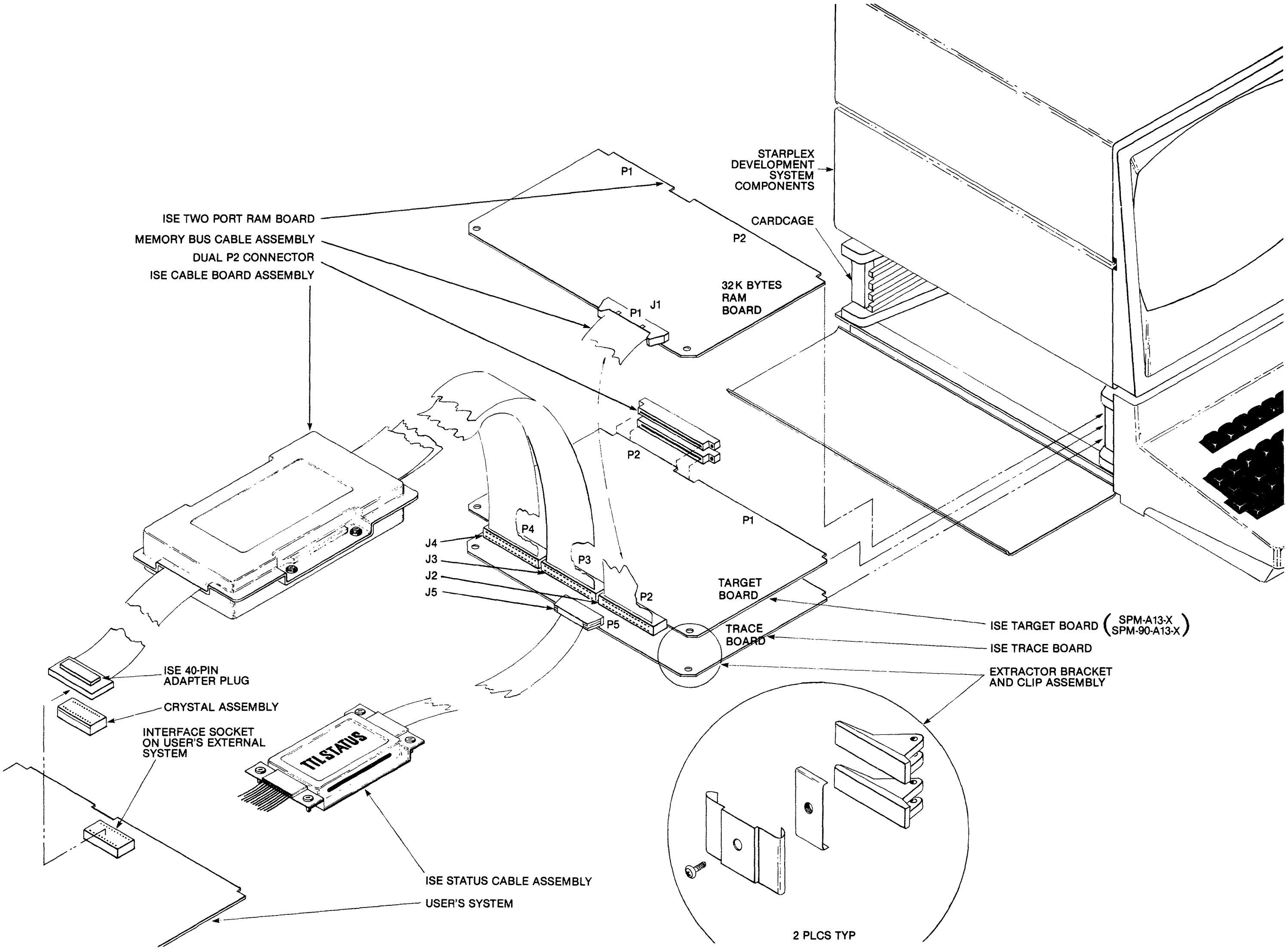


FIGURE 1. Integral ISE Components Installation

Order Information

(Includes One 32K Byte Mapped Memory Board, One Trace Board, Cables, and One ISE TTL Status Cable Pod.)

For STARPLEX Development Systems:

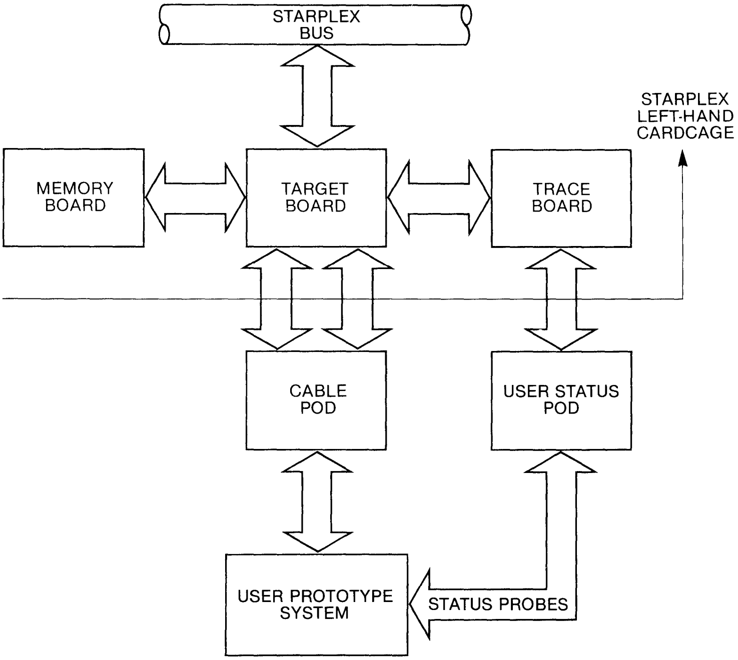
SPM-A13	Integral In-System Emulator Package
SPM-A13-2	80CX48 Emulator Package
SPM-A13-3	8085 Emulator Package
SPM-A13-4	NSC800 Emulator Package
SPM-A13-7	Z80 Emulator Package

For STARPLEX II Development Systems:

SPM-90-A13	Integral In-System Emulator Package
SPM-90-A13-2	80CX48 Emulator Package
SPM-90-A13-3	8085 Emulator Package
SPM-90-A13-4	NSC800 Emulator Package
SPM-90-A13-7	Z80 Emulator Package

Documentation

420305789-001	8080/8085 Macroassembler Software User's Manual
420306198-001	NSC800 Cross-Assembler Software User's Manual
420306240-001	8085 Integral ISE User's Manual
420306421-001	NSC800 Integral ISE User's Manual
420306692-001	Z80 Integral ISE User's Manual
420308101-001	80CX48 Integral ISE User's Manual



**Integral ISE System Configuration
SPM-A13 with SPM-A13-X,
SPM-90-A13 with SPM-90-A13-X
(Total: 3 Boards and 2 Pods)**

National Semiconductor

NS80CX48 Emulator Package

- **Real time emulation of NS80CX48 microprocessors family devices (NS80CX35, 80CX39, 80CX40, NS80CX48, 80CX49, 80CX50, 80C35, 80C39, 80C48, 80C49, 80C50)**
- **Supports two model of operation**
  - Program development
  - Single processor emulation
- **Plugs directly into any STARPLEX™/ STARPLEX II™ development system**
- **Contains NS80CX48 and 80C48 architectural firmware, status decode logic and CPU control circuitry**
- **4 Kbytes of on-board RAM to allow disassembly and debug of program ROM**

Product Overview

National's NS80CX48 Emulator Package gives the designer of NS80CX48 based systems the kind of sophisticated tool required for efficient microcomputer development. The NS80CX48 Emulator Package, in conjunction with the Trace Board in the Integral ISE Package™ and a STARPLEX/STARPLEX II Development System, provides capabilities that up to now have not been available in any NS80CX48 or 80C48 Emulator Package.

The Trace Board in National's Integral ISE Package is installed directly into any STARPLEX/STARPLEX II Development System. The Integral ISE Package consists of two logic boards—TRACE logic and 32 Kbyte of dedicated MAPPED MEMORY; performing NS80CX48 emulation requires the use of only the trace board from the Integral ISE Package. The trace logic board provides the user with all the necessary logic for breakpoints, tracing and memory mapping for NS80CX48 development work. In general, the Integral ISE™ provide the resources that are available for the emulation of any microprocessor since the individual emulation packages are the only package with the components dedicated to the particular microprocessors. This approach simplifies emulation since the user needn't learn a new ISE™ language each time he changes emulation packages.

The NS80CX48 Emulator Package provides the physical and electrical interface between the trace board, a STARPLEX/STARPLEX II development system and an NS80CX48 based system undergoing development. When installed in a STARPLEX/ STARPLEX II Development System, the ISE connects to the user's system under development via the 40-pin plug extending from the cable pod assembly. In this configuration, the entire ISE system supports two modes of operation. These modes are program development and single processor emulation.

The program development mode permits the user to develop and debug his software even though he may have no prototype hardware available. The Emulator Package provides the clocks and memory necessary for this task.

During single processor emulation, the user's system under development provides the actual clock signal thus forcing the entire NS80CX48 Integral ISE system to operate at the actual clock rate of the user's system.

Whether in program development or single processor emulation, NS80CX48 development work is done in real time. That's because no demand is made of the STARPLEX/STARPLEX II development system's resources.

Optional NS80CX48 Emulator Package-Complete

An optional NS80CX48 Emulator Package-Complete is available for the first time users who want to do real time emulation of the NS80CX48 microprocessor. Included in this package are: target board, lightweight plastic cable pod, cables, ISE Host Driver software, NS80CX48 Display Change software for mnemonic assembly and disassembly, 80CX48 Cross-Assembler Software, trace board and a CMOS status pod.

(**Note:** The first time user is recommended purchasing Integral ISE Package in conjunction with the NS80CX48 Emulator Package if there are considerations for future microprocessor development work such as the NSC800™ microprocessor.)

Specifications

Environmental

- Operating Temperature 0°C (32°F) to 40°C (104°F)
- Storage Temperature −40°C (7°F) to 75°C (167°F)

Power

All cards +5 VDC each

Card Involved:	Typical	Worst
NS80CX48 Target Card	3.6A (18.0W)	4.2A (21.0W)
NS80CX48 Cable Card	1.4A (7.0W)	1.9A (9.5W)
Trace Board	2.0A (10.0W)	3.6A (18.0W)
Status Card	0.3A (1.5W)	
Total:	Typical 36.5W	Worst 48.5W

Physical

- Target Card
  Height: 7.15 in. (18.16 cm)
  Width: 12.00 in. (30.48 cm)
  Depth: 0.50 in. (1.27 cm)
- Cable Card
  Height: 9.00 in. (22.86 cm)
  Width: 4.55 in. (11.56 cm)
  Depth: 0.54 in. (1.37 cm)
- Cable Card Housing
  Height: 9.75 in. (24.76 cm)
  Width: 5.75 in. (14.60 cm)
  Depth: 1.62 in. (4.11 cm)
- Status Card Housing
  Height: 6.00 in. (15.24 cm)
  Width: 3.00 in. (7.62 cm)
  Depth: 0.50 in. (1.27 cm)
- Trace Board
  Height: 6.75 in. (17.75 cm)
  Width: 12.00 in. (30.48 cm)
  Depth: 0.50 in. (1.27 cm)
- Cables
  Target Board Cables: 72.0 in. (6 ft)
  Cable Board/User Cable: 15.0 in. (1 ft 3 in)
  Approximate overall length from STARPLEX base module to the 40-pin connector: 96 in. (8 ft)

Prerequisites

Any STARPLEX/STARPLEX II Development System and Integral In-System Emulator Package (SPM-A13, SPM-90-A13).

Order Information

(Includes Target Board, Lightweight Plastic Cable Pod, Cables, ISE Host Driver Software, NS80CX48 Display Change Software for Mnemonic Assembly and Disassembly, NS80CX48 Cross Assembler Software.)

For use in STARPLEX Development Systems:

SPM-A13	Integral ISE Package
SPM-A13-2	NS80CX48 Emulator Package

For use in STARPLEX II Development Systems:

SPM-90-A13	Integral ISE Package
SPM-90-A13-2	NS80CX48 Emulator Package

For the first time user:
(**Note:** No pre-requisites other than a STARPLEX/STARPLEX II Development System.)

For use in STARPLEX Development Systems:

SPM-A25	NS80CX48 Integral In-System Emulator Package-Complete

For use in STARPLEX II Development Systems:

SPM-90-A25	NS80CX48 Integral In-System Emulator Package-Complete

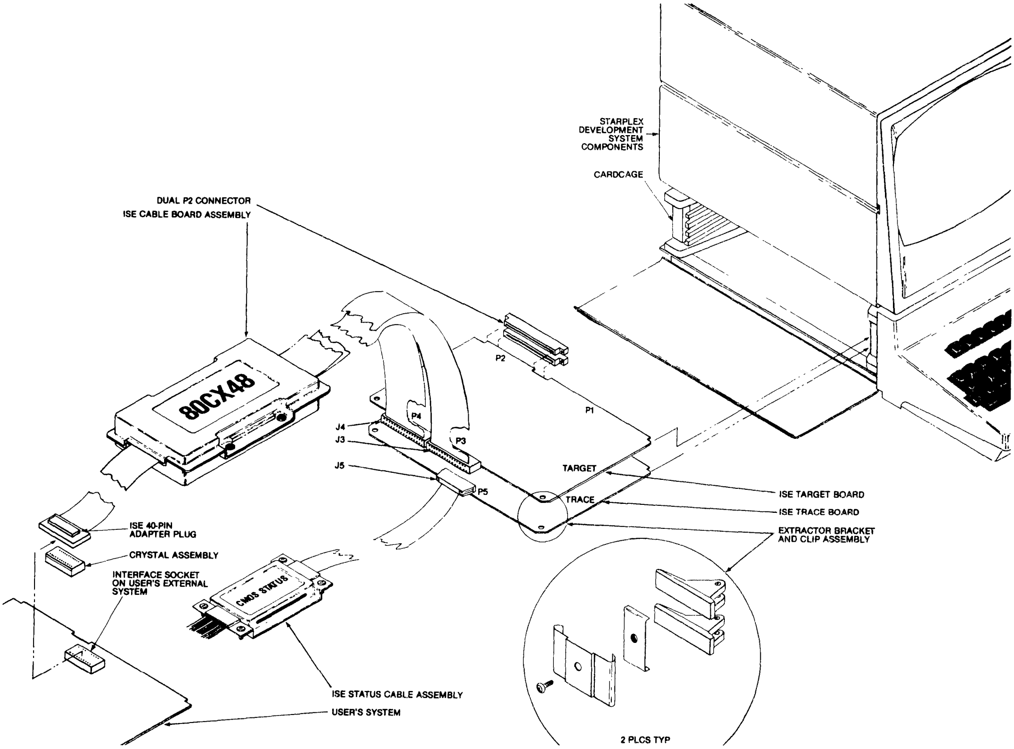
Documentation

420306064-001	NS8048 Cross Assembler Software User's Manual
420308101-001	Integral In-System Emulator (NS80CX48) User's Manual

ISE 80CX48 AC and DC Characteristics

Signals	Pin	V_{OH} = 2.4V I_{OH}(mA)	V_{OL} = 0.8V I_{OL}(mA)	V_{IH} = 2.0V I_{IH}(μA)	V_L = 0.8V I_L(μA)	Worst Case T_D(NS)
TO	1	−.01	2	11	−11	0
XTAL1	2					0
XTAL2	3					0
RESET/	4			1	−1	38
SS/	5			1	−1	20
INT/	6			1	−1	20
EA	7			1	−1	20
RD/	8	−4	4			20
PSEN/	9	−4	4			20
WR/	10	−4	4			20
ALE	11	−4	4			20
DB0	12	−2	2	1	−1	50
DB1	13	−2	2	1	−1	50
DB2	14	−2	2	1	−1	50
DB3	15	−2	2	1	−1	50
DB4	16	−2	2	1	−1	50
DB5	17	−2	2	1	−1	50
DB6	18	−2	2	1	−1	50
DB7	19	−2	2	1	−1	50
V_{SS}	20					
P20	21	−.01	2	11	−11	0
P21	22	−.01	2	11	−11	0
P22	23	−.01	2	11	−11	0
P23	24	−.01	2	11	−11	0
PROG/	25	−.01	2	11	−11	0
V_{dd} *	26					
P10	27	−.01	2	11	−11	0
P11	28	−.01	2	11	−11	0
P12	29	−.01	2	11	−11	0
P13	30	−.01	2	11	−11	0
P14	31	−.01	2	11	−11	0
P15	32	−.01	2	11	−11	0
P16	33	−.01	2	11	−11	0
P17	34	−.01	2	11	−11	0
P24	35	−.01	2	11	−11	0
P25	36	−.01	2	11	−11	0
P26	37	−.01	2	11	−11	0
P27	38	−.01	2	11	−11	0
T1	39			1	−1	100
V_{cc}/IDLE/	40					

*V_{dd} 5V ± 10%

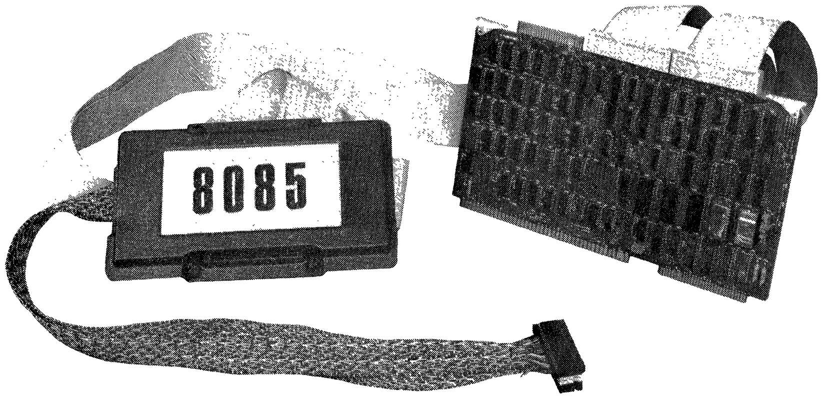

STARPLEX DEVELOPMENT SYSTEM COMPONENTS
CARDCAGE
DUAL P2 CONNECTOR ISE CABLE BOARD ASSEMBLY
80CX48
P2
P1
J4
J3
P4
P3
J5
P5
TARGET
TRACE
ISE TARGET BOARD
ISE TRACE BOARD
EXTRACTOR BRACKET AND CLIP ASSEMBLY
ISE 40-PIN ADAPTER PLUG
CRYSTAL ASSEMBLY
INTERFACE SOCKET ON USER'S EXTERNAL SYSTEM
CMOS STATUS
ISE STATUS CABLE ASSEMBLY
USER'S SYSTEM
2 PLCS TYP

TL/R/5273-1

Integral ISE 80CX48 Components Installation

National Semiconductor

8085 Emulator Package



- **Real Time In-System Emulation of 8085 Microprocessor Devices (8085A and 8085A-2)**
- **Supports Both 8085A and 8085A-2 Applications Without Alterations**
- **Supports Two Modes of Operation**
  - Program Development
  - Single Processor Emulation
- **Plugs directly into any STARPLEX™/ STARPLEX II™ Development System**

Product Overview

National's 8085 Emulator Package gives the designer of 8085 based system the kind of sophisticated tool required for efficient microcomputer development. The 8085 Emulator Package, in conjunction with the Integral ISE™ Package and a STARPLEX/STARPLEX II Development System, provides capabilities that up to now have not been available in this type of instrument.

National's Integral ISE Package is installed directly in any STARPLEX/STARPLEX II Development System. This package consists of two logic boards (TRACE logic and MAPPED MEMORY). These two logic boards provide the user with 32 K bytes tracing and memory mapping. These resources are available for the emulation of any processor since the individual emulation packages are the only components dedicated to particular processors. This approach simplifies changing processors since the user needn't learn a new ISE™ language each time he changes emulation packages.

The 8085 Emulator Package provides the physical and electrical interface between the Integral ISE package, the STARPLEX Development System and an 8085 based system undergoing development. When installed in a STARPLEX Development System, it connects to the User's System via the Cable Pod and a 40-pin plug to the system under development. In this configuration, the entire system supports two modes of operation. These modes are program development and single processor emulation.

The program development mode permits the user to develop and debug his software even though he has no prototype hardware available. The emulator package provides the clocks and memory necessary for this task. During emulation of a single processor, the user's hardware provides the actual clock signal thus forcing the entire Integral ISE system to operate at the actual clock rate of the user's system.

Characteristics for 40-Pin Connector

	Output Load		Input Load			
Pin No.	$V_{OH} = 2.4V$ I_{OH} (mA)	$V_{OL} = 0.8V$ I_{OL} (mA)	$V_{IH} = 2.0V$ I_{IH} (μA)	$V_{IL} = 0.8V$ I_{IL} (μA)	Time Delay (T_D) Between 8085 & 40 Pin Plug	Mnemonic
1	—	—	—	—	0	X_1
2	—	—	—	—	0	X_2
3	−.36	1.6	—	—	22 ns	RESET OUT
4	−.35	1.6	—	—	0	SOD
5	—	—	60.0	−410	0	SID
6	—	—	30.0	−210	38 ns	TRAP
7	—	—	30.0	−210	38 ns	RST 7.5
8	—	—	30.0	−210	38 ns	RST 6.5
9	—	—	30.0	−210	38 ns	RST 5.5
10	—	—	30.0	−210	38 ns	INTR
11	−.36	1.6	—	—	0	INTA
12	−10.0	48	80	−250	27 ns	AD_0
13	−10.0	48	80	−250	27 ns	AD_1
14	−10.0	48	80	−250	27 ns	AD_2
15	−10.0	48	80	−250	27 ns	AD_3
16	−10.0	48	80	−250	27 ns	AD_4
17	−10.0	48	80	−250	27 ns	AD_5
18	−10.0	48	80	−250	27 ns	AD_6
19	−10.0	48	80	−250	27 ns	AD_7
20	—	—	—	—	—	V_{SS}
21	−.35	1.6	—	—	0	A_8
22	−.35	1.6	—	—	0	A_9
23	−.35	1.6	—	—	0	A_{10}
24	−.35	1.6	—	—	0	A_{11}
25	−.35	1.6	—	—	0	A_{12}
26	−.35	1.6	—	—	0	A_{13}
27	−.35	1.6	—	—	0	A_{14}
28	−.35	1.6	—	—	0	A_{15}
29	−.35	1.6	—	—	0	S_D
30	−.35	1.6	—	—	0	ALE
31	−15.0	64	—	—	9 ns	$\overline{WR}$
32	−15.0	64	—	—	9 ns	$\overline{RD}$
33	−.35	1.6	—	—	0	S_1
34	−.35	1.6	—	—	0	$IO/\overline{M}$
35	—	—	—	—	30 ns	READY
36	—	—	60	−410	38 ns	READY
37	−.35	1.6	—	—	0	$\overline{RESET\ IN}$
38	−.38	1.8	—	—	0	HLDA
39	—	—	30	−210	43	HOLD
40	*	*	*	*	—	V_{CC}

*V_{CC}@ +5V I_{CC} = 1mA maximum

Specifications

Environmental

Operating Temperature 10°C (50°F) to 32°C (90°F)

Storage Temperature −40°C (−40°F) to 75°C (167°F)

Power

All boards +5VDC each

Board Involved	Typical	Worst
8085 Target Board	3.6A (18.0W)	4.2A (21.0W)
8085 Cable Board	1.4A (7.0W)	1.9A (9.5W)
Integral ISE Set		
Trace Board	2.0A (10.0W)	3.6A (18.0W)
Status Board	0.3A (1.5W)	
Memory Board	2.7A (13.5W)	5.5A (27.5W)
Total:	50.0W	77.0W

Physical

Target Board
Height — 7.15 in. (18.16 cm.)
Width — 12.00 in. (30.48 cm.)
Depth — 0.50 in. (1.27 cm.)

Cable Board
Height — 9.00 in. (22.86 cm.)
Width — 4.55 in. (11.56 cm.)
Depth — 0.54 in. (1.37 cm.)

Cable Board Housing
Height — 9.75 in. (24.76 cm.)
Width — 5.75 in. (14.60 cm.)
Depth — 1.62 in. (4.11 cm.)

Status Board Housing
Height — 6.00 in. (15.24 cm.)
Width — 3.00 in. (7.62 cm.)
Depth — 0.50 in. (1.27 cm.)

Memory Board
Height — 6.75 in. (17.75 cm.)
Width — 12.00 in. (30.48 cm.)
Depth — 0.50 in. (1.27 cm.)

Trace Board
Height — 6.75 in. (17.75 cm.)
Width — 12.00 in. (30.48 cm.)
Depth — 0.50 in. (1.27 cm.)

Cables
Target Board Cables — 72.0 in (6 ft.)
Cable-Board/User Cables — 15.0 in (1 ft. 3 in.)
Approximate overall length from STARPLEX base module to the 40-pin connector — 96 in. (8 ft.)

Prerequisites

Any STARPLEX /STARPLEX II Development System and Integral In-System Emulator Package (SPM-A13, SPM-90-A13)

Order Information

(Includes Target Board, Lightweight Plastic Cable Pod, Cables, Software for ISE Host Driver, and 8085 Display Change for Mnemonic Assembly and Disassembly. SPM-90-A13-3 also includes 8080/8085 Cross-Assembler Software.)

For STARPLEX Development Systems:

SPM-A13	Integral In-System Emulator Package
SPM-A13-3	8085 Emulator Package

For STARPLEX II Development Systems:

SPM-90-A13	Integral In-System Emulator Package
SPM-90-A13-3	8085 Emulator Package

Documentation

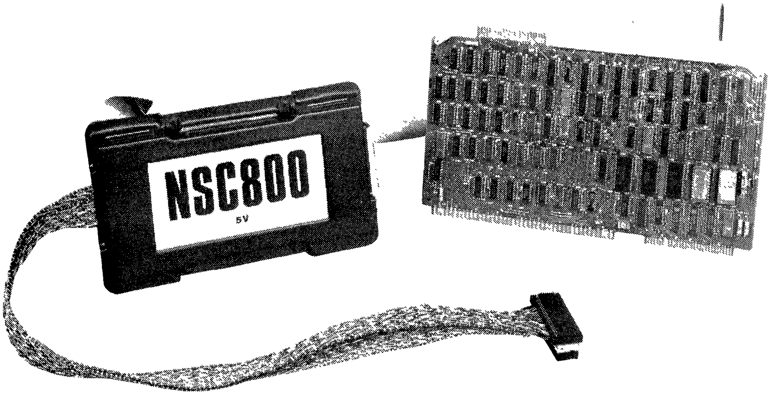
420305789-001	STARPLEX 8080/8085 Assembler Software User's Manual[1]
420306240-001	Integral In-System Emulator (8085) User's Manual[2]

[1] Included with SPM-90-A13-3 not SPM-A13-3

[2] Included with both SPM-90-A13-3 and SPM-A13-3

National Semiconductor

NSC800™ Emulator Package



- **Real-Time In-System Emulation of NSC800 Microprocessor**
- **Supports Two Modes of Operation**
  - Program Development
  - Single Processor Emulation
- **Plugs Directly Into Any STARPLEX™/ STARPLEX II™ Development System**

Product Overview

National's NSC800 Emulator Package gives the designer of NSC800 based systems the kind of sophisticated tool required for efficient microcomputer development. The NSC800 Emulator Package, in conjunction with the Integral ISE™ Package and the STARPLEX/STARPLEX II Development System, provides capabilities that up to now have not been available in this type of instrument.

National's Integral ISE Package is installed directly in any STARPLEX/STARPLEX II Development System. This package consists of two logic boards (TRACE logic and MAPPED MEMORY). These two logic boards provide the user with 32K bytes tracing and memory mapping. These resources are available for the emulation of any processor since the individual emulation packages are the only components dedicated to particular processors. This approach simplifies changing processors since the user needn't learn a new ISE™ language each time he changes emulation packages.

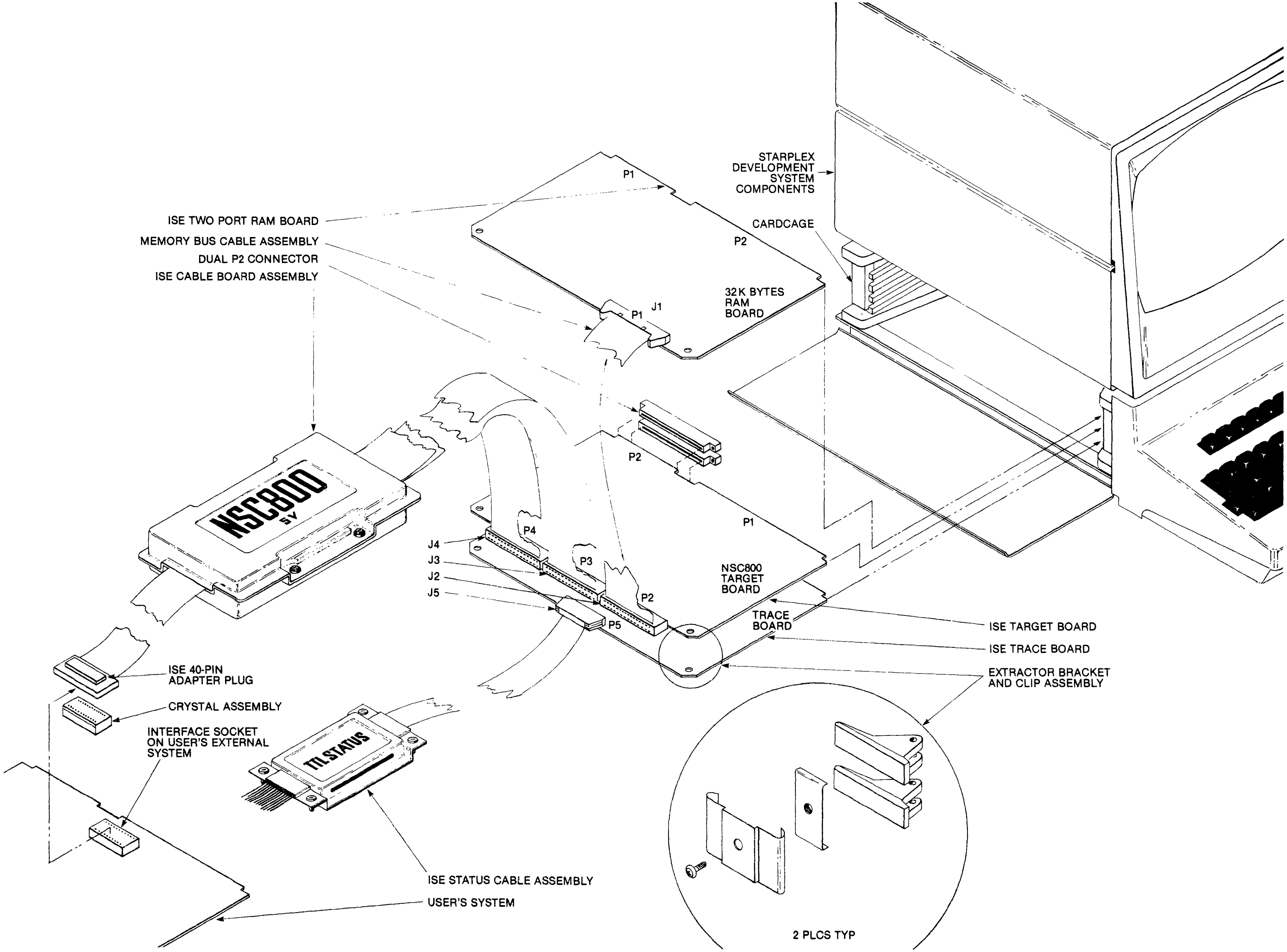
The NSC800 Emulator Package provides the physical and electrical interface between the Integral ISE package, the STARPLEX Development System and an NSC800 based system undergoing development. When installed in a STARPLEX Development System, it connects to the User's System via the Cable Pod and a 40-pin plug to the system under development. In this configuration, the entire system supports two modes of operation. These modes are program development and single processor emulation.

The program development mode permits the user to develop and debug his software even though he has no prototype hardware available. The emulator package provides the clocks and memory necessary for this task. During emulation of a single processor, the user's hardware provides the actual clock signal thus forcing the entire Integral ISE system to operate at the actual clock rate of the user's system.

Loading Information for 40-Pin Connector

Pin No.	Output Load (mA)		Input Load (μA)		Time Delay (T_D) Between NSC800 & 40 Pin Plug	Mnemonic
	$V_{OH} = I_{OH}$	$V_{OL} = I_{OL}$	$V_{IH} = I_{IH}$	$V_{IL} = I_{IL}$		
1	−.95	1.6	—	—	0	A_8
2	−.95	1.6	—	—	0	A_9
3	−.95	1.6	—	—	0	A_{10}
4	−.95	1.6	—	—	0	A_{11}
5	−.95	1.6	—	—	0	A_{12}
6	−.95	1.6	—	—	0	A_{13}
7	−.95	1.6	—	—	0	A_{14}
8	−.95	1.6	—	—	0	A_{15}
9	−.95	1.6	—	—	0	CLK
10	—	—	—	—	—	X_{OUT}
11	—	—	—	—	0	X_{IN}
12	−10	48	80	−250	27 ns	AD_0
13	−10	48	80	−250	27 ns	AD_1
14	−10	48	80	−250	27 ns	AD_2
15	−10	48	80	−250	27 ns	AD_3
16	−10	48	80	−250	27 ns	AD_4
17	−10	48	80	−250	27 ns	AD_5
18	−10	48	80	−250	27 ns	AD_6
19	−10	48	80	−250	27 ns	AD_7
20	—	—	—	—	—	GND
21	—	—	20	−200	38 ns	$\overline{NMI}$
22	—	—	20	−200	38 ns	$\overline{RSTA}$
23	—	—	20	−200	38 ns	$\overline{RSTB}$
24	—	—	20	−200	38 ns	$\overline{RSTC}$
25	—	—	20	−200	38 ns	$\overline{INTR}$
26	−.96	1.6	—	—	0	$\overline{INTA}$
27	−.96	1.6	—	—	0	S_1
28	−.96	1.6	—	—	0	$\overline{RFSH}$
29	−.95	1.6	—	—	0	S_O
30	−.95	1.6	—	—	0	ALE
31	−15	64	—	—	9 ns	$\overline{WR}$
32	−15	64	—	—	9 ns	$\overline{RD}$
33	—	—	50	−400	38 ns	$\overline{RESET\ IN}$
34	−.95	1.6	—	—	0	$IO/\overline{M}$
35	−.98	1.8	—	—	0	$\overline{BACK}$
36	—	—	20	−200	43 ns	$\overline{BREQ}$
37	−.96	1.6	—	—	22 ns	RESET OUT
38	—	—	50	−400	30 ns	WAIT
39	—	—	20	−200	0	$\overline{PS}$
40	*	*	*	*	—	V_{CC}

* V_{CC}@ +5 V I_{CC} = 1 mA max.



NSC800 Emulator with Integral ISE Components Installation

Specifications

Environmental

Operating Temperature 10°C (50°F) to 32°C (90°F)

Storage Temperature −40°C to −40°F

Power

All Boards +5V_{DC} each

Board Involved	Typical	Worst
NSC800 Target Board	3.6A (18.0W)	4.2A (21.0W)
NSC800 Cable Board	1.4A (7.0W)	1.9A (9.5W)
Integral ISE Set		
Trace Board	2.0A (10.0W)	3.6A (18.0W)
Status Board	0.3A (1.5W)	
Memory Board	2.7A (13.5W)	5.5A (27.5W)
Total:	50.0 W	77.0 W

Physical

Target Board
Height — 7.15 in. (18.16 cm.)
Width — 12.00 in. (30.48 cm.)
Depth — 0.50 in. (1.27 cm.)

Cable Board
Height — 9.00 in. (22.86 cm.)
Width — 4.55 in. (11.56 cm.)
Depth — 0.54 in. (1.37 cm.)

Cable Board Housing
Height — 9.75 in. (24.76 cm.)
Width — 5.75 in. (14.60 cm.)
Depth — 1.62 in. (4.11 cm.)

Status Board Housing
Height — 6.00 in. (15.24 cm.)
Width — 3.00 in. (7.62 cm.)
Depth — 0.50 in. (1.27 cm.)

Memory Board
Height — 6.75 in. (17.75 cm.)
Width — 12.00 in. (30.48 cm.)
Depth — 0.50 in. (1.27 cm.)

Trace Board
Height — 6.75 in. (17.75 cm.)
Width — 12.00 in. (30.48 cm.)
Depth — 0.50 in. (1.27 cm.)

Cables
Target Board Cables — 72.0 in (6 ft.)
Cable-Board/User Cables — 15.0 in (1 ft. 3 in.)
Approximate overall length from STARPLEX base module to the 40-pin connector — 96 in. (8 ft.)

Prerequisites

Any STARPLEX/STARPLEX II Development System and Integral In-System Emulator Package (SPM-A13, SPM-90-A13)

Order Information

(Includes Target Board, Lightweight Plastic Cable Pod, Cables, Software for ISE Host Driver and NSC800 Display Charge for Mnemonic Assembly and Disassembly. SPM-A13-4 also includes NSC800 Cross-Assembler Software.)

For STARPLEX Development Systems:

SPM-A13	Integral In-System Emulator Package
SPM-A13-4	NSC800 (5V) Emulator Package

For STARPLEX II Development Systems:

SPM-90-A13	Integral In-System Emulator Package
SPM-90-A13-4	NSC800 Emulator Package

Documentation

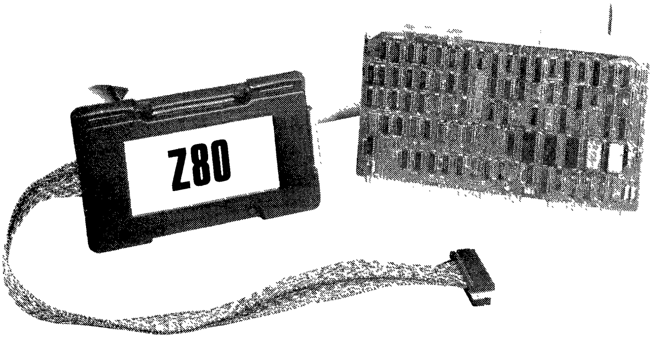
420306198-001	NSC800 Assembler Software User's Manual[1]
420306241-001	Integral In-System Emulator (NSC800) User's Manual[2]

[1] Included with SPM-A13-4 not SPM-90-A13-4.

[2] Included with both SPM-A13-4 and SPM-90-A13-4.

National Semiconductor

Z80 Emulator Package



- **Real-time emulation of Z80, Z80A and Z80B microprocessors**
- **Supports two modes of operation**
  - Program development
  - Single processor emulation
- **Plugs directly into any STARPLEX™/ STARPLEX II™ development system**
- **Includes target board, cable pod with cables and complete software**

Product Overview

National's Z80 Emulator Package gives the designer of Z80 based systems the kind of sophisticated tool required for efficient microcomputer development. The Z80 Emulator Package, in conjunction with the Integral ISE™ Package and a STARPLEX/STARPLEX II Development System, provides capabilities that up to now have not been available in this type of instrument.

National's Integral ISE Package is installed directly into any STARPLEX/STARPLEX II Development System. This package consists of two logic boards (TRACE logic and MAPPED MEMORY). These two logic boards provide the user with 32 K bytes of real-time map memory and all the necessary logic for break-points, tracing and memory mapping. These resources are available for the emulation of any processor since the individual emulation packages are the only components dedicated to particular processors. This approach simplifies changing processors since the user needn't learn a new ISE™ language each time he changes emulation packages.

The Z80 Emulator Package provides the physical and electrical interface between the Integral ISE package, the STARPLEX Development System and a Z80 based system undergoing development. When installed in a STARPLEX Development System, it connects to the user's system via the cable pod and a 40-pin plug to the system under development. In this configuration, the entire system supports two modes of operation. These modes are program development and single processor emulation.

The program development mode permits the user to develop and debug his software even though he has no prototype hardware available. The emulator package provides the clocks and memory necessary for this task. During emulation of a single processor, the user's hardware provides the actual clock signal, thus forcing the entire Integral ISE system to operate at the actual clock rate of the user's system.

Z80 (6 MHz) Emulator Package Option

An optional 6 MHz Z80 Emulator Package is available for first time users who want to do real-time emulation of the Z80B microprocessor. Included in this package are: target board, lightweight plastic cable pod, cables, software for ISE Host Driver, Z80 Display Change software of mnemonic assembly and disassembly, trace board, high-speed (55 ns) 32 K bytes mappable memory board and TTL status pod. (SPM-A20 also includes Z80 (NSC800™) Cross-Assembler Software.)

Characteristics for 40-Pin Connector

Pin No.	Output Load		Input Load		Time Delay (T_D) Between Z80 & 40-Pin Plug	Mnemonic
	$V_{OH} = 2.4\,V$ I_{OH}(mA)	$V_{OL} = 0.4\,V$ I_{OL}(mA)	$V_{IH} = 2.0\,V$ I_{IH}(μA)	$V_{IL} = 0.8\,V$ I_{IL}(μA)		
1	−0.20	1.4	—	—	0	A11
2	−0.20	1.4	—	—	0	A12
3	−0.20	1.4	—	—	0	A13
4	−0.20	1.4	—	—	0	A14
5	−0.20	1.4	—	—	0	A15
6	—	—	—	—	0	CLK
7	−10	20	80	−200	18	D4
8	−10	20	80	−200	18	D5
10	−10	20	80	−200	18	D6
11	—	—	—	—	—	V_{CC}
12	−10	20	80	−200	18	D2
13	−10	20	80	−200	18	D7
14	−10	20	80	−200	18	D0
15	−10	20	80	−200	18	D1
16	—	—	50	−400	9	$\overline{\text{INT}}$
17	—	—	50	−400	16	$\overline{\text{NMI}}$
18	−15	64	—	—	18	$\overline{\text{HALT}}$
19	−15	64	—	—	18	$\overline{\text{MREQ}}$
20	−15	64	—	—	18	$\overline{\text{IORQ}}$
21	−15	64	—	—	18	$\overline{\text{RD}}$
22	−15	64	—	—	18	$\overline{\text{WR}}$
23	−15	64	—	—	18	$\overline{\text{BUSAK}}$
24	—	—	100	−800	9	$\overline{\text{WAIT}}$
25	—	—	100	−800	24	$\overline{\text{BUSRQ}}$
26	—	—	100	−800	36	$\overline{\text{RESET}}$
27	−15	64	—	—	18	$\overline{\text{M1}}$
28	−0.20	1.4	—	—	0	$\overline{\text{RFSH}}$
29	—	—	—	—	—	GND
30	−0.20	1.4	—	—	0	A0
31	−0.20	1.4	—	—	0	A1
32	−0.20	1.4	—	—	0	A2
33	−0.20	1.4	—	—	0	A3
34	−0.20	1.4	—	—	0	A4
35	−0.20	1.4	—	—	0	A5
36	−0.20	1.4	—	—	0	A6
37	−0.20	1.4	—	—	0	A7
38	−0.20	1.4	—	—	0	A8
39	−0.20	1.4	—	—	0	A9
40	−0.20	1.4	—	—	0	A10

Specifications

Environmental

Operating Temperature —	10°C (50°F) to 32°C (90°F)
Storage Temperature —	−40°C (70°F) to 75°C (167°F)

Power

All Boards $+5V_{DC}$ each

Board Involved	Typical		Worst	
Z80® Target Board	3.6 Amps (18 Watts)		4.2 Amps (21 Watts)	
Z80 Cable Board	1.4 Amps (7.0 Watts)		1.9 Amps (9.5 Watts)	
Integral ISE Set:				
Trace Board	2.0 Amps (10 Watts)		3.6 Amps (18 Watts)	
Status Board	0.3 Amps (1.5 Watts)			
Memory Board	2.7 Amps (13.5 Watts)		5.5 Amps (27.5 Watts)	
Total:	Typical	50 Watts	Worst	77 Watts

Physical

Target Board —	Height	7.15 in.	(18.16 cm)
	Width	12.00 in.	(30.48 cm)
	Depth	0.50 in.	(1.27 cm)
Cable Board —	Height	9.00 in.	(22.86 cm)
	Width	4.55 in.	(11.56 cm)
	Depth	0.54 in.	(1.37 cm)
Cable Board Housing —	Height	9.75 in.	(24.76 cm)
	Width	5.75 in.	(14.60 cm)
	Depth	1.62 in.	(4.11 cm)
Status Board Housing —	Height	6.00 in.	(15.24 cm)
	Width	3.00 in.	(7.62 cm)
	Depth	0.50 in.	(1.27 cm)
Memory Board —	Height	6.75 in.	(17.75 cm)
	Width	12.00 in.	(30.48 cm)
	Depth	0.50 in.	(1.27 cm)
Trace Board —	Height	6.75 in.	(17.75 cm)
	Width	12.00 in.	(30.48 cm)
	Depth	0.50 in.	(1.27 cm)

Cables — Target Board Cables 72.0 in. (6 ft)
Cable-Board/User Cable 15.0 in. (1 ft. 3 in.)
Approximate overall length from STARPLEX base module to the 40-pin connector: 96 in.(8 ft.)

Prerequisites

Any STARPLEX/STARPLEX II Development System and Integral In-System Emulator Package (SPM-A13, SPM-90-A13).

Order Information

(Includes Target Board, Lightweight Plastic Cable Pod, Cables, Software for ISE Host Driver, Z80 Display Change for Mnemonic Assembly and Disassembly. SPM-A13-7 and SPM-A20 also include Z80 (NSC800) Cross-Assembler Software.)

For STARPLEX Development System:

SPM-A13	Integral ISE Package
SPM-A13-7	Z80 (4 MHz) Emulator Package

For STARPLEX II Development System:

SPM-90-A13	Integral ISE Package
SPM-90-A13-7	Z80 (4 MHz) Emulator Package

For complete 6 MHz development work:

Note: No prerequisites other than a STARPLEX/STARPLEX II Development System.

For STARPLEX Development System:

SPM-A20	Z80 (6 MHz) Integral In-System Emulator Package Complete

For STARPLEX II Development System:

SPM-90-A20	Z80 (6 MHz) Integral In-System Emulator Package Complete

Documentation

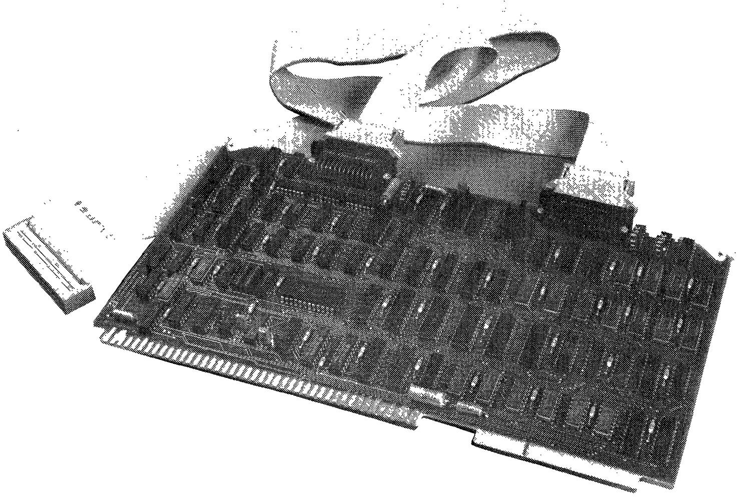
420306198-001	STARPLEX NSC800 Cross-Assembler Software User's Manual[1]
420306692-001	Integral In-System Emulator (Z80) User's Manual[2]

[1] Included with SPM-A13-7 and SPM-A20.

[2] Included with SPM-A13-7, SPM-90-A13-7, SPM-A20 and SPM-90-A20.

National Semiconductor

COPS™ In-System Emulator (ISE™) Package



- **True Real-Time Emulation of the COP400 Family of Microcontrollers**
- **Plugs Directly into Any STARPLEX™/STARPLEX II™ Development System**
- **Compatible with the Required Optional COP400 Family Emulator Boards**
- **Easy to Use**
  - Hardware
    - Real-time trace of 256 × 20-bit instruction cycles
    - 4K × 8-bit of Shared RAM Memory for rapid downloading of programs from STARPLEX/STARPLEX II peripherals
    - 1K × 12-bit dump memory used in place of control firmware
    - External hardware breakpoint
    - Breakpoint timer in milliseconds
    - Fully compatible with a STARPLEX/STARPLEX II system bus
    - One target card handles entire series of microcontrollers and COP400 Emulator Boards
  - Software
    - Software breakpoints
    - Lists user-specified registers when selected breakpoint is detected
    - Mnemonic modification of object code
    - Step-list-restart command
    - Dump routines for various COPS microcontroller chips

Product Overview

The COP400 In-System Emulator (ISE) is designed for users with the STARPLEX/STARPLEX II Development System. Coupled with the power of STARPLEX/STARPLEX II, COP400 ISE is a very powerful tool available for developing and debugging COP400 family based microcontroller products. The COP400 ISE target board plugs directly into any STARPLEX/STARPLEX II Development System and interfaces easily with any COP400 system. The designer has the capability of executing the target system program in real-time while collecting up to 256 instruction cycles of true real-time trace data. In addition, he can single step through his program and display the data from a 4K Shared Memory location.

Functional Description

Hardware

The COP400 ISE hardware consists of a printed circuit board (Target Board) which resides in the STARPLEX/ STARPLEX II chassis. This target board interfaces via a flat ribbon cable to a required external emulator board which interfaces to the user's prototype system. This interface from the emulator board to the user's prototype system is accomplished through a COP400 pin-compatible plug — e.g., 20, 24 or 28 pin pin-compatible plugs, depending on the microcontroller chip. With the external COP400-E02, COP400-E02C, COP400-E04L, COP400-E24 or any other COP400 Emulator Board, a designer can perform emulation of the entire COP400 family of microcontrollers. They include:

- COP420, COP420L, COP420C
- COP421, COP421L, COP421C
- COP444L
- COP445L
- COP410L, COP410C
- COP411L, COP411C
- COP440, COP441, COP442
- COP2440, COP2441, COP2442

Note: "C" and "L" denote CMOS and Low-power versions respectively.

The COP400 ISE target board has 4K × 8-bit of Shared RAM Memory to allow rapid downloading of programs from STARPLEX/STARPLEX II peripherals. Also implemented on the target board is a single hardware breakpoint to allow the user to halt execution of the user program at a specified point in order to obtain information on the internal state of the COPS microcontroller device under emulation before resuming execution.

Also, on the target board is a 1K × 12-bit dump memory, used in place of a control firmware. The purpose of this dump memory is to allow different dump routines, contained on the main host driver diskette, to be entered in the dump memory for different microcontroller chips. Thus, the target board can be used for the entire series of microcontrollers.

On the Emulator Boards are two features that facilitate tracing. They are: 1) a "Trace Out" test point to help trigger oscilloscopes and logic analyzers, and 2) four user defined "external event" inputs into the Trace Logic circuitry to allow the user to define his own "events" for tracing.

Software

The COP400 ISE software is a STARPLEX/STARPLEX II systems program which performs as the interface between the STARPLEX/STARPLEX II user and the COPS hardware system. The host driver, called COPMON™, allows the user the interrupt the flow of a program as it is being executed. The interruption is directly controlled by one of several events, all under user control. This interruption is called a "breakpoint." Possible conditions for a breakpoint are "address," "next instruction," or any combination of two external events. COPMON can maintain a ten (10) level "condition" stack to aid easy debugging of large programs. In addition, COPMON can be specified to "break" only on the nth occurrence of a particular condition. A breakpoint timer allows COPMON to display the time in milliseconds between two "conditions."

COPMON also has one other primary function, "trace" control and display. The trace command allows the user to specify: 1) conditions that will initiate the trace and 2) how many steps prior to meeting that condition will be traced. The "**G**o" (see Command Summary below) command then arms the trace logic and executes the user's program. After a trace has been completed, the user may wish to examine the trace data by using a "**TY**pe" command or the user may wish to search for an address in the trace memory by using a "**SE**arch" command.

The COPS Cross-Assembler is also included with the COPS STARPLEX System Software package. It assembles COPS programs written with the STARPLEX Editor and stores them as object code load modules on the system diskette. There the load modules are accessible to the COPMON program which loads them into the Shared Memory on the COP400 ISE target board and executes them through the Emulator Board.

The third program included with the software package is called MASKTR™. MASKTR accepts final object code load modules prepared by the cross-assembler as input files and translates them into "Transmittal Files" which are stored on another diskette. The Transmittal Files each are in a format acceptable for National Semiconductor to prepare "hard" mask patterns from for custom ROM-based COP400 chip programs. A Transmittal File contains:

1. Name and phone number of the customer
2. Company name and address
3. Date
4. Chip number
5. Listing of option showing option number, option name, and option value
6. ROM data including addresses
7. Source, object, and transmittal file checksums.

COPMON Console Command Summary

ALter	Alter Shared Memory
AUtoprint	Breakpoint printout control
Breakpoint	Breakpoint condition/occurrence control
Clear	Clears Breakpoint and Trace enables, and disables Timer
CHip	Selects Chip under emulation
COmpare	Compares Shared Memory to a disk file
Deposit	Deposit value into Shared Memory
Find	Searches Shared Memory for a specified value

END	Exit COPMON
Go	Begin Program Execution
Help	Prints out complete COPMON command summary for quick reference
List	Prints out the contents of Shared Memory
LOad	Loads Shared Memory from a disk file
Modify	Alters register contents of COPS chip under emulation
Next	Executes a single instruction but skips subroutines
Put	Alters Shared Memory mnemonically
Reset	Resets the COPS device
Singlestep	Executes a single instruction
SAve	Saves Shared Memory in a disk file
SEarch	Searches Trace memory for a specified address
SET	Set SIOMODE or STACKMODE Flags
SHared Mem	Enables Shared Memory operation
STatus	Prints out the emulation status
TIme	Breakpoint timer control
TYpe	Prints out register contents of COPS chip under emulation
TRace	Set Trace Conditions
UNassemble	Display Instruction Mnemonics of the data in Shared Memory

MASKTR Console Command Summary

Abort	Aborts the creation of a Transmittal File
COmpany	Prompts for Company Name and Address
Date	Prompts for Date
Error	Summarizes any option conflicts
Finish	Finishes the creation of the Transmittal File
List	Lists the Transmittal File
Name	Prompts for name/phone number of the person responsible for the program
Option	Prompts for the valid options
Print	Prints allowable options for chip specified
Transmittal	Load "Load Module" into memory

Prerequisites

Any STARPLEX/STARPLEX II Development System and a COP400-E02, COP400-E04L, COP400-E02C, COP400-E24 or any other COP400 Emulator Board.

Specifications

Note: The following specifications apply when the COPS ISE is configured with a standard COP400-E04L Emulator Board.

Environmental

Operating Temperature: 0 to 40°C
Storage Temperature: – 40°C to 80°C
Humidity: 10% to 90% non-condensing
Shock (Drop): 30g on 3 axis in shipping container

Power

(DC Characteristics — SPM-A15/SPM-90-A15 Power Consumption for Multiple Configurations)

Target Board: $+5V_{DC}$
Emulator Board: $+5V_{DC}$; $-12V_{DC}$

	Typical	Reasonable Worst Case
Target with Emulator and no PROMS:	2.25A	3.20A
Target with Emulator with PROMS:	2.50A	3.75A
Maximum User Supplied V_{CC}:	150mA	250mA

Physical

Target Board
Length: 6.75 inches
Width: 12.00 inches
Depth: 0.50 inches

Emulator Board
Length: 8.50 inches
Width: 4.55 inches
Depth: 1.00 inches (with 4-0.50 inch nylon standoffs)

Cables

Target/Emulator
Length: 3 feet
Material: 50 × 28 AWG flat ribbon
Termination: 50-pin PCB edge
RS232 male
RS232 female

Emulator User
Length: Approx. 1 foot
Material: 20, 24 or 28 × 28 AWG flat ribbon
Termination: 20-pin IC plug both ends
24-pin IC plug both ends
28-pin IC plug both ends

COPS Emulator/User Interface

COP411L – 20-Pin

Pin	Signal
1	L4
2	VCC
3	L3
4	L2
5	L1
6	L0
7	SI
8	SO
9	SK
10	GND
11	L5
12	L6
13	L7
14	RESET/
15	CKI
16	D0
17	D1
18	G2
19	G1
20	G0

COP 421/410/445 – 24-Pin

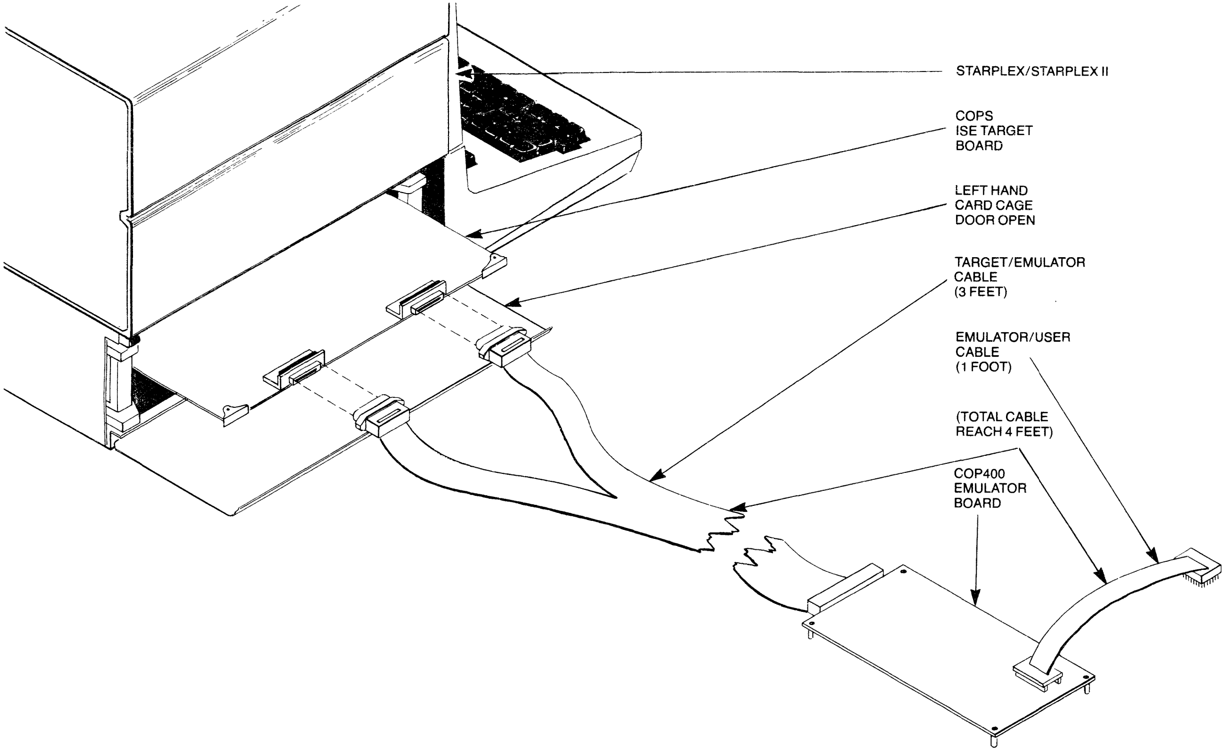
Pin	Signal
1	GND
2	CKO
3	CKI
4	RESET/
5	L7
6	L6
7	L5
8	L4
9	VCC
10	L3
11	L2
12	L1
13	D0
14	D1
15	D2
16	D3
17	G3
18	G2
19	G1
20	G0
21	SK
22	SO
23	SI
24	L0

COP420/444 – 28-Pin

Pin	Signal
1	GND
2	CKO
3	CKI
4	RESET/
5	L7
6	L6
7	L5
8	L4
9	IN1
10	IN2
11	VCC
12	L3
13	L2
14	L1
15	D0
16	D1
17	D2
18	D3
19	G3
20	G2
21	G1
22	G0
23	IN3
24	IN0
25	SK
26	SO
27	SI
28	L0

User Plug DC Characteristics Combined Specs For All Three Sockets

Signal	Symbol	Parameter	Conditions	Value		Unit
				Min	Max	
L0 – L7	VOH	Voltage, Output High	IOH = 100μA	2.4		V
D0 – D3	VOL	Voltage, Output Low	IOL = 1.6mA		0.4	V
G0 – G3	IOH	Current, Output High			−100	μA
SO, SK	IOFF	Hi-z Output Leakage		−10	+10	μA
CKO	IOL	Current, Output Low			1.6	mA
L0 – L7						
CKI	VIH	Voltage, Input High		2.0		V
SI	VIL	Voltage, Input Low			0.8	V
IN0 – IN3						
G0 – G3						
RESET/	VIH	Voltage, Input High		.7VCC		V
	VIL	Voltage, Input Low			0.6	V
		Hysteresis		1.0		V


STARPLEX/STARPLEX II
COPS ISE TARGET BOARD
LEFT HAND CARD CAGE DOOR OPEN
TARGET/EMULATOR CABLE (3 FEET)
EMULATOR/USER CABLE (1 FOOT)
(TOTAL CABLE REACH 4 FEET)
COP400 EMULATOR BOARD

Installation of the COPS ISE Target Board and an Emulator Board

Order Information

(Includes ISE Target Board, STARPLEX/STARPLEX II Emulator Cable, Cross Assembler, complete software and user's manuals, software to create a disk file for transmission of customer ROM patterns and device I/O options.)

For STARPLEX Development Systems:

SPM-A15	COPS In-System Emulator (ISE) Package

For STARPLEX II Development Systems:

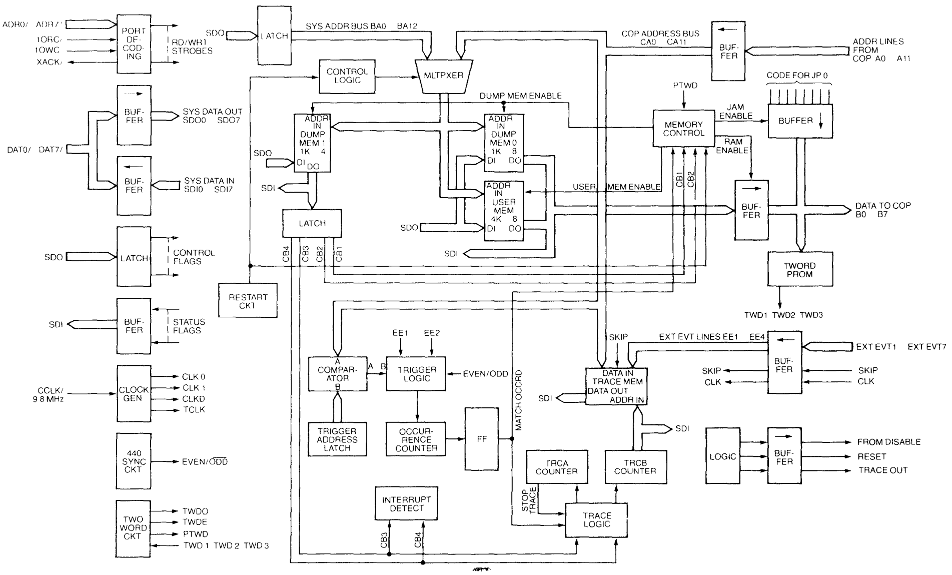
SPM-90-A15	COPS In-System Emulator (ISE) Package

COP400 Family Emulator Boards:

COP400-E02	402 Emulator Board
COP400-E02C	CMOS Emulator Board
COP400-E04L	404L Emulator Board
COP400-E24	440/2440 Emulator Board

Documentation

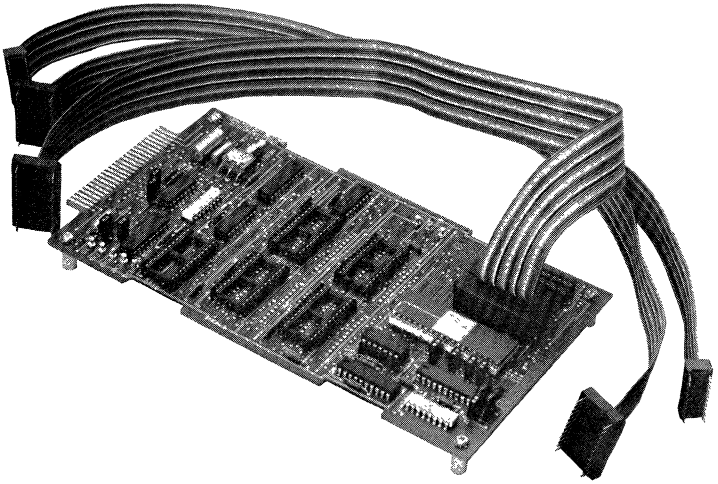
420305785-001	COP400 Microcontroller Family User's Manual
420306469-001	COP400 In-System Emulator Card User's Manual
420306253-001	COPS Cross-Assembler Software User's Manual
420306254-001	COPS ISE Operator's Manual



Target PWA Block Diagram

National Semiconductor

COP400 In-System Emulator™ Boards



Product Overview

The COP400 Emulator Boards enable in-system emulation of the entire COP400 Microcontroller family. To emulate the desired COP4XX part requires the appropriate COP400 Emulator Board with the appropriate ROMless COP4XX microcontroller part. COP400 In-System Emulator Boards currently available are the following:

COP400-E02 COP400-E02C COP400-E04L and COP400-E24 boards.

Emulator Boards and ROMless Parts for COP4XX Device Emulation

Emulator Board	ROMless Part	Parts Emulated
COP400-E02	COP401L*	COP410L
COP400-E04L	COP401L	COP411L
COP400-E02	COP402	COP420 COP421 COP422
COP400-E04L	COP404L	COP420L COP421L COP422L COP444L COP445L
COP400-E02C	SLO402	COP410C COP411C COP420C COP421C
COP400-E24	COP404	COP404 COP441 COP442
COP400-E24	COP2404**	COP2440 COP2441 COP2442

Flexible Configurations

The emulator board may be used stand-alone with EPROMs and external power supply or as a peripheral to a development system. The COP400 Emulator Boards are designed to interface with either the COP400-PDS Development System or the STARPLEX™ Development Systems.

Useful and Informative Features

When used in conjunction with a development system, the emulator adds the capabilities of real-time program tracing, breakpoint/single-stepping, and speedy program updating, resulting in rapid program evolution from conception through debug to final product.

Physical Features

The emulator board is a double-sided printed circuit board mounted on four 0.5-inch nylon stand-offs for easy access to jumpers and sockets. The processing is carried out by the ROMless microcontroller located near the top center of the board. At the bottom of the board is a 50-pin edge connector used to interface to the development system via the emulator-card cable. Three DIP-to-DIP cables are supplied with each emulator board.

For the COP400-E02 and COP400-E04L boards, the DIP-to-DIP cables are the 20-pin, 24-pin, and 28-pin. The 20-pin socket is used for emulating a COP411L device. The 24-pin socket is used for emulating the COP410L, COP421L, COP421, and COP445L devices. The 28-pin socket is used to emulate the COP420L, COP420, and COP444L devices.

* The COP401L has the CKO pin selected as the RAM Keep Alive Option. This pin must be connected to the V_{CC} power supply in the user's system and W4 installed on the COP400-E02 or COP400-E04L Board.

** As shipped, the COP400-E24 board contains a COP404 ROMless part. For emulating the COPs 2440, 2441 or 2442, install the COP2404 shipped with the board.

For the COP400-E02C board, the DIP-to-DIP cables are the 20-pin, 24-pin, and 28-pin. The 20-pin socket is used for emulating the COP411C. The 24-pin socket is used for emulating the COP410C and COP412C. The 28-pin socket is used for emulating the COP420C.

For the COP400-E24 board, the DIP-to-DIP cables are the 24-pin, 28-pin, and 40-pin. The 40-pin socket is used for emulating the COP440 and COP2440 devices. The 28-pin socket is used for emulating the COP441 and COP2441 devices. The 24-pin socket is used for emulating the COP442 and COP2442 devices.

Emulator 50-pin Edge Connector Assignments

Connector No.	Name	Description
1	GND	Signal and power return
2	GND	Signal and power return
3	V_{CC}	$+5V_{DC}$ power from Development System
4	V_{CC}	$+5V_{DC}$ power from Development System
5	EX2	Buffered External Event
6	EX1	Buffered External Event
7	EX4	Buffered External Event
8	EX3	Buffered External Event
9	CLK	Buffered AD/$\overline{\text{DATA}}$ signal from COP4XX
10	SKIP	COP4XX skip status line
11	A8	Address Bit
12	A9	Address Bit
13	A3	Address Bit
14	A7	Address Bit
15	A1	Address Bit
16	A2	Address Bit
17	A4	Address Bit
18	A0	Least significant address bit
19	A6	Address Bit
20	A5	Address Bit
21	Not Used	
22	A10	Most significant address bit
23	Not Used	
24	Not Used	
25	Not Used	
26	Not Used	
27	Not Used	
28	Not Used	
29	Not Used	
30	Not Used	
31	Not Used	
32	Not Used	
33	B0	Least significant COP object code bit
34	B7	Most significant COP object code bit
35	B2	Object code bit
36	B5	Object code bit
37	B3	Object code bit
38	B4	Object code bit
39	B6	Object code bit
40	B1	Object code bit
41	TRIGGER OUT	BREAKPOINT/TRACE indicator
42	Not Used	
43	RST*	Same as RESET*
44	PROM DISABLE	Select PROM or Shared Memory mode
45	See Note 1	
46	See Note 1	
47	V_{CC}	$+5V_{DC}$ power from Development System
48	V_{CC}	$+5V_{DC}$ power from Development System
49	GND	Power and signal return
50	GND	Power and signal return

Note 1: Pins 45 and 46 are used as follows:

PDS	with target board 980306552 REV A or later, normally not used. with target board 980305551 REV F or earlier, $-12V_{DC}$ from the PDS.
STARPLEX	with target board 980306254, normally not used. However, jumper W5 on the target board may be installed to supply $-12V_{DC}$ to the emulator board.

Prerequisites

- As A Stand-Alone
  — Appropriate EPROMs with external power supply.
- As A Peripheral To A PDS Development System.
  For COP400-E02, COP400-E04L, and COP400-E02C Emulator Boards:
  — Any COP400 PDS Development System
  For COP400-E24 Emulator Board:
  — (See Footnote '1' Below.)
- As A Peripheral To A STARPLEX™ Development System.
  — Any STARPLEX/STARPLEX II™ Development System.
  — COPS™ In-System Emulator (ISE™) Package (SPM-A15 or SPM-90-A15)

[1] The COP400-E24 is compatible with all STARPLEX and STARPLEX II systems as well as COP400-PDS systems shipped after October 1981. PDS systems shipped prior to this date may be upgraded to this level by purchase of an upgrade kit. Order number of the upgrade kit is COP400-A2. Order number for PDS systems shipped after October 1981 will be COP400-PDS2.

[2] Number not available at printing time.

Order Information

COP400-E02	Emulator Board
COP400-E02C	CMOS Emulator Board
COP400-E04L	Emulator Board
COP400-E24	404/2404 Emulator Board

Documentation

- Minimum Documentation

420306469-001	COP400 In-Circuit Emulator Boards
(See Footnote '2' Below)	COP400 Microcontroller Family Databook

- When Used With COP400 PDS
  Above Minimum Documentation Plus:

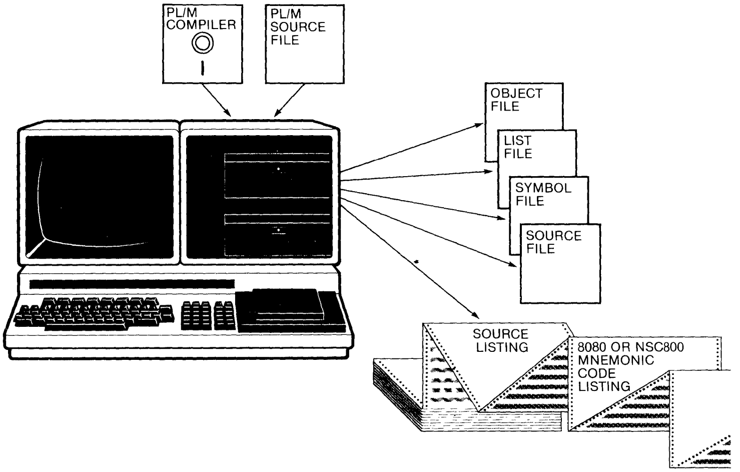
420305548-002	COP400 PDS (Product Development System)

- When Used With STARPLEX Development Systems
  Above Minimum Documentation Plus:

420306254-001	COPS ISE (In-System Emulator, SPM-A15, SPM-90-A15) Operator's Manual
420306253-001	COPS Cross-Assembler (SFW-A006-1C, SFW-90-A006) User's Manual

National Semiconductor

PLM80
PL/M High Level Language Compiler for STARPLEX™ Development Systems



- **Executes on all STARPLEX/ STARPLEX II™ Development Systems**
- **Code generation for 8080/8085 and NSC800™/Z80 microprocessors**
- **Relocatable and linkable object code output**
- **Reentrant procedures as specified by user**
- **Compatible with existing industry standard PL/M-80**
- **Hardware access via highlevel statements (interrupt systems, absolute addresses, and input/output ports)**

Product Description

PLM80 is a high level language compiler designed for STARPLEX and STARPLEX II Development Systems. Available in two versions, this highly efficient compiler generates relocatable object code for 8080/8085 and NSC800/Z80 microprocessors.

PL/M has proven to be one of the most popular, effective and powerful program development tools available. Programmer productivity and reliability are greatly improved because the programmer can concentrate on system development rather than all the details of assembly languages. Since PL/M uses data structures that are very close to typical microprocessor architectures, it allows for efficient use of the machine. PL/M programs are efficiently converted to assembly language instructions, thus requiring fewer statements. Software development and maintenance costs are significantly reduced.

Free form PL/M source programs are efficiently and effectively converted into 8080/8085 or NSC800/Z80 assembly language instructions. A given program, when written in PL/M, requires fewer statements than would the equivalent program written in assembly language. Thus, software development and maintenance costs are significantly reduced due to the problem oriented structure that results naturally from the use of PL/M. User programming conventions and structured programming techniques are easily accommodated by the free form source statements of PL/M.

Functional Description

The PLM80 Compiler is a STARPLEX System program which accepts STARPLEX PLM80 language source modules and produced linkable object modules. Object modules may be linked to form executable PLM80 programs. The PLM80 compiler is also designed to accept programs written in the industry standard PL/M programming language.

The STARPLEX PLM80 compiler invocation is similar to that of other STARPLEX software. The 8080 version of the compiler in particular has all the features of the existing industry standard PL/M-80 compiler. In many cases, no changes to existing PL/M-80 programs are required. However, STARPLEX PLM80 has a number of superior enhancements which may be incorporated into existing PL/M-80 programs to make it faster, smaller, and easier to debug. What modification is required can be done very easily.

Compilation is one step in the formation of an executable PLM80 program. The formation of a complete program involves the following steps:

- Writing the PLM80 "Source Modules" using the TEXT EDITOR.
- Compiling the source files to produce "Object Modules."
- Linking the object modules to create an executable PLM80 "Load Module."

When the source module(s) have been created using the TEXT EDITOR for compilation, choose the correct PLM80 diskette for the type of compilation desired. The 8080 version may be used for programs to be executed on 8080 and 8085 based systems. The NSC800 version may be used for NSC800 or Z80 based systems.

Enhancements

Lexical Extensions

PLM80 will allow the underscore character "_" in identifiers and in numeric constants, to aid legibility. For example, NAME_TABLE or 1100_0111B. Unlike the industry standard PL/M "$", which PLM80 also will accept, the underscore is a significant character in identifiers; thus, A_TO_M is a distinct identifier from AT_OM and from ATOM.

PLM80 will accept the ASCII form-feed character as lexically equivalent to a blank; the form-feed, like the EJECT compiler control, will cause a page eject in the listing file.

Explicit Locator References

In the industry standard PL/M, each based variable is associated with a unique pointer. The pointer is specified in the based declaration, and does not appear explicitly in references to the based variable.

Declare Statement Syntax

The industry standard PL/M requires attributes to appear in a specified order within a declaration. This restriction has been relaxed in PLM80.

Declaration of Arrays

The keyword ARRAY has been added for optional use in dimension-specifications.

The industry standard syntax for based array declarations is misleading because the dimension-specifier appears to be "attached" to the wrong variable:

```
DECLARE B BASED P(100) BYTE;
```

creates a 100-byte array b, based on a scalar pointer P. PLM80 will provide a number of superior alternative forms, e.g.,

```
DECLARE B(100) BYTE BASED P;
DECLARE B BASED P ARRAY(100) BYTE:
```

The second of these forms permits the industry standard form to be modified easily, the only difference is the addition of the keyword "ARRAY". It also accepts the standard form without the usage of "ARRAY".

Empty Blocks and Procedures

PLM80 will accept a block or procedure that contains no executable statements. This is not permitted by the industry standard.

Do-Case Extensions

PLM80 will accept case-selectors in the range of a do-case statement, thus permitting the programmer to create sparsely populated case constructs without sacrificing efficiency. Multiple specifiers will be permitted on a single statement, so the programmer need not write duplicate code.

PLM80 will accept an otherwise-clause in the range of a do-case statement. This makes it unnecessary to write out the action for every case if most of them are identical.

PLM80 will do range checking in the case construct. Unspecified or out-of-range cases will cause a jump to the statement following the do-case-block.

Example: The following code executes special statements if I is 6, 28, 496 or 8128. If I has any other value, the statement in the otherwise-clause is executed.

```
DO CASE I;
    6:
   28: DO;          /* This entire do-block is    */
            . . .   /*    executed if I = 6 or 28. */
       END;
 8128: . . .        /* This statement is executed */
                    /*    if I = 8128.            */
  496: . . .        /* This statement is executed */
                    /*    if I = 496.             */
  OTHERWISE . . .   /* This statement is executed */
                    /*    if I has any other value. */
END;
```

Iterative DO

In the industry standard, the expressions in "TO" and "BY" options in an iterative do-statement are evaluated each time the loop is executed. Worse, the time of evaluation depends upon the datatype of the index variable. PLM80 adopts the convention that these expressions are evaluated once, prior to entry to the loop. The values calculated at that time will be preserved and reused. This makes for faster running time.

User Interface

Listings

The PLM80 compiler can provide, upon request, source and object listings. Diagnostics will be provided; regardless of list options.

Source listings will include statement numbers, block nesting depth, diagnostics, a list of the options present for the compilation, and statistics (e.g., resources used) for the compilation.

Object listings will show object code (pseudo-assembly language and actual machine code) and approximate statement numbers.

Compile-Time Diagnostics

For syntax errors, the diagnostic message will appear in the source listing immediately after the point at which the error was recognized. For example:

```
LINE STMT LEVEL .... + ....1.... + ....2.... + ....3.... + ....4.... + ....5.... +

  13      9   3                     z = x + * x;
****    ERROR  21 **** Stmt 9 — near '*' — Syntax error; skipping input to ";"
```

At the end of the source listing for a module, the compiler will list all other diagnostic messages for that module, sorted by statement number. Each message will clear and concise, and will describe the error in detail. For example:

```
****   ERROR 48 **** Stmt  8 — Missing data type attribute
****   ERROR 43 **** Stmt 11 — Undeclared identifier
****   ERROR 54 **** Stmt 13 — Reference to member of undeclared
                               structure
```

Code Generation and Optimization

The PLM80 compiler handles: local optimizations, basic block optimization, efficient register allocation, special casing for common constructs, some strength reduction, removal of dead code and of branch-around-branch. This, in effect, produces smaller, faster and more efficient object code than the industry standard PL/M compiler.

Run-Time Support

The run-time support package contains those built-in procedures that are not compiled as in-line code, procedures for the arithmetic operations not performed in-line, and stack management.

Example PLM80 Program

```
STARPLEX PL/M-80 Rev A-810428   MODULE:SEARCH__MODULE
OPTIONS:  FDS1:EXPROG LIST CODE
LINE  STMT  LEVEL .... + .... 1.... + .... 2.... + .... 3.... + .... 4.... + .... 5.... + .... 6.... + .... 7.... + .... 8.... +

  1    1     0   SEARCH__MODULE:
  2                 D O ; /* This module contains a typed procedure named SEARCH. SEARCH *
  3                        * searches the based array BUFFER for the first occurrence of the strind *
  4                        * contained in the based array WORD. If the strind is found, SEARCH *
  5                        * returns the subscript value of the element of BUFFER containing the *
  6                        * first character. Otherwise, SEARCH returns a value greater than the *
  7                        * length of the buffer.                                              */

  9    2     2      SEARCH: PROCEDURE   (BUF__PTR,LENGTH,WORD__,PTR,WORD__LENGTH) ADDRESS PUBLIC;
 10    3     2            DECLARE       (BUF__PTR,LENGTH,WORD__PTR,WORD__LENGTH) ADDRESS,
 11                                     BUFFER BASED BUF__PTR ARRAY(1) BYTE,
 12                                     WORD BASED WORD__PTR ARRAY(1) BYTE,
 13                                     FIRST__CHAR ADDRESS,
 14                                     (I, K) ADDRESS,
 15                                     FOUND BYTE,
 16                                     TRUE LITERALLY '0FFH',
 17                                     FALSE LITERALLY '00H';

 20    4     2            SET__FIRST__CHAR:
 21                            DO FIRST__CHAR = 0 TO LENGTH – 1;
 22    5     3                      I = FIRST__CHAR;
 23    6     3                      K = 0;
 24    7     3                      FOUND = TRUE;

 26    8     3                      COMPARE:
 27                                      DO WHILE (FOUND = TRUE) AND (K<WORD__LENGTH);
 28    9     5                                IF BUFFER (I) = WORD (K) THEN DO;
 29   11     5                                     I = I + 1;
 30   12     5                                     K = K + 1;
 31   13     4                                   END;
 32   14     4                                ELSE FOUND = FALSE;
 33   15     3                      END COMPARE;
 34   16     3                 IF FOUND = TRUE THEN RETURN FIRST__CHAR;
 35   18     2            END SET__FIRST__CHAR;

 37   19     2            RETURN LENGTH + 1;

 39   20     1      END SEARCH;
 40   21     0   END SEARCH__MODULE
```

```
00A2'              ;>>>> STATEMENT 2
00A2'    EB        XCHG
00A3'    22 0006"  SHLD     WORD__LENGTH
00A6'    69        MOV      L,C
00A7'    60        MOV      H,B
00A8'    22 0004"  SHLD     WORD__PTR
00AB'    D1        POP      D
00AC'    E1        POP      H
00AD'    22 0002"  SHLD     LENGTH
00B0'    E1        POP      H
00B1'    D5        PUSH     D
00B2'    22 0000"  SHLD     BUF__PTR
  .        .         .        .
  .        .         .        .
  .        .         .        .
0087'              ;>>>> STATEMENT 18

0087'                                $00012:

                   CSEG
                   ORG      0081
0081'    0087'     DW       $00012
                   CSEG
                   ORG      $00012
0087'    2A 0008"  LHLD     $00002
008A'    23        INX      H
008B'    22 0008"  SHLD     $00002

008E'                                $00003:

                   CSEG
                   ORG      0021
```

```
0021'    008E'     DW       $00003

                   CSEG
                   ORG      $00003
008E'    2A 000F"  LHLD     $00005
0091'    EB        XCHG
0092'    2A 0008"  LHLD     $00002
0095'    7B        MOV      A,E
0096'    95        SUB      L
0097'    7A        MOV      A,D
0098'    9C        SBB      H
0099'    D2 0023'  JNC      $00004

009C'              ;>>>> STATEMENT 19
009C'    2A 0002"  LHLD     LENGTH
009F'    23        INX      H
00A0'    C9        RET

00A1'              ;>>>> STATEMENT 20
00A1'    C9        RET

00A2'              ;>>>> STATEMENT 21
00A2'    FB        EI
00A3'    76        HLT
MODULE STATISTICS:

     21 STATEMENTS PROCESSED
      0 DIAGNOSTIC(S) ISSUED

  CODE SEGMENT SIZE                   181D (00B5H)
  DATA SEGMENT SIZE                    19D (0013H)
```

Prerequisites

Any STARPLEX /STARPLEX II Development System.

Order Information

SFW-A50 PLM80, PL/M Compiler to generate 8080/8085 object program on STARPLEX Development System.

SFW-A60 PLM80, PL/M Compiler to generate NSC800/Z80 object program on STARPLEX Development System.

SFW-90-A50 PLM80, PL/M Compiler to generate 8080/ 8085 object program on STARPLEX II Development System.

SFW-90-A60 PLM80, PL/M Compiler to generate NSC800/Z80 object program on STARPLEX II Development System.

Documentation

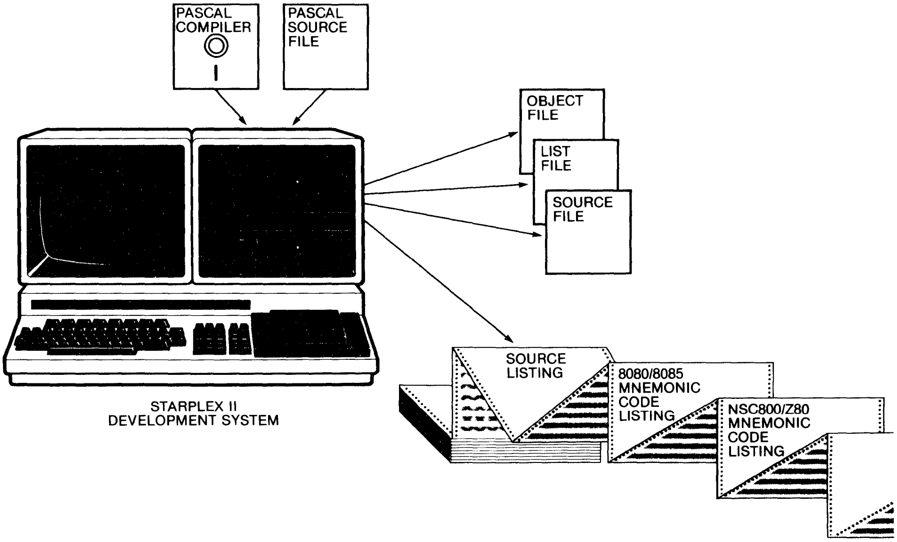
420306371-001 STARPLEX PLM80 Compiler Software Reference Manual (Included with SFW-A50, SFW-A60, SFW-90-A50, SFW-90-A60)

420305789-001 8080/8085 Macroassembler Software User's Manual

420306198-001 NSC800 Macroassembler Software User's Manual

National Semiconductor

PASCAL
PASCAL High Level Language Compiler For STARPLEX II™ Development Systems


PASCAL COMPILER
PASCAL SOURCE FILE
OBJECT FILE
LIST FILE
SOURCE FILE
STARPLEX II DEVELOPMENT SYSTEM
SOURCE LISTING
8080/8085 MNEMONIC CODE LISTING
NSC800/Z80 MNEMONIC CODE LISTING

- **Executes On All STARPLEX II Development Systems**
- **Compatible With Existing ISO Standard PASCAL**
- **Highly Portable And Extended Source Programs**
- **Code Generation For 8080/8085 and NSC800™/Z80 Microprocessors**
- **Relocatable And Linkable Object Code Output**
- **Reentrant Procedures as Specified by User**
- **Extensions For Easy Hardware Access Via High Level Statements (Absolute Addresses and Input/Output Ports)**

Product Description

PASCAL is a high level language compiler designed for STARPLEX II Development Systems. Available in two versions, this highly efficient and powerful compiler generates relocatable object code for 8080/8085 and NSC800/Z80 microprocessors.

PASCAL has proven to be one of the most popular, effective and powerful program development tools available today. With STARPLEX II PASCAL, programmer productivity is greatly improved because the programmer can concentrate on system development rather than all the details of assembly languages. Since PASCAL uses data structures that are very close to typical microprocessor architectures, it allows for efficient use of the machine. PASCAL programs are efficiently converted to assembly language instructions thus requiring fewer statements. Software development and maintenance costs are significantly reduced.

Free form PASCAL source programs are efficiently and effectively converted into 8080/8085 and NSC800/Z80 assembly language Instructions. A given program, when written in PASCAL, requires much fewer statements than would the equivalent program written in assembly language. Thus, software development and maintenance costs are significantly reduced due to the block oriented structure that results naturally from the use of PASCAL. User programming conventions and structured programming techniques are easily accommodated by the free form source statements of PASCAL.

Functional Description

The STARPLEX II PASCAL compiler is a system program which accepts PASCAL language source modules and produces linkable object modules. Object modules may be linked to form executable PASCAL programs. The STARPLEX II PASCAL compiler is compatible with the International Standards Organization (ISO) standard and has extensions to facilitate access to and manipulation of machine data structures. Code generated by the STARPLEX II PASCAL compiler is native machine code, rather than the intermediate p-code found in other microcomputer PASCAL compilers. The execution speed of programs compiled to machine code is much faster than those complied to p-code, thus maintaining the programming advantages of a high-level language without sacrificing execution speed.

The STARPLEX II PASCAL compiler invocation is similar to that of other STARPLEX II software. In many cases, no changes to existing PASCAL programs are required. STARPLEX II PASCAL has a number of extensions which may be incorporated into existing PASCAL programs to make it faster, smaller, and easier to debug. In many cases, because of STARPLEX II PASCAL's many low-level escapes to the machine level, programs written in STARPLEX II PASCAL can be comparable in speed to programs written in assembly language.

The STARPLEX II PASCAL compiler reads source files containing PASCAL source modules and produces:

- a linkable object module containing object code,
- a listing of the STARPLEX II PASCAL source statements,
- a listing of syntax and semantic errors and warning messages,
- an optional listing of object code in assembly language mnemonics.

Compilation is one step in the formation of an executable PASCAL program. The formation of a complete program involves the following steps:

- Writing the PASCAL "Source Modules" using the TEXT EDITOR.
- Compiling the source files to produce linkable "Object Modules".
- Linking the object modules to create an executable PASCAL "Load Module".

When the source module(s) have been created using the STARPLEX II TEXT EDITOR, choose the correct PASCAL diskette for the type of compilation desired. The 8080/8085 version of the PASCAL compiler can generate code that can be used for an 8080 or 8085 based system, while the NSC800 version of the PASCAL compiler can generate code that can be used for an NSC800 or Z80 based systems.

Extensions

As stated before, the STARPLEX II PASCAL provides many extensions to the ISO standard PASCAL. The following is an overview of the extensions.

1. Direct files: To enhance standard PASCAL's file capabilities, direct (random access) files are implemented, and accessed with the SEEK procedure.

2. Variable-length strings: A special variable-length string type called the LSTRING is implemented in STARPLEX II PASCAL to overcome standard PASCAL's inadequate string-handling capabilities. Special predeclared procedures and functions are available to facilitate use of the feature.

3. Super arrays: A special variable-length array declaration permits both passing arrays of different lengths to a reference parameter, as well as dynamic allocation of arrays of difference lengths.

4. BYTE/WORD types: Predeclared BYTE (0-255) and WORD (0-65535) types are available to facilitate programming at the system level.

5. String reads: Strings can be read as structures rather than character by character as with the standard procedures READ and READLN.

6. Nondecimal numbering: Hexadecimal, octal, and binary numbering are allowed to facilitate programming at the byte and bit level.

7. Address types (segmented and unsegmented): A special address type is implemented to allow manipulation of actual machine addresses.

8. Interface to assembly language: PUBLIC and EXTERN procedures, functions and variables are implemented to allow for low-level interfacing to assembly language and library routines.

9. Separate compliation: MODULES are implemented to allow portions of a program to be compiled at separate times.

10. VALUE section: Variables in a program can be given initial constant values in the VALUE section of a program.

11. Structured function return values: Functions can return values of a structured type, as well as values of simple type.

12. Support for interactive files: A special internal mechanism called "lazy evaluation" allows normal interactive input from terminals.

13. OTHERWISE in CASE statements: An OTHERWISE clause can be used in CASE statements to avoid explicitly specifying each case constant.

14. STATIC attribute for variables: Variables can be given the STATIC attribute to indicate that they are allocated at a fixed location in memory rather than on the stack.

15. ORIGIN attributes: Variables, procedures, and functions can be given the ORIGIN attribute to indicate their absolute location in memory.

16. Underscores in identifiers: Identifiers may contain underscores to improve their readability.

User Interface

Listings

The PASCAL compiler can provide, upon request, source and object listings. Diagnostics will be provided, regardless of list options.

Source listings will include statement numbers with corresponding source statements.

Object listings will show line number with corresponding object code (pseudo-assembly language) as well as relative memory locations and statistics (e.g., resources used) for the compilation.

Diagnostic Messages

Each error and warning flag contains a code number and a brief message. The code number indicates where in the list of error messages, a detailed explanation of that particular error or warning can be found. The brief message indicates an overview explanation of the incorrect condition detected.

Code Generation and Optimization

The STARPLEX II PASCAL compiler handles local optimizations: basic block optimization, competent register allocation, special casing for common constructs, some strength reduction, removal of dead code and of branch-around-branch. This produces smaller, faster and efficient object code.

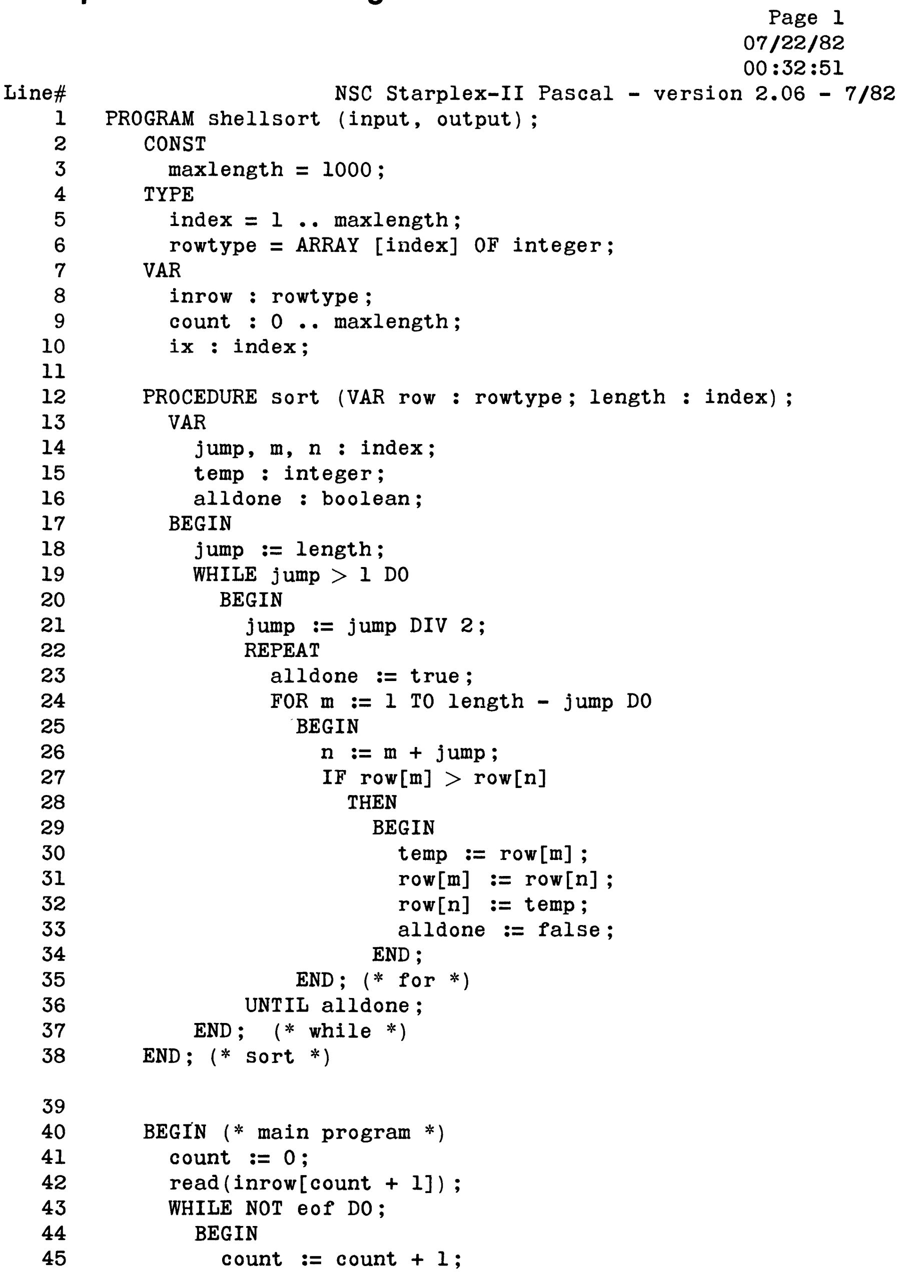
Predefined Procedures and Functions for Run-Time Support

A number of predefined procedures and functions are included in the PASCAL compiler library which the user can use to facilitate his programming. These procedures and functions perform I/O, data allocation, arithmetic, string, and system operations. The procedures and functions are divided into the following categories:

- I/O routines
- Dynamic allocation routines
- Mathematic routines
- String routines
- Manipulation routines
- Library management routines

While the Library procedures and functions must be declared EXTERN, all the other functions and procedures are predeclared and hence do not have to be declared in the user's program. The use of these procedures and functions therefore do not require extra statement lines in the program itself.

Example of a PASCAL Program



```
                                                            Page 1
                                                          07/22/82
                                                          00:32:51
                          NSC Starplex-II Pascal - version 2.06 - 7/82
PROGRAM shellsort (input, output);
   CONST
     maxlength = 1000;
   TYPE
     index = 1 .. maxlength;
     rowtype = ARRAY [index] OF integer;
   VAR
     inrow : rowtype;
     count : 0 .. maxlength;
     ix : index;

   PROCEDURE sort (VAR row : rowtype; length : index);
     VAR
       jump, m, n : index;
       temp : integer;
       alldone : boolean;
     BEGIN
       jump := length;
       WHILE jump > 1 DO
         BEGIN
           jump := jump DIV 2;
           REPEAT
             alldone := true;
             FOR m := 1 TO length - jump DO
               BEGIN
                 n := m + jump;
                 IF row[m] > row[n]
                   THEN
                     BEGIN
                       temp := row[m];
                       row[m] := row[n];
                       row[n] := temp;
                       alldone := false;
                     END;
               END; (* for *)
           UNTIL alldone;
       END;  (* while *)
   END; (* sort *)


   BEGIN (* main program *)
     count := 0;
     read(inrow[count + 1]);
     WHILE NOT eof DO;
       BEGIN
         count := count + 1;
```

Example of a PASCAL Program (Cont'd)

```
                                                                Page 2
                                                              07/22/82
                                                              00:33:33
Line#                     NSC Starplex-II Pascal - version 2.06 - 7/82
   46              read(inrow[count + 1]);
   47            END; (* while *)
   48          IF count  >  0
   49           THEN
   50             BEGIN
   51               sort(inrow, count);
   52               FOR ix := 1 TO count DO
   53                 write(inrow[ix])
   54             END
   55           ELSE write('no input')
   56       END.  (* shellsort *)

   57
```

```
 procedure / function:    SORT


    ** 0001"    DB      01    ; level
    ** 0002"    CALL    RENGQQ
    ** 0005"    DW      0004, 0014    ; return displacement, frame length
L18:
    ** 0009"    CALL    LSAGQQ
    ** 000C"    <B>     0002
    ** 000D"    PUSH    HL
    ** 000E"    LD      HL,0100
    ** 0011"    PUSH    HL
    ** 0012"    LD      HL,E803
    ** 0015"    PUSH    HL
    ** 0016"    CALL    RCIEQQ
    ** 0019"    CALL    ASAGQQ
    ** 001C"    <B>     0008
L19:
I4:
    ** 001D"    CALL    LSAGQQ
    ** 0020"    <B>     0008
    ** 0021"    LD      DE,FEFF
    ** 0024"    LD      A,H
    ** 0025"    ADD     A,A
    ** 0026"    JP      C,I5
    ** 0029"    ADD     HL,DE
    ** 002A"    JP      NC,I5
L21:
    ** 002D"    LD      DE,0100
    ** 0030"    CALL    LSAGQQ
    ** 0033"    <B>     0008
    ** 0034"    CALL    SRDGQQ
    ** 0037"    PUSH    HL
    ** 0038"    LD      HL,0100
    ** 003B"    PUSH    HL
```

Example of a PASCAL Program (Cont'd)

```
    ** 003C"    LD      HL,E803
    ** 003F"    PUSH    HL
    ** 0040"    CALL    RCIEQQ
    ** 0043"    CALL    ASAGQQ
    ** 0046"    <B>     0008
L22:
I8:
L23:
    ** 0047"    LD      HL,0100
    ** 004A"    CALL    ASGGQQ
    ** 004D"    <B>     0010
L24:
    ** 004E"    CALL    LSBGQQ
    ** 0051"    <B>     0002
    ** 0052"    CALL    LSAGQQ
    ** 0055"    <B>     0008
    ** 0056"    CALL    SVBGQQ
    ** 0059"    CALL    ASAGQQ
    ** 005C"    <B>     0012
    ** 005D"    CALL    LSAGQQ
    ** 0060"    <B>     0012
    ** 0061"    LD      DE,FFFF
    ** 0064"    LD      A,H
    ** 0065"    ADD     A,A
    ** 0066"    JP      C,I10
    ** 0069"    ADD     HL,DE
    ** 006A"    JP      NC,I10
    ** 0070"    PUSH    HL
    ** 0071"    PUSH    HL
    ** 0072"    LD      HL,E803
    ** 0075"    PUSH    HL
    ** 0076"    CALL    RCIEQQ
    ** 0079"    CALL    ASAGQQ
    ** 007C"    <B>     000A
    ** 007D"    CALL    LSAGQQ
    ** 0080"    <B>     0012
    ** 0081"    PUSH    HL
    ** 0082"    LD      HL,0100
    ** 0085"    PUSH    HL
    ** 0086"    LD      HL,E803
    ** 0089"    PUSH    HL
    ** 008A"    CALL    RCIEQQ
I11:
L26:
    ** 008D"    CALL    LSBGQQ
    ** 0090"    <B>     0008
    ** 0091"    CALL    LSAGQQ
    ** 0094"    <B>     000A
    ** 0095"    CALL    AEBGQQ
    ** 0098"    PUSH    HL
    ** 0099"    LD      HL,0100
    ** 009C"    PUSH    HL
    ** 009D"    LD      HL,E803
    ** 00A0"    PUSH    HL
    ** 00A1"    CALL    RCIEQQ
    ** 00A4"    CALL    ASAGQQ
    ** 00A7"    <B>     000C
L27:
    ** 00A8"    CALL    LSAGQQ
    ** 00AB"    <B>     000C
```

Example of a PASCAL Program (Cont'd)

```
    ** 00AC"   ADD    HL,HL
    ** 00AD"   EX     DE,HL
    ** 00AE"   DEC    HL,DE
    ** 00AF"   DEC    HL,DE
    ** 00B0"   CALL   LSAGQQ
    ** 00B3"   <B>    0000
    ** 00B4"   CALL   0VBGQQ
    ** 00B7"   CALL   LSAGQQ
    ** 00BA"   <B>    000A
    ** 00BB"   ADD    HL,HL
    ** 00BC"   PUSH   DE
    ** 00BD"   EX     DE,HL
    ** 00BE"   DEC    HL,DE
    ** 00BF"   DEC    HL,DE
    ** 00C0"   CALL   LSAGQQ
    ** 00C3"   <B>    0000
    ** 00C4"   CALL   0VAGQQ
    ** 00C7"   POP    DE
    ** 00C8"   LD     A,D
    ** 00C9"   XOR    H
    ** 00CA"   LD     A,D
    ** 00CB"   JP     M,I4094
    ** 00CE"   LD     A,E
    ** 00CF"   SUB    L
    ** 00D0"   LD     A,D
    ** 00D1"   SBC    A,H
I4094:
    ** 00D2"   ADD    A,A
    ** 00D3"   JP     NC,I12
L30:
    ** 00D6"   CALL   LSAGQQ
    ** 00D9"   <B>    000A
    ** 00DB"   EX     DE,HL
    ** 00DC"   DEC    HL,DE
    ** 00DD"   DEC    HL,DE
    ** 00DE"   CALL   LSAGQQ
    ** 00E1"   <B>    0000
    ** 00E2"   CALL   0VAGQQ
    ** 00E5"   CALL   ASAGQQ
    ** 00E8"   <B>    000E
L31:
    ** 00E9"   CALL   LSAGQQ
    ** 00EC"   <B>    000C
    ** 00ED"   ADD    HL,HL
    ** 00EE"   EX     DE,HL
    ** 00EF"   DEC    HL,DE
    ** 00F0"   DEC    HL,DE
    ** 00F1"   CALL   LSAGQQ
    ** 00F4"   <B>    0000
    ** 00F5"   CALL   0VBGQQ
    ** 00F8"   CALL   LSAGQQ
    ** 00FB"   <B>    000A
    ** 00FC"   ADD    HL,HL
    ** 00FD"   PUSH   DE
    ** 00FE"   EX     DE,HL
    ** 00FF"   DEC    HL,DE
    ** 0100"   DEC    HL,DE
    ** 0101"   CALL   LSAGQQ
    ** 0104"   <B>    0000
    ** 0105"   ADD    HL,DE
```

Example of a PASCAL Program (Cont'd)

```
    ** 0106"    POP     DE
    ** 0107"    LD      (HL),E
    ** 0108"    INC     HL
    ** 0109"    LD      (HL),D
L32:
    ** 010A"    CALL    LSAGQQ
    ** 010D"    <B>     000C
    ** 010E"    ADD     HL,HL
    ** 010F"    EX      DE,HL
    ** 0110"    DEC     HL,DE
    ** 0111"    DEC     HL,DE
    ** 0112"    CALL    LSAGQQ
    ** 0115"    <B>     0000
    ** 0116"    ADD     HL,DE
    ** 0117"    CALL    LSBGQQ
    ** 011A"    <B>     000E
    ** 011B"    LD      (HL),E
    ** 011C"    INC     HL
    ** 011D"    LD      (HL),D
L33:
    ** 011E"    LD      HL,0000
    ** 0121"    CALL    ASGGQQ
    ** 0124"    <B>     0010
L34:
I12:
L35:
    ** 0125"    CALL    LSBGQQ
    ** 0128"    <B>     0012
    ** 0129"    CALL    LSAGQQ
    ** 012C"    <B>     000A
    ** 012D"    INC     HL
    ** 012E"    CALL    ASAGQQ
    ** 0131"    <B>     000A
    ** 0132"    DEC     HL,HL
    ** 0133"    LD      A,L
    ** 0134"    CP      E
    ** 0135"    JP      NZ,I11
    ** 0139"    CP      D
    ** 013A"    JP      NZ,I11
I10:
L36:
    ** 013D"    CALL    LSAGQQ
    ** 0140"    <B>     0010
    ** 0141"    LD      A,L
    ** 0142"    RRA
    ** 0143"    JP      NC,I8
L37:
    ** 0146"    JP      I4
I5:
L38:
I3:
    ** 0149"    CALL    PRAGQQ
    ** 014C"    DB      04
    ** 014D"    DB      00
```

Example of a PASCAL Program (Cont'd)

```
 procedure / function:    SHELLSOR


     ** 014E"    DB      00     ;level
     ** 014F"    CALL    RENGQQ
     ** 0152"    DW      0000, 0006    ; return displacement, frame length
     ** 0156"    CALL    INIFQQ
L41:
     ** 0159"    LD      HL,0000
     ** 015C"    LD      (COUNT),HL
L42:
     ** 015F"    LD      HL,INPFQQ
     ** 0162"    PUSH    HL
     ** 0163"    LD      HL,(COUNT)
     ** 0166"    ADD     HL,HL
     ** 0167"    EX      DE,HL
     ** 0168"    LD      HL,INROW
     ** 016B"    ADD     HL,DE
     ** 016C"    PUSH    HL
     ** 016D"    LD      HL,0180
     ** 0170"    PUSH    HL
     ** 0171"    LD      HL,FF7F
     ** 0174"    PUSH    HL
     ** 0175"    CALL    RTIFQQ
L43:
I14:
     ** 0178"    LD      HL,INPFQQ
     ** 017B"    PUSH    HL
     ** 017C"    CALL    EOFFQQ
     ** 017F"    LD      A,L
     ** 0180"    RRA
     ** 0181"    JP      C,I15
     ** 0184"    JP      I14
I15:
L45:
     ** 0187"    LD      HL,(COUNT)
     ** 018A"    CALL    INDGQQ
     ** 018D"    DB      01
     ** 018E"    PUSH    HL
     ** 018F"    LD      HL,0000
     ** 0192"    PUSH    HL
     ** 0193"    LD      HL,E803
     ** 0196"    PUSH    HL
     ** 0197"    CALL    RCIEQQ
     ** 019A"    LD      (COUNT),HL
L46:
     ** 019D"    LD      HL,INPFQQ
     ** 01A0"    PUSH    HL
     ** 01A1"    LD      HL,(COUNT)
     ** 01A4"    ADD     HL,HL
     ** 01A5"    EX      DE,HL
     ** 01A6"    LD      HL,INROW
     ** 01A9"    ADD     HL,DE
     ** 01AA"    PUSH    HL
     ** 01AB"    LD      HL,0180
     ** 01AE"    PUSH    HL
     ** 01AF"    LD      HL,FF7F
     ** 01B2"    PUSH    HL
     ** 01B3"    CALL    RTIFQQ
```

Example of a PASCAL Program (Cont'd)

```
L48:
    ** 01B6"    LD      HL,(COUNT)
    ** 01B9"    LD      DE,FFFF
    ** 01BD"    ADD     A,A
    ** 01BE"    JP      C,I16
    ** 01C1"    ADD     HL,DE
    ** 01C2"    JP      NC,I16
L51:
    ** 01C5"    LD      HL,INROW
    ** 01C8"    PUSH    HL
    ** 01C9"    LD      HL,(COUNT)
    ** 01CC"    PUSH    HL
    ** 01CD"    LD      HL,0100
    ** 01D0"    PUSH    HL
    ** 01D1"    LD      HL,E803
    ** 01D4"    PUSH    HL
    ** 01D5"    CALL    RCIEQQ
    ** 01D8"    PUSH    HL
    ** 01D9"    CALL    SORT
L52:
    ** 01DC"    LD      HL,(COUNT)
    ** 01DF"    CALL    ASAGQQ
    ** 01E2"    <B>     0004
    ** 01E3"    CALL    LSAGQQ
    ** 01E6"    <B>     0004
    ** 01E7"    LD      DE,FFFF
    ** 01EA"    LD      A,H
    ** 01EB"    ADD     A,A
    ** 01EC"    JP      C,I18
    ** 01EF"    ADD     HL,DE
    ** 01F0"    JP      NC,I18
    ** 01F3"    LD      HL, 0100
    ** 01F6"    PUSH    HL
    ** 01F7"    PUSH    HL
    ** 01F8"    LD      HL,E803
    ** 01FB"    PUSH    HL
    ** 01FC"    CALL    RCIEQQ
    ** 01FF"    LD      (IX),HL
    ** 0202"    CALL    LSAGQQ
    ** 0205"    <B>     0004
    ** 0206"    PUSH    HL
    ** 0207"    LD      HL,0100
    ** 020A"    PUSH    HL
    ** 020B"    LD      HL,E803
    ** 020E"    PUSH    HL
    ** 020F"    CALL    RCIEQQ
I19:
L53:
    ** 0212"    LD      HL,OUTFQQ
    ** 0215"    PUSH    HL
    ** 0216"    LD      HL,(IX)
    ** 0219"    ADD     HL,HL
    ** 021A"    EX      DE,HL
    ** 021B"    LD      HL,INROW+FFFE
    ** 021E"    CALL    OVAGQQ
    ** 0221"    PUSH    HL
    ** 0222"    LD      HL,FF7F
    ** 0225"    PUSH    HL
    ** 0226"    PUSH    HL
    ** 0227"    CALL    WTIFQQ
```

Example of a PASCAL Program (Cont'd)

```
    ** 022A"    CALL    LSBGQQ
    ** 022D"    <B>     0004
    ** 022E"    LD      HL,(IX)
    ** 0231"    INC     HL
    ** 0232"    LD      (IX),HL
    ** 0235"    DEC     HL,HL
    ** 0236"    LD      A,L
    ** 0237"    CP      E
    ** 023B"    LD      A,H
    ** 023C"    CP      D
    ** 023D"    JP      NZ,I19
I18:
    ** 0240"    JP      I20
I16:
L55:
    ** 0243"    LD      HL,OUTFQQ
    ** 0246"    PUSH    HL
    ** 0247"    LD      HL,0800
    ** 024A"    PUSH    HL
    ** 024B"    LD      HL,<const>      ;offset = 2
    ** 024E"    PUSH    HL
    ** 024F"    LD      HL,FF7F
    ** 0252"    PUSH    HL
    ** 0253"    PUSH    HL
    ** 0254"    CALL    WTSFQQ
I20:
I13:
    ** 0257"    CALL    PRAGQQ
    ** 025A"    DB      00
    ** 025B"    DB      00


Rom size: 614 decimal
Ram size: 2006 decimal
```

Prerequisites

Any STARPLEX II Development System with Rev F operating system or later.

Order Information

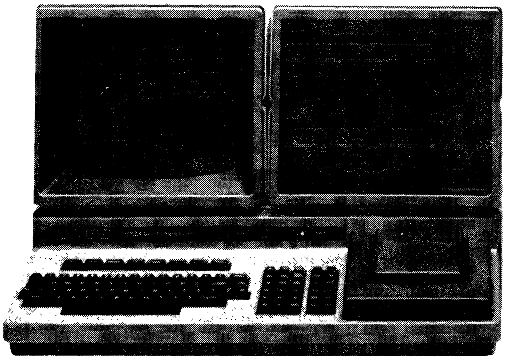
SFW-90-A300	PASCAL compiler to generate 8080/8085 linkable object code module(s) on STARPLEX II Development Systems.
SFW-90-A320	PASCAL compiler to generate NSC800/Z80 linkable object code module(s) on STARPLEX II Development Systems.

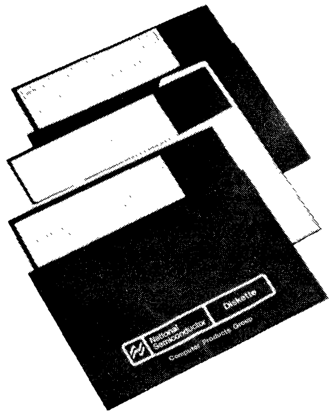
Documentation

420306680-001	STARPLEX II PASCAL Compiler Software Reference Manual (Included with SFW-90-A300 and SFW-90-A320)

National Semiconductor

PALASM™ Software Program





- **STARPLEX™ and STARPLEX II™ compatible**
- **Program generation for National PAL devices**
- **Compatible with National's Universal PROM Programmer Interface**

Product Description

Like PROM, the Program-Array-Logic (PAL) device has a single array of fusible links. These links may be left intact or "blown" to create various combinations of AND and OR gates to peform the desired function.

Programming a PAL device may be done manually with the designer marking PAL logic diagrams with appropriate fuse interconnections or via PALASM, an automatic fuse pattern generator.

PALASM is a FORTRAN IV program which transforms PAL symbolic equations into a format compatible with standard PAL programming personality modules. The output of this program is formatted to create the proper fuse patterns in PAL devices.

The pins of a PAL devices are represented as symbolic names and equations are given to specify how the pins are to be connected. For example, P = Q*R indicates that P is the logical AND of pins Q and R. The PALASM program translates these symbolic equations into a fuse pattern, absolute format, hex format, BHLF and/or BPNF format.

PALASM is supported on both STARPLEX and STARPLEX II development systems, and is an integral part of the Universal PROM Programmer Interface.

Functional Description

PALASM uses three designated files for input/output. These files are:

1. Input file
2. ABSOLUTE file
3. Object file

The Input file is the user-defined data file, a series of Boolean expressions defining the input/output relationship for each pin of the PAL devices.

The Output file may be either an ABSOLUTE file for input to the standard PROM programmer or an object file in one of several formats. These optional formats are useful for verification of the plot, or for inputs to various PROM programmers not directly supported by STARPLEX.

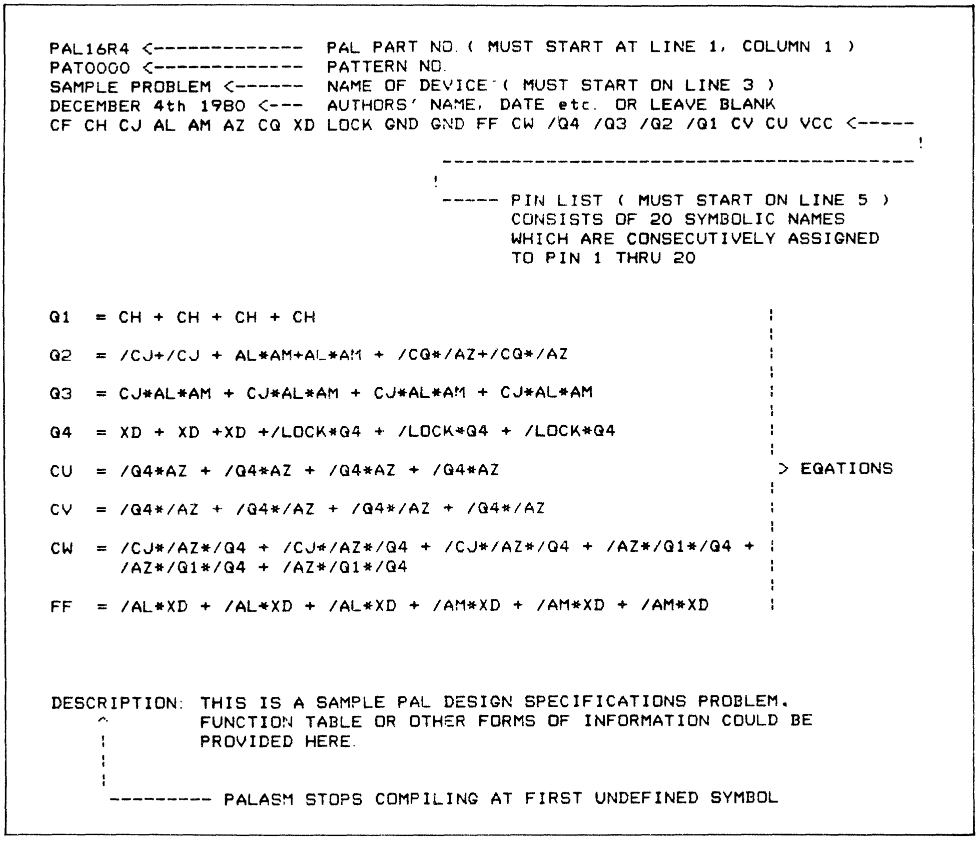
Optional formats are:

A = ABSOLUTE
B = PLOT
H = HEX
S = SHORT HEX
L = BHLF HIGH LOW
N = BPNF POSITIVE-NEGATIVE
M = MAP

When PALASM is invoked, the source file name is entered and the PALASM program then assembles the symbolic equations in the user's data file. Upon completion, PALASM requests the type of output file required. The program then executes the desired functions. Upon completion of the desired function, the user is prompted to enter the name of another file. This will continue until the Operation Code 'Q' (Quit) is entered.

At this point, the PROM Programmer utility program is used to program the PAL device.

Sample PAL Design Specification (Source File)



```
PAL16R4 <-------------  PAL PART NO. ( MUST START AT LINE 1, COLUMN 1 )
PAT0000 <-------------  PATTERN NO.
SAMPLE PROBLEM <------  NAME OF DEVICE ( MUST START ON LINE 3 )
DECEMBER 4th 1980 <---  AUTHORS' NAME, DATE etc. OR LEAVE BLANK
CF CH CJ AL AM AZ CQ XD LOCK GND GND FF CW /Q4 /Q3 /Q2 /Q1 CV CU VCC <-----
                                                                          !
                                  -----------------------------------------
                                 !
                                  ----- PIN LIST ( MUST START ON LINE 5 )
                                        CONSISTS OF 20 SYMBOLIC NAMES
                                        WHICH ARE CONSECUTIVELY ASSIGNED
                                        TO PIN 1 THRU 20


Q1   = CH + CH + CH + CH                                      !
                                                              !
Q2   = /CJ+/CJ + AL*AM+AL*AM + /CQ*/AZ+/CQ*/AZ                !
                                                              !
Q3   = CJ*AL*AM + CJ*AL*AM + CJ*AL*AM + CJ*AL*AM              !
                                                              !
Q4   = XD + XD +XD +/LOCK*Q4 + /LOCK*Q4 + /LOCK*Q4            !
                                                              !
CU   = /Q4*AZ + /Q4*AZ + /Q4*AZ + /Q4*AZ                       > EQATIONS
                                                              !
CV   = /Q4*/AZ + /Q4*/AZ + /Q4*/AZ + /Q4*/AZ                  !
                                                              !
CW   = /CJ*/AZ*/Q4 + /CJ*/AZ*/Q4 + /CJ*/AZ*/Q4 + /AZ*/Q1*/Q4 + !
       /AZ*/Q1*/Q4 + /AZ*/Q1*/Q4                              !
                                                              !
FF   = /AL*XD + /AL*XD + /AL*XD + /AM*XD + /AM*XD + /AM*XD    !



DESCRIPTION: THIS IS A SAMPLE PAL DESIGN SPECIFICATIONS PROBLEM.
    ^        FUNCTION TABLE OR OTHER FORMS OF INFORMATION COULD BE
    !        PROVIDED HERE.
    !
    !
     --------- PALASM STOPS COMPILING AT FIRST UNDEFINED SYMBOL
```

Prerequisites

Any configuration STARPLEX/ STARPLEX II Development System with a PROM Programmer Interface (SPM-A02) package installed.

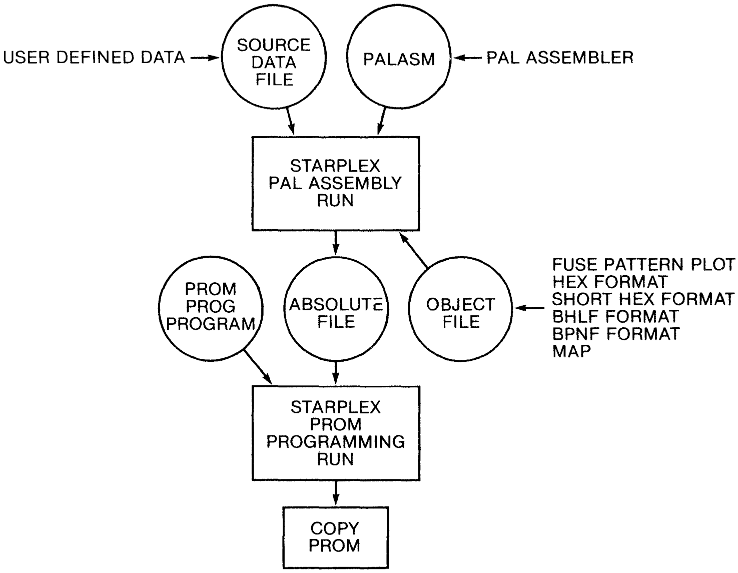


Figure 1. System Flow Chart for Programming PROM/PAL PROMs

Order Information

SPM-A02	PROM Programmer Interface Package (includes Universal PROM programmer software and PALASM — SFW-A200)
SFW-A200	Universal PROM Programmer Software (includes PALASM)

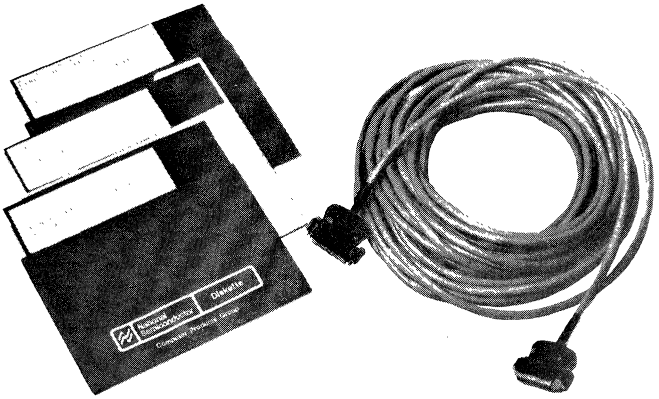
Documentation

420305788-001	STARPLEX Software Reference Manual
420306183-001	STARPLEX PROM Programmer User's Manual
	PAL Family Data Handbook *

*Number not available at the time of printing.

National Semiconductor

STARLINK™
STARPLEX II™-to-MDS Comlink



- **Permits communication between STARPLEX II and INTEL MDS230 or MDS800 systems**
- **Simple operating procedures**
- **No changes required to either system**
- **All necessary hardware and software included**

Product Overview

STARLINK is a serial link between a STARPLEX II and an INTEL MDS230 or 800 development system. The link provides the capability of transmitting or receiving data files over a 50-foot cable connected to the systems' RS232 ports. Because the MDS800 does not provide an extra serial port, an I/O expansion board such as the BLC-517 is required for operation of STARLINK.

The STARLINK kit consists of the cable with connectors and three diskettes containing software necessary for operation of the link. Included are a STARPLEX II diskette and two MDS diskettes (one single density and one double density). For MDS800 users, a separate kit is available (Part No. AEE-A002, AEE-90-A002) which includes the BLC-517 I/O expansion board.

The file transfer procedure is as follows:

- Enter the command line for the applicable MDS system into the MDS keyboard.

For MDS230 enter — :fx:XFER MDS230 followed by a carriage return.

For the MDS800 enter — :fx:XFER MDS800 followed by a carriage return.

Note: x refers to the disk drive.

- Enter the name FDSx:XFER into the STARPLEX keyboard and then press the RETURN key. The STARPLEX II transfer program then prompts with an asterisk "*" to indicate that commands may now be entered.
- When the asterisk appears on the screen, enter one of the appropriate commands listed below.
- The END key is depressed to exit from the Transfer program. Data is then saved and files are closed in an orderly fashion.
- The HELP key enables the user to review the list of applicable STARLINK commands. The HELP key is a feature of the STARPLEX II Development System which is available to the development engineer at all stages of his program development.

Once the MDS transfer program is loaded on the MDS system, all commands to send or receive files are issued from the STARPLEX system. No other interaction is required with the MDS system. The form of commands that are issued on the STARPLEX system are as follows:

```
SEND<STARPLEX file>[[TO]MDS
file][$Delete]
RECEIVE<STARPLEX file>[[FROM]MDS
file][$Delete]
```

Where,

STARPLEX file — is the name of the file to be sent or received by the STARPLEX II system.

TO MDS file — is the optional parameter that specifies the name of the MDS file that is sent to the MDS system from the STARPLEX II system. If this filename is not specified, the STARPLEX filename is used with the device code changed to Fx: from FDSx:

FROM MDS file — is an optional parameter that specifies the name of the file that is received from the MDS system. If this filename is not specified, the STARPLEX II filename is used and changed to FDSx: from Fx:

$D or $ DELETE — If the filename to be sent or received already exists, the existing file is deleted prior to transfer. If $D is specified, the file is automatically deleted. If $D is not specified, the system prompts the user for permission to delete the existing file with the following message:
Delete filename? (Y or N)

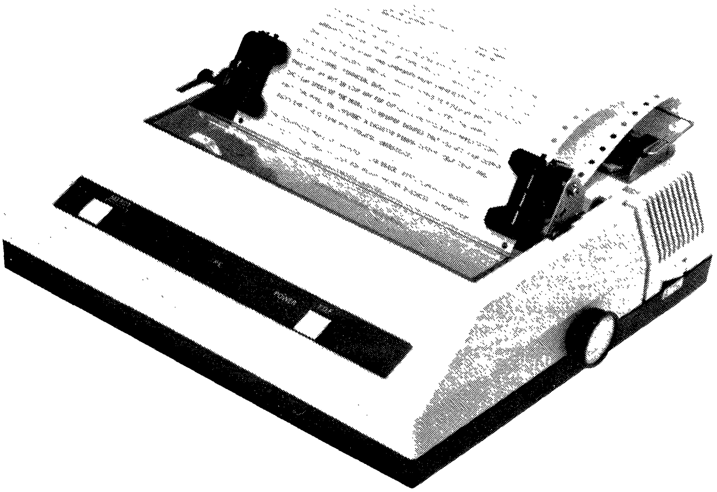
All data transmissions include a checksum and retransmissions are performed automatically. Fatal systems errors are reported to the user with descriptive information so that error recovery is simplified.

Order Information

Part Number	Description
For use under STARPLEX:	
AEE-A001	STARLINK — SPX/MDS-220/230
AEE-A002	STARLINK — SPX/MDS-800/888 Link with BLC-517 I/O Expansion Board
For use under STARPLEX II:	
AEE-90-A001	STARLINK — SPX II/MDS-200/230 Link
AEE-90-A002	STARLINK — SPX II/MDS-800/888 Link with BLC-517 I/O Expansion Board

National Semiconductor

Impact Printer



- Compatible with STARPLEX™ and STARPLEX II™ Development Systems
- 150 CPS at 10 CPI
- 100% duty cycle
- 9 × 7 dot matrix
- Bidirectional, logic seeking
- Uses standard cut-sheet, fan-fold paper
- 40, 80, and 132 column format
- Light, compact, rugged

Product Overview

The Centronics Model 150 Impact Printer is a light, compact, versatile, and rugged printer designed for most applications where size is a consideration. It is capable of 100% duty cycle applications, thus making it a highly efficient and high-speed printer. Its snap-on tractors, top-of-form feature, and condensed print capability allow the printer to be able to print out a variety of computer output such as computer program listings, business forms, financial data and labels. The printer includes a cassette ribbon system, self-test, and paper-empty detection for operator convenience.

Specifications

Operator Control/ Indicators	Power on/off switch Select switch Select light Paper empty light Power light
Data Input	7-Bit ASCII parallel, TTL levels with strobe, acknowledge, busy 8th bit selects second character set. 768 character buffer. Remote select/deselect.

Temperature	Operating: 5°C (40°F) to 40°C (100°F) Storage: −20°C (−28°F) to 71°C (160°F)
Humidity	Operating: 10% to 80% Storage: 5% to 90% (No condensation)
Electrical Requirements	60 Hz, 115 V_{AC}, +10% −15% of nominal 50 Hz, 230 V_{AC}, +10% −15% of nominal
Physical Dimensions	(Dimensions exclusive of roll paper holder) Weight 22.2 lbs. (10 kg.) Height 5.9 in. (15 cm.) Width 14.9 in. (38 cm.) Depth 13.75 in. (35 cm.)
Standard Features	150 CPS at 10 CPI 5, 8.18, 10, 16.36 CPI 9 x 7 dot matrix Bidirectional, logic seeking Cassette ribbon Adjustable snap-on tractors Paper-empty detection Variable top-of-form 96-character ASCII
Standard Features (Cont.)	Full 1-line buffer Paper tear bar Self-test Auto line feed

Prerequisites

- STARPLEX Development System Operating System Software Rev. H or later.
- STARPLEX II Development System Operating System Rev F or later.

Order Information

For STARPLEX Development Systems:

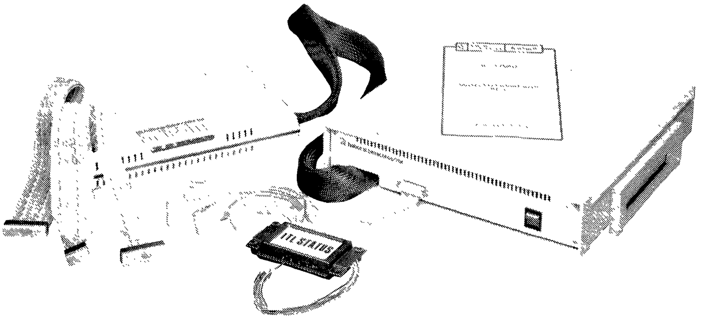
SPM-A60	Impact Printer (Centronics 150)

For STARPLEX II Development Systems:

SPM-90-A60	Impact Printer (Centronics 150)

National Semiconductor

NS16032 In-System Emulator (ISE/16™)



- **Operation up to 6 MHz**
- **Emulation of NS16032 Central Processing Unit, NS16082 Memory Management Unit, NS16201 Timing Control Unit**
- **Host resident high-level language and assembly language symbolic debugger**
- **Generalized event driven system**
- **Memory mapping, up to 30 K bytes**
- **Write protection/detection of 2 K byte memory blocks**
- **Program flow tracing, up to 255 non-sequential fetches**
- **Complete bus activity trace**
- **Qualified tracing**
- **Pre-, post-, or center-triggering on trace**
- **Count-down event counter**
- **Count-up execution timer/counter**
- **Supports Memory Management Unit functions**
- **Runs on VAX/11 (VMS) and STARPLEX II™ hosts**
- **Hierarchical help facility (on-line manual)**
- **Self-diagnostic**

Description

The NS16032 In-System Emulator (ISE/16) is a powerful tool for both hardware and software development of NS16032 microprocessor-based products.

When used with a host system such as VAX (VMS) or STARPLEX II Development Systems, ISE/16 emulates a complete NS16000™ chip set. This chip set includes the 16032 Central Processing Unit (CPU), the 16082 Memory Management Unit (MMU), and the 16201 Timing Control Unit (TCU). ISE/16 allows users to test and debug both hardware and software in their own hardware environment. ISE/16 operates in either of two modes: emulation mode, when ISE/16 is actually running the user's program, or monitor mode, when ISE/16 is communicating with the user via the host system.

ISE/16 is a complete unit, including an internal clock oscillator and 30K bytes of dedicated user's ISE™ memory. With ISE/16, users can easily stop emulation and examine the contents of CPU registers, slave processor registers, and memory.

ISE/16 consists of the ISE hardware, the ISE monitor, a host-dependent debugger (IDBG16), an RS232 serial port cable and manual.

ISE/16 hardware is the circuitry required for emulation of a user's target system. It interfaces to the host system with an RS232-compatible serial link and provides a second RS232 port for an optional terminal connection. The ISE/16 hardware also has three target cables for connections to the target system. The target cables plug into the target system CPU, MMU, and TCU sockets.

The ISE monitor is the ISE hardware control program that monitors the host system serial data link. The ISE monitor receives monitor commands from the host system, acknowledges these commands, and generates the appropriate responses. The ISE monitor also controls the target system emulation program.

IDBG16 is the interactive debugger program for ISE/16. It runs on the host system and makes the host system facilities available to the ISE/16 user. IDBG16 automatically translates commands entered at a host system terminal to the equivalent ISE monitor commands, and communicates with the ISE monitor via the serial data link.

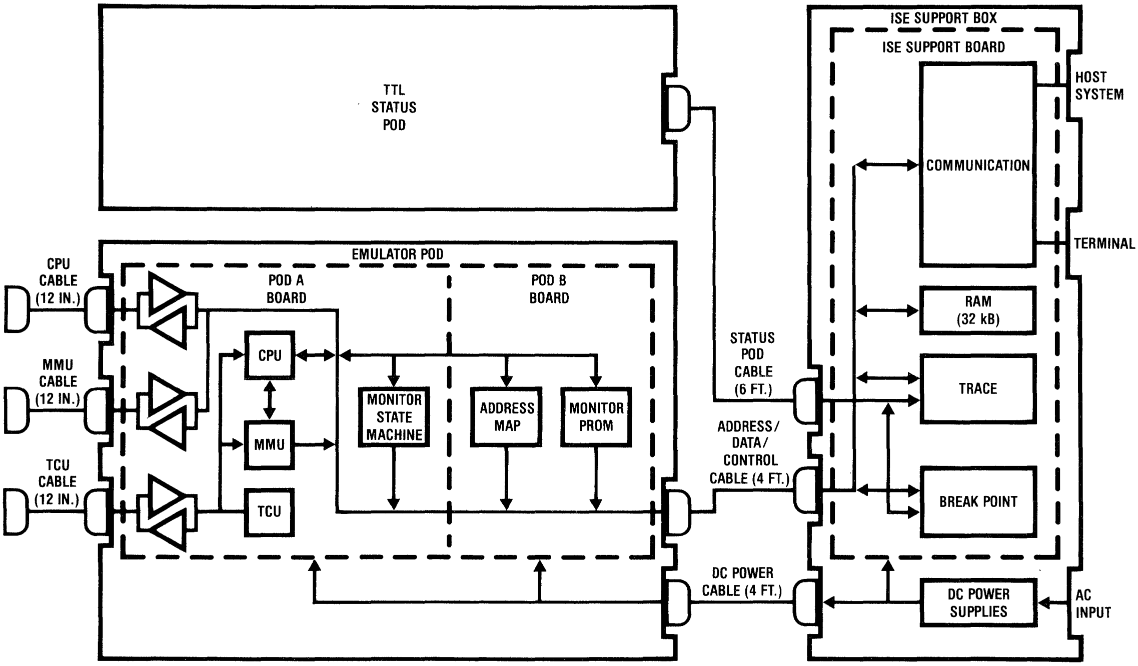
Hardware Description

The ISE/16 hardware is housed in three enclosures: the ISE Support Box, the Emulator Pod, and the TTL Status Pod. Figure 1 is a block diagram of ISE/16 hardware. The ISE/16 enclosures are described in the following paragraphs.

The ISE Support Box is the largest enclosure. It contains the emulation support circuits for trace, breakpoints, and mapped memory. It also contains power supplies and the hardware for the RS232 serial ports.

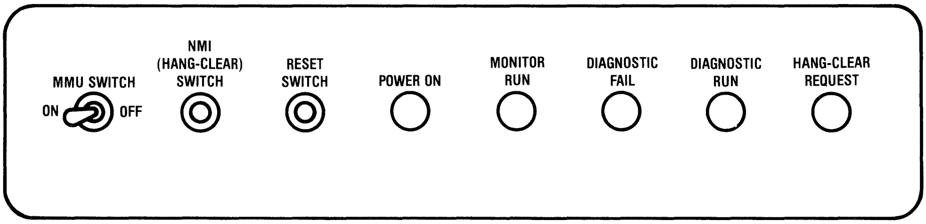
The Emulator Pod contains the 16032 CPU, 16082 MMU, and 16201 TCU required for target system emulation. It also contains the ISE Monitor firmware and houses the ISE/16 controls and indicators. Figure 2 shows the location of the ISE/16 controls and indicators. Table 1 lists the function of each switch and LED.

The Emulator Pod connects to the ISE Support Box through a 4-foot twisted-pair cable assembly. Connec-



TL/R5127-1

FIGURE 1. ISE/16 Block Diagram



TL/R5127-2

FIGURE 2. ISE/16 Controls and Indicators

tions to the target system are made with 12-inch target cables. One target cable is provided for each member of the 16000 chip set. (CPU, MMU, and TCU).

The Status Pod is the smallest enclosure. It provides TTL-compatible input and output signals for use during ISE operation. The Status Pod has eleven leads and seven binder posts that can be connected to either the target system or test equipment such as logic analyzers or oscilloscopes. Table 2 lists the function of each lead and post on the Status Pod. The Status Pod connects to the ISE Box front panel status connector via a 6-foot cable.

ISE/16 Software Overview

The ISE/16 software consists of two modules; the ISE monitor, residing in firmware on the Emulator Pod, and the ISE Debugger (IDBG16), residing in the host system. The monitor controls the ISE hardware. IDBG16, a high-level language debugger program, drives the ISE/16 unit. IDBG16 runs on the host computer and it communicates with the ISE/16 unit. Optionally, IDBG16 can also communicate with a terminal connected to ISE/16. The ISE/16 unit communicates with the IDBG16 program (running on the host) only while the ISE/16 unit is running the monitor program (in monitor mode), not while it is running the user's program (in emulation mode).

IDBG16 software is available for VAX/11 (VMS) and STARPLEX II hosts.

IDBG16, The ISE/16 Debugger

IDBG16 is user compatible with the standard non-ISE NS16000 Cross-Software Debugger, DBG16. Compatibility minimizes learning time for users of the various development tools. IDBG16 fully supports all the power debugging and emulation facilities provided by

Table 1. ISE/16 Control and Indicator Functions

Control/Indicator	Function
MMU Switch	When on, it enables MMU operation (Mbit in CPU Configuration Register set to 1). When off, disables MMU (Mbit set to 0).
NMI Switch	When pressed, <HANG-CLEAR> occurs. <HANG-CLEAR> restores control to ISE monitor.
RESET Switch	When pressed, resets the ISE hardware.
POWER ON	Indicates power to ISE.
MONITOR RUN	Indicates ISE monitor is running.
DIAGNOSTIC RUN	Indicates ISE diagnostics are running.
DIAGNOSTIC FAIL	Indicates failure during diagnostic tests.
HANG-CLEAR REQUEST	Indicates CPU has stopped executing instructions.

Table 2. Status Pod Signal Description

Status Pod Label	ISE Function
1-WHT-USRCLK-U	IS0 (input sync 0)
2-BLK-GND	Common Ground
3-BRN-EXTO-U	EXT0 (external input 0)
4-RED-EXT1	EXT1 (external input 1)
5-ORN-EXT2	EXT2 (external input 2)
6-YEL-EXT3	EXT3 (external input 3)
7-GRN-EXT4	EXT4 (external input 4)
8-BLU-EXT5	EXT5 (external input 5)
9-VIO-EXT6	EXT6 (external input 6)
10-GRY-EXT7	EXT7 (external input 7)
11-WHT-USEBRK/U	IS1 (input sync 1)
TBRUN	Multi-Processor Sync
BK SYNCH/-U	DO (output sync)
TR SYNCH/-U	TO (trace sync)
GND	Common Ground
TSYNC31/	Not Used
TSYNC21	Not Used
GND	Common Ground

the ISE/16 hardware, and supplements these features with a very powerful software-based program debugging environment.

The basic debugging features of IDBG16 are:

(1) Both high-level and assembly languages are supported.

(2) Breakpoints can be set at the source code level, even when using high-level languages.

(3) Variables can be accessed by their source code names, i.e., IDBG16 is symbolic in nature.

(4) Procedure parameters and local variables are easily displayed.

(5) Structured data types and pointers are easily displayed.

(6) Both command and history files are supported.

(7) Memory can be displayed in many different ways, including a disassembly mode displaying memory as 16032 instructions.

(8) All the emulation and debug facilities provided by the ISE/16 hardware are supported.

The ISE Monitor

When the ISE/16 unit is not running an emulation program, it is running a program called the ISE monitor. The monitor communicates with IDBG16 and it provides a command protocol that allows the host complete control of the ISE/16 hardware.

The monitor is invisible to the user, who normally communicates with the system via the friendly IDBG16 program.

Optional Terminal Feature

As an option, the ISE monitor communciates with a terminal connected to the ISE/16 unit. This terminal also serves as a terminal for the host. Thus the ISE/16 unit and the user's terminal share one RS232 port from the host.

Operation with the optional terminal is called Transparent Mode; operation without the optional terminal is called Stand-Aside Mode.

Conversion Kit for NS16008 In-System Emulation (Available December 1983)

An optional conversion kit is available for those who wish to do NS16008 development work. Contained in this kit are the following: ISE/08™ Emulator Pod, ISE/08 Symbolic Debugger (IDBG08), ISE/08 Monitor Firmware, and ISE/08 Manual. Thus, because the ISE Support Box can be used for either ISE/16 or ISE/08 development work, a user wishing to do NS16008 development work but who already has an ISE/16 unit can purchase this conversion kit (in comparison to the purchase of an entire ISE/08 unit).

ISE/16 Operation

Human Interface

ISE/16 is easy to learn and easy to use. The software includes a complete on-line manual. Invoking the "HELP" command gives a summary of all ISE/16 commands, an individual command, or an individual command's parameters. This feature helps the user get his work done quickly with less frustration.

Real-Time Emulation

The ISE/16 unit has its own CPU, MMU, and TCU components. These components are connected to the target system via cables, and they perform the same functions, with close to the same timing characteristics as they would if mounted in the target system. ISE/16 does not add wait states in its operation.

Emulation memory, resident in ISE/16, can be used in lieu of target system memory. This feature is implemented by the mapping capabilities. ISE/16 can run and debug programs, without a working target system.

User target memory address space (whether it exists or not) can be mapped onto the ISE/16 emulation memory. A memory read or write operation to an address mapped onto emulation memory is performed on emulation memory only and not on the target system's memory.

Memory from the entire 24-bit physical address space of the CPU or MMU can be mapped onto emulation memory if the following restrictions are observed:

(1) Up to four, non-contiguous segments can be defined.

(2) The address range mapped by a segment must lie within an integral 128K byte division of the address space, e.g. 00000 to h'1FFFF, or h'20000 to h'3FFFF.

(3) The address range mapped by a segment must start at the beginning of an integral 2K byte division of the address space, and end at the end of one such division e.g. h'00 to h'FFF, or h'2800 to h'37FF.

(4) The total memory space mapped by all segments must not exceed 30K bytes.

Associated with the emulation memory mapping scheme is a capability for write protection/detection. Any 2K byte block within any of the four 128K byte segments selected can be protected. A write operation to a protected memory segment causes an IM (Illegal Map) event to occur. Write operations to protected memory are inhibited only if they occur on emulation memory. They are not inhibited if they occur in target system memory.

Related commands:

MC — Map Create
MP — Map Print

Note: For the syntax of these, and other commands listed in this section, refer to the IDBG16 Command Summary.

Generalized Events

To provide a versatile way of observing and controlling the significant state changes on the microprocessor, ISE/16 allows the use and definition of "events". In general, a simple event is a breakpoint, a bus change, or a significant observation. An event can also be a logical combination of simple events (an Event-Expression).

Simple Event Definition

The simple events are:

- Breakpoints
- Latched Breakpoints
- Counter Done
- Status Pod Inputs
- Illegal Map
- Trace Done

Breakpoint Events

ISE/16 provides three common breakpoint events, named A, B, and C. The breakpoint event can be used in two ways:

(1) Execution Breakpoint — occurs just prior to execution of an instruction fetched from a specified address.

(2) Memory Reference Breakpoint — occurs on a match when sampling:

- Address Bits
- Data Bits
- External Status Bits
- User/Supervisor Pin
- High Byte Enable Pin
- Data Direction Pin
- And where any of the above options or bits can be masked.

Either virtual or physical addresses can be sampled.

ISE/16 also provides a range breakpoint event, R. The range breakpoint occurs on any read or write operation to an address in a specified address range.

All breakpoints can cause emulation to stop immediately. Also, if used with the No Stop (/NS) option, breakpoints can be combined with other events to cause a variety of action.

Latched Breakpoint Events, Counted Events

Latched breakpoint events, named LA, LB, LC, occur at some time after a cycle where the corresponding breakpoint event (A, B, or C) has taken place. The occurrence of a latched breakpoint event remains asserted until cleared.

Events, instructions, memory cycles, and clock cycles can be counted with the breakpoint counter (up to 12 bits). Upon reaching a certain count provided by the Define Counter (DC) command, the Counter Done (CD) event takes place.

Other Simple Events

The other simple events available are:

(1) IS0, IS1 — Status Pod Input Sync 0 and Input Sync 1.

(2) IM — Write operation to write-protected address.

(3) TD — End of trace.

Related commands:

BC — Breakpoint Create
BD — Breakpoint Delete
DP — Breakpoint Print

Event-Expressions

An event-expression is a Boolean expression made up of simple events, i.e., a logical combination of simple events. This allows the user to generate many different event combinations, tailored to system activity of particular interest to the user. These generalized events are used by many ISE/16 commands such as stop, trace, event counting, etc. Event-expressions provide creative and flexible debugging procedures.

Event-expressions can be evaluated as either logically true or logically false. Valid logic operations for event-expressions are: Negation (NOT), AND, and OR.

Stopping Execution on Events

A common debugging activity is to stop emulation on the occurrence of an event of interest. Stopping emulation puts ISE/16 in the monitor mode so the user can examine and alter the state of the CPU, memory, and ISE/16 functions. Emulation can be stopped on either simple events or event-expressions.

Related commands:

DS — Define Stop
BS — Breakpoint Create

Flexible Tracing

ISE/16 maintains a 255-entry trace memory. Trace memory captures bus activity in one of two trace modes. The trace modes are:

- Program Flow Trace
- Memory Bus Trace

Program Flow Trace

The Program Flow Trace mode captures the CPU Program Counter address of 255 non-sequential instructions. This mode also maintains a count of sequential instructions executed between each non-sequential instruction stored in the trace memory.

Memory Bus Trace

The Memory Bus Trace mode captures a summary of system parameters during 255 memory bus access cycles. The following parameters are captured:

- Address bus contents
- Data bus contents
- CPU Status (data transfer, non-sequential fetch, interrupt acknowledge, etc.)
- Status Pod External Inputs
- States of the Following CPU Pins:
  PFSC — Program Flow Status (start of instruction)
  UNS — User/Not Supervisor
  NHBE — Not High Byte Enable
  NDDIN — Not Data In

A tracing event can qualify the memory bus tracing mode. This event allows the user to reduce the number of events captured.

When enabled, tracing in either mode continues until a specified terminating event occurs. The actual end of tracing can be delayed after the terminating event by a count of 1 to 255. This allows trace data to be captured before, after, or around the terminating event.

Execution Timer

The execution timer is a 24-bit counter with an overflow flag that may be used to count events, instruction cycles, memory cycles, or clock cycles. The timer may be programmed to start and stop counting on specific events. This permits using the execution timer to determine relative timing differences between various events. One use of this feature is to measure software or hardware performance.

Related Commands:

DE — Define Execution Timer

LD — List Definitions

Event Trigger for External Test Equipment

ISE/16 events can trigger external test equipment, such as oscilloscopes and logic analyzers. This test equipment can be used in conjunction with ISE/16 debugging features to solve system timing problems. Two external trigger sources are provided:

- General Event (or Event-Expression)
- Trace Trigger Event (i.e., an event that causes an entry into trace memory.)

The external trigger signals are available at two status pod outputs:

- BKSYNCH/-U (General Event)
- TRSYNCH/-U (Trace Trigger Event)

Related Commands:

DO — Define Output Sync Command

ISE/16 Timing Options

ISE/16 includes the following timing options:

- Sampling time can be set to sample either virtual or physical addresses
- Status Pod external lines can be sampled at either data valid or address valid times
- The emulation clock frequency can be set to one of the following frequencies:

  1.5 MHz
  3.0 MHz
  6.0 MHz
  Target Board Frequency

Note: Selection of target board frequency may require synchronous and/or asynchronous delay compensation. Refer to ISE/16 User's Manual, Chapter 6, for details.

Related Commands:

SO — Select Options

Self-Test Diagnostics

At power-up or reset, ISE/16 runs a diagnostic program to verify ISE software integrity and proper hardware function.

Required User-Supplied Equipment

For use under VAX/11 systems:

- Valid DEC VAX/11 configuration, with available RS232 port.
- VMS Operating System, Version 3.0 or later.
- NSX-16 Cross Software Package, or NS-ASM-16 NS16000 Cross-Assembler Package.

For use with STARPLEX II systems:

- STARPLEX II Development System.
- STARPLEX II Operating System, Version G or later.
- SFW-90-A010 NS16000 Cross-Assembler Package.

For use with a system that has a Berkeley 4.1 based UNIX™ Operating System:
[Contact Marketing for Availability Information.]

- Valid computer system with an available RS232 port.
- Appropriate cross software package.
  [Contact Marketing for further information.]

Specifications

Environmental	Operating Temperature +10°C to +40°C Storage Temperature −20°C to +65°C
Power	3A @ 115 V_{AC}, 50/60 Hz, single phase 1.5A @ 220 V_{AC}, 50/60 Hz, single phase Approximately 1170 BTU.
Physical	
ISE Support Box —	Height: 4.125 in. (10.5 cm) Width: 19.0 in. (48.3 cm) Depth: 17.5 in. (44.5 cm)
Emulation Pod —	Height: 2.25 in. (6.4 cm) Width: 9.25 in. (23.5 cm) Depth: 14.0 in. (35.6 cm)
TTL Status Pod —	Height: 1.0 in. (2.5 cm) Width: 3.125 in. (7.9 cm) Depth: 6.125 in. (15.6 cm)
Cable Lengths —	ISE Support Box to Emulation Pod: 4.0 ft. (1.22 M) ISE Support Box to TTL Status Pod: 6.0 ft. (1.83 M) Emulation Pod to Target Board: 1.0 ft. (0.30 M)

Electrical

Operating Frequency —	User selectable to one of the following: 1.5 MHz 3.0 MHz 6.0 MHz Target Board Frequency **Note:** Selection of target board frequency may require synchronous and/or asynchronous delay compensation. Refer to ISE/16 User's Manual, Chapter 6, for details.
Target Interface Electrical Characteristics —	See Tables 3 through 5.

Order Information

Complete ISE/16 Units

NS-ISE-16	ISE/16 (NS16032), 115 V_{AC} for VAX/11 (VMS) Computer System.
SPM-90-A1632	ISE/16 (NS16032), 115 V_{AC} for STARPLEX II Development Systems.
NS-SYS-2004	ISE/16 (NS16032), 115 V_{AC} for UNIX OS based operating systems. [Contact Marketing for Availability Information.]

Conversion Kits to Allow for ISE/08 Emulation

[Contain ISE/08 Emulator Pod, ISE Debugger (IDBG08), appropriate ISE/08 monitor firmware, and ISE/08 manual.]

AEE-90-A1608	ISE/16 to ISE/08 kit for STARPLEX II use.
AEE-ISE-08	ISE/16 to ISE/08 kit for VAX/11 (VMS) use.
AEE-ISENIX-08	ISE/16 to ISE/08 kit for UNIX based OS systems use. [Contact Marketing for Availability Information.]

Documentation

420306675-002	ISE/16 User's Manual (Included with NS-ISE-16, and SPM-90-A1632.)

Table 3. Electrical Characteristics for TCU Interface

Signal Name	Interface Device	Input And/Or Output Current		Propagation Delay Time T_{pd}*
		I_{OH}	I_{OL}	
OUTGOING SIGNALS:				
NTSO	74S244	15mA	64mA	14.6ns
CTTL	74S244	15mA	64mA	14.6ns
FCLK	74S244	15mA	64mA	14.6ns
NDBE	74S244	15mA	64mA	14.6ns
NRD	74S244	15mA	64mA	14.6ns
NWR	74S244	15mA	64mA	14.6ns
NRST	74S244	15mA	64mA	14.6ns
RDY	74S244	15mA	64mA	14.6ns
		I_{IH}	I_{IL}	
INCOMING SIGNALS:				
NPER	74S244	50μA	400μA	14.6ns
NCWAIT	74S244	50μA	400μA	14.6ns
NWAIT1	74S244	50μA	400μA	14.6ns
NWAIT2	74S244	50μA	400μA	14.6ns
NWAIT3	74S244	50μA	400μA	14.6ns
NWAIT4	74S244	50μA	400μA	14.6ns
XCTL1	74S244	50μA	400μA	14.6ns
NCEN	74S244	50μA	400μA	14.6ns
NRST1	74S244	50μA	400μA	14.6ns

*Interface device, plus cable.

Table 4. Electrical Characteristics for MMU Interface

Signal Name	Interface Device	Input And/Or Output Current		Propagation Delay Time T_{pd}*
		I_{OH}	I_{OL}	
OUTGOING SIGNALS:				
A24	74S244	15mA	64mA	14.6ns
MMUMINT	74S244	15mA	64mA	14.6ns
NPAV	74S244	15mA	64mA	14.6ns
NABT	74S244	15mA	64mA	14.6ns
NFLT	74S244	15mA	64mA	14.6ns
NHLDA0	74S244	15mA	64mA	14.6ns
		I_{IH}	I_{IL}	
INCOMING SIGNALS:				
NHOLD	74S244	50μA	400μA	14.6ns

*Interface device, plus cable.

Table 5. Electrical Characteristics for CPU Interface

Signal Name	Interface Device	Input And/Or Output Current				Propagation Delay TimeT_{pd}*
		I_{OH}	I_{OL}	I_{IH}	I_{IL}	
BIDIRECTIONAL SIGNALS:						
NSPC	none	—	—	—	—	1.4ns
AD15	8T28	10mA	48mA	25μA	200μA	18.4ns
AD14	8T28	10mA	48mA	25μA	200μA	18.4ns
AD13	8T28	10mA	48mA	25μA	200μA	18.4ns
AD12	8T28	10mA	48mA	25μA	200μA	18.4ns
AD11	8T28	10mA	48mA	25μA	200μA	18.4ns
AD10	8T28	10mA	48mA	25μA	200μA	18.4ns
AD09	8T28	10mA	48mA	25μA	200μA	18.4ns
AD08	8T28	10mA	48mA	25μA	200μA	18.4ns
AD07	8T28	10mA	48mA	25μA	200μA	18.4ns
AD06	8T28	10mA	48mA	25μA	200μA	18.4ns
AD05	8T28	10mA	48mA	25μA	200μA	18.4ns
AD04	8T28	10mA	48mA	25μA	200μA	18.4ns
AD03	8T28	10mA	48mA	25μA	200μA	18.4ns
AD02	8T28	10mA	48mA	25μA	200μA	18.4ns
AD01	8T28	10mA	48mA	25μA	200μA	18.4ns
AD00	8T28	10mA	48mA	25μA	200μA	18.4ns
OUTGOING SIGNALS:						
A23	74S244	15mA	64mA	—	—	14.6ns
NILO	74S244	15mA	64mA	—	—	14.6ns
ST0	74S244	15mA	64mA	—	—	14.6ns
ST1	74S244	15mA	64mA	—	—	14.6ns
ST2	74S244	15mA	64mA	—	—	14.6ns
ST3	74S244	15mA	64mA	—	—	14.6ns
NPFS	74S244	15mA	64mA	—	—	14.6ns
NDDIN	74S244	15mA	64mA	—	—	14.6ns
NADS	74S244	15mA	64mA	—	—	14.6ns
UNS	74S244	15mA	64mA	—	—	14.6ns
NHBE	74S244	15mA	64mA	—	—	14.6ns
HHLDA	74S244	15mA	64mA	—	—	14.6ns
A22	74S244	15mA	64mA	—	—	14.6ns
A21	74S244	15mA	64mA	—	—	14.6ns
A20	74S244	15mA	64mA	—	—	14.6ns
A19	74S244	15mA	64mA	—	—	14.6ns
A18	74S244	15mA	64mA	—	—	14.6ns
A17	74S244	15mA	64mA	—	—	14.6ns
A16	74S244	15mA	64mA	—	—	14.6ns
INCOMING SIGNALS:						
TSYSPWR	1N4002	—	—	—	—	—
NINT	74S244	—	—	50μA	400μA	14.6ns
NNMI	74S244	—	—	50μA	400μA	14.6ns
NHOLD	74S244	—	—	50μA	400μA	14.6ns

*Interface device, plus cable.

Documentation Conventions

The following documentation conventions are used in describing the IDBG16 commands and parameters. Upper-case and lower-case letters are used in these conventions; any combination of upper-case and lower-case letters may actually be used when entering commands.

UPPER-CASE letters show the command letters, parameters and options. The names must be entered exactly as shown.

Spaces and blanks have been added for readability. When actually entering commands, spaces and blanks may only appear between the command and its parameters and between the parameters and the local radix.

< > — angle brackets enclose descriptive names (in lower-case) for user-supplied parameters/options.

{ } — braces enclose more than one item out of which one, and only one, must be used. The items are separated from each other by a logical OR sign "|".

[ ] — brackets enclose optional item(s).

| — logical OR sign separates items out of which one, and only one, may be used.

... — three consecutive periods indicate optional repetition of the preceding item.

IDBG16 Command Summary

The following is a comprehensive list of the IDBG16 commands. Commands are in alphabetical order. See the ISE/16 User's Manual for a detailed description of each command.

Command	Syntax	Function
Begin	B [<file>] [/NL] [/NI] [/R] [/Z]	(if no switches) Loads the program <file> into target board memory, and initializes registers. /NL — No Load /NI — No Initialize /R — Reset /Z — Zero-Fill data areas
Breakpoint Create	BC [A,|B,|C,] <address> [/NS]	Creates execution breakpoint A,B, or C at specified <address>. /NS — No Stop
	BC [A,|B,|C,] {<address> | <mask>} <breakpoint-options> [/NS]	Creates memory reference breakpoint A, B, or C at address specified by <address> or <mask> and with specified <breakpoint-options>. /NS — No Stop
	BC R, <address-range> [/NS]	Creates range breakpoint R at specified <address-range>. /NS — No Stop
Breakpoint Delete	BD [A|B|C|R]	Deletes specified breakpoints.
Breakpoint Revive	BR [A|B|C|R]	Revives specified breakpoint.
Breakpoint Print	BP [A|B|C|R]	Prints address and conditions of specified breakpoints.
Command File	@ {<file> | <n>}	Executes command <file> or debugger string sequence beginning at <n>.
Debugger String	$ <n> = [<string>]	Sets debugger string <n> to <string>.
Define Counter	DC <n> [/B = <event-expression>] [/C = {<event-expression> |I|M|C}] DC/R	Defines set up for ISE counter. <n> — Number of counts /B — Begin event /C — Counter type /R — Reset

IDBG16 Command Summary (Continued)

<table>
<tr><th>Command</th><th>Syntax</th><th>Function</th></tr>
<tr><td>Define Execution-Timer</td><td>DE [/B =<event-expression>] [/E =< event-expression>]
[/C = {<event-expression> |I|M|C}]
DE/R</td><td>Defines set up for ISE execution timer.
/B — Begin event
/E — End event
/C — Count type
/R — Reset</td></tr>
<tr><td>Define Output Sync</td><td>DO <event-expression></td><td>Defines output sync event.</td></tr>
<tr><td>Define Stop</td><td>DS <event-expression></td><td>Defines stop event.</td></tr>
<tr><td>Define Trace</td><td>DT [/E =<event-expression> [/D =<n>]]
[/P | /M [= { <address> | <mask>}] <qualify-options>]]]</td><td>Defines the end, delay, and trace mode parameters for trace.
/E — End event
/D — Delay count
/P — Program Flow mode
/M — Memory Bus mode
/R — Reset</td></tr>
<tr><td>Disassemble</td><td>D <address-range> [/I =<n>] [/NA]</td><td>Disassemble instructions in <address-range>.
/NA — No Address
/I — Number of Instructions to be disassembled.</td></tr>
<tr><td>Go</td><td>G [/F =<address>] [[/T =] <breakpoint>]</td><td>Starts execution of the program at the current PC address of from the <address>. Execution continues until <breakpoint>.</td></tr>
<tr><td>Help</td><td>H [<string>]</td><td>Displays general help or command syntax or parameter syntax.</td></tr>
<tr><td>In</td><td>I {<address> | <register>} [<radix>]
<value1>
[<value2>]</td><td>Checks that contents of <address> or <register> are in the range specified by <value1> and <value2 >, inclusively.</td></tr>
<tr><td>List Calls</td><td>LC [<n>]</td><td>Lists first <n> entries in call chain.</td></tr>
<tr><td>List Definitions</td><td>LD [/T|/E|/C|/O|/S]</td><td>Lists current definitions.
/T — Trace definition
/E — Execution timer definition
/C — Counter definition
/O — Output sync definition
/S — Stop definition</td></tr>
<tr><td>List Files</td><td>LF [<line> [/<file>]]</td><td>Lists lines in <file>.</td></tr>
<tr><td>List Information</td><td>LI</td><td>Lists current IDBG16 status.</td></tr>
<tr><td>List Modules</td><td>LM</td><td>Lists modules in current program.</td></tr>
<tr><td>List Procedures</td><td>LP</td><td>Lists procedures in current program.</td></tr>
<tr><td>List Strings</td><td>LS</td><td>Lists current debugger string values.</td></tr>
<tr><td>List Trace</td><td>LT [<n> |*|/A|/J]</td><td>Lists nine trace entries centered around entry <n> or around the trigger point (*).
/A — All entries
/J — Jumps only</td></tr>
</table>

IDBG16 Command Summary (Continued)

Command	Syntax	Function
Map Create	MC [<address-range>] [/S = {A|B|C|D}] [/M =<n> |/NM] [/P|/NP]	Creates segment assignment and mapping for special <address-range>. /S — Segment assignment /M — Mapping to ISE block <n> /NM — No Mapping /P — Write Protection /NP — No Protection
Map Print	MP [/S = {A|B|C|D}]	Prints current mapping for specified segment.
Memory Fill	MF <address-range> [<radix>] <value>	Fills memory at <address-range> with <value>.
Memory Move	MM <address-range>,<address> [<radix>]	Moves memory from <address-range> to <address>.
Memory Search	MS <address-range> [<radix>] <value>	Searches for <value> in <address-range>.
On	O {FAIL|RESET|NMI|EXIT [({A|B|C|R|S|HC})] {@ <n> |/O}	Sets IDBG16 response on condition: @ <n> — Executes debugger string sequence on condition beginning with $ <n>, /O — Response Off, restores normal response.
Print	P [<address-range> |<register-range>] [<radix>]	Prints contents of <address-range> or <registers>.
Print Address	PA <address>	Prints absolute address and module area associated with <address>.
Protection Create	PC <address-range> [/UW|/UR|/SW|SR] [/V|/NV] [/R|/NR] [/M|/NM] [/T =<gn>] [/P]	Creates protection/translation for pages specified by <address-range>.
Potection Print	PP [<address-range>]	Prints protection level status for pages specified by <address-range>.
Quit	Q [/S]	Terminates session. /S — Save IDBG16 status in IDBG16.IND
Repeat	<cr>	Repeats previous command.
Replace	R {<address> | <register>} [/NV] [<radix>] [<value>]	Replaces contents of <address> or <register> with <value>. /NV — No Verify
Select Echo	SE [/O]	Selects echo mode. /O — Echo Off.
Select Full	SF [/O]	Selects full symbolic PC. /O — Full Off.
Select History	SH {<file> [/F] |/O}	Selects history file <file>: /F — Full history (with responses) /O — History Off
Select Link	SL <file> [,<file>] [/L]	Selects communications channel(s): /L — List communications
Select Module	SM <module>	Selects module.

IDBG16 Command Summary (Continued)

Command	Syntax	Function
Select Options	SO [/AS = {NADS|NPAV} [/XS = {D|A}] [/EC = {10|5|2.5|U}] [/MC = {10|EC}] [/L = {C|NC}] [/T = {0|1|N|A}]	Selects current ISE operation options. /AS — Address Sample time /XS — External Sample time /EC — Emulator Clock frequency /MC — Monitor Clock frequency /L — Latched Clear or No-Clear /T — Translation
Select Procedure	SP [<n> | <procedure>:]	Selects procedure.
Select Radix	SR <radix>	Select global radix.
Step	S [<gn>]	Executes <gn> machine instructions (Assembly programs) or one Pascal statement (Pascal programs). <gn> illegal in Pascal.
Step Call	SC	Executes until a call or return.
Step Down	SD	Executes one instruction inside a procedure; skips over call instructions.
Step Instruction	SI [<gn>]	Executes <gn> machine instructions.
Step Until	SU {<address> | <register>} [<radix>] <value1> [<value2>]	Executes instructions until contents of <address> or <register> within the range specified by <value1> and <value2>.
Step While	SW {<address> | <register>} [<radix>] [<value1>] [<value2>]	Executes instructions while contents of <address> or <register> are within the range specified by <value1> and <value2>.

Parameter Summary

The following is a comprehensive list of command parameters. See the ISE/16 User's Manual for a detailed description of each.

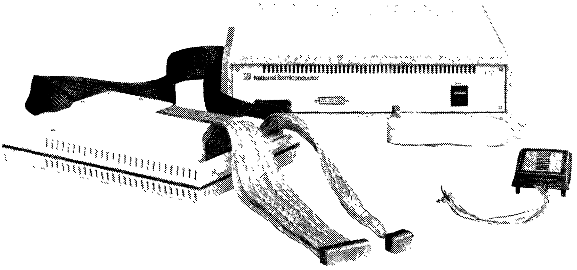
Command	Syntax	Function
Number (<n>)	<digits>	An unsigned, decimal number in the range 0 to 32767.
General Number (gn)	[H'|Q'|O'|D'] [–] <digits>	A signed, octal, decimal, or hexadecimal number in the range of -2^{31} to $2^{31}-1$.
Mask	M' {0|1|X|_}...	An unsigned number which consists of binary digits, "don't care" bits, and optional underscores. A mask represents the value of address, data, status, or external bits.
Name	<letters, numbers, underscores, tildes>	A combination of letters, digits, underscores, and tildes which does not start with a digit.
Module	<name>	The name of a module in the program.
Procedure	<name> [# <n>]	The name of a procedure in the selected module. # <n> specifies the <n>th procedure having <name> in the selected module.
Symbol	<name>	The name of a variable in the selected module or procedure.
Register	<register-name> [% <n>]	One of the registers shown in Table 3. % <n> specifies the field starting at the <n>th bit in <register>.

Parameter Summary (Continued)

<table>
<tr><th>Command</th><th>Syntax</th><th>Function</th></tr>
<tr><td>Register-Range</td><td>{•CPU__ | •MMU__ | •FPU__ | •PSR__ | •MSR__ | •FSR__ | <register>}</td><td>Specifies all CPU, MMU, or FPU registers or all PSR, MSR, and FSR fields.</td></tr>
<tr><td>Address</td><td><basic-address> [+<abs> | –<abs> | % <abs> | ∧ | • <field> | <indexing>]...</td><td>A byte or bit address. The address consists of a basic address and any combination of optional operators.
+<abs> — Adds <abs> to address.
–<abs> — Subtracts <abs> from address.
%<abs> — Takes <abs>th bit at address.
∧ — Takes contents as address.
• <field> — Address is address of a field in a Pascal record.
<indexing> — Address is address of an array element.</td></tr>
<tr><td>Address-Range</td><td>{<address1>..<address2> | < address> ! <n> | <address>}</td><td>The range of addresses from <address1> to <address2>, from <address> to <address> + (<n> – 1)*(<current radix>), or from <address> to itself.</td></tr>
<tr><td>Radix</td><td>[%] <n> <base></td><td>Specifies the length and type of input/output in IDBG16 commands. [%] specifies length. If % is specified, length is in bits. <n> must be within the range 1 to 256. <base> specifies type and may be binary (B), decimal (D), octal (O), hexadecimal (H), hexadecimal dump (H), floating-point (F), logical (L), ASCII (A), Pascal set (S), or Pascal string (G).</td></tr>
<tr><td>Value</td><td></td><td>A value to be entered or displayed after issuing a Print, Replace, Step, Memory, or In command. Syntax is defined by current radix.</td></tr>
<tr><td>File</td><td></td><td>The name of any file in the host system. File syntax is host dependent.</td></tr>
<tr><td>Event</td><td>{IS0|IS1|IM|TD|CD|A|B|C|LA|LB|LC|R}</td><td>The name of an ISE response to a specific set of run-time conditions.</td></tr>
<tr><td>Event-Expression</td><td>([′] <event> [*[′] <event>...[+ [′] <event> [*<event>]...]...])
(1)
(0)</td><td>A Boolean expression consisting of one or more <events>s and the logical NOT (′), AND (*), and OR (+) operators.
(1) — always true.
(2) — always false.</td></tr>
</table>

National Semiconductor

NS16008 In-System Emulator (ISE/08™)



- Operation up to 6MHz
- Emulation of NS16008 Central Processing Unit, NS16201 Timing Control Unit
- Host resident high-level language and assembly language symbolic debugger
- Generalized event driven system
- Memory mapping, up to 30K bytes
- Write protection/detection of 2K byte memory blocks
- Program flow tracing, up to 255 non-sequential fetches
- Complete bus activity trace
- Qualified tracing
- Pre-, post-, or center-triggering on trace
- Count-down event counter
- Count-up execution timer/counter
- Supports Memory Management Unit functions
- Runs on VAX/11 (VMS) and STARPLEX II™ hosts
- Hierarchical help facility (on-line manual)
- Self-diagnostic

Description

The NS16008 In-System Emulator (ISE/08) is a powerful tool for both hardware and software development of NS16008 microprocessor-based products.

When used with a host system such as VAX (VMS) or STARPLEX II Development Systems, ISE/08 emulates a complete NS16000™ chip set. This chip set includes the 16008 Central Processing Unit (CPU), and the 16201 Timing Control Unit (TCU). ISE/08 allows users to test and debug both hardware and software in their own hardware environment. ISE/08 operates in either of two modes: emulation mode, when ISE/08 is actually running the user's program, or monitor mode, when ISE/08 is communicating with the user via the host system.

ISE/08 is a complete unit, including an internal clock oscillator and 30K bytes of dedicated user's ISE™ memory. With ISE/08, users can easily stop emulation and examine the contents of CPU registers, slave processor registers, and memory.

ISE/08 consists of the ISE hardware, the ISE monitor, a host-dependent debugger (IDBG08), an RS232 serial port cable and manual.

ISE/08 hardware is the circuitry required for emulation of a user's target system. It interfaces to the host system with an RS232-compatible serial link and provides a second RS232 port for an optional terminal connection. The ISE/08 hardware also has two target cables for connections to the target system. The target cables plug into the target system CPU, and TCU sockets.

The ISE monitor is the ISE hardware control program that monitors the host system serial data link. The ISE monitor receives monitor commands from the host system, acknowledges these commands, and generates the appropriate responses. The ISE monitor also controls the target system emulation program.

IDBG08 is the interactive debugger program for ISE/08. It runs on the host system and makes the host system facilities available to the ISE/08 user. IDBG08 automatically translates commands entered at a host system terminal to the equivalent ISE monitor commands, and communicates with the ISE monitor via the serial data link.

Hardware Description

The ISE/08 hardware is housed in three enclosures: the ISE Support Box, the Emulator Pod, and the TTL Status Pod. Figure 1 is a block diagram of ISE/08 hardware. The ISE/08 enclosures are described in the following paragraphs.

The ISE Support Box is the largest enclosure. It contains the emulation support circuits for trace, breakpoints, and mapped memory. It also contains power supplies and the hardware for the RS232 serial ports.

The Emulator Pod contains the 16008 CPU, and 16201 TCU required for target system emulation. It also contains the ISE Monitor firmware and houses the ISE/08 controls and indicators. Figure 2 shows the location of the ISE/08 controls and indicators. Table 1 lists the function of each switch and LED.

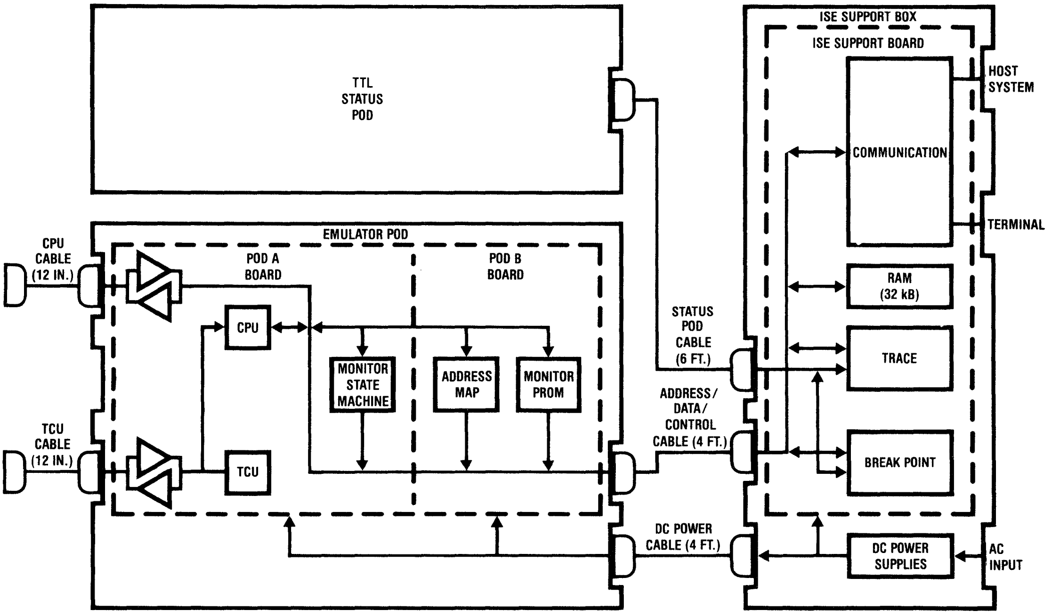
The Emulator Pod connects to the ISE Support Box through a 4-foot twisted-pair cable assembly. Connections to the target system are made with 12-inch target cables. One target cable is provided for each member of the 16000 chip set. (CP and TCU).

The Status Pod is the smallest enclosure. It provides TTL-compatible input and output signals for use during ISE operation. The Status Pod has eleven leads and seven binder posts that can be connected to either the target system or test equipment such as logic analyzers or oscilloscopes. Table 2 lists the function of each lead and post on the Status Pod. The Status Pod connects to the ISE Box front panel status connector via a 6-foot cable.

ISE/08 Software Overview

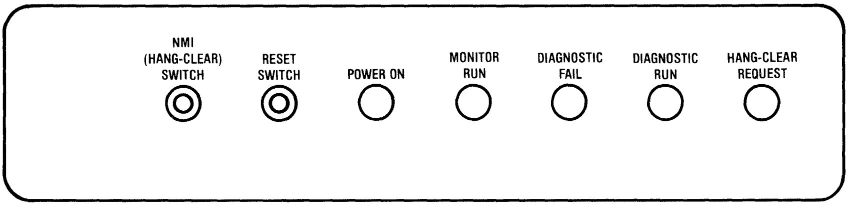
The ISE/08 software consists of two modules; the ISE monitor, residing in firmware on the Emulator Pod, and the ISE Debugger (IDBG08), residing in the host system. The monitor controls the ISE hardware. IDBG08, a high-level language debugger program, drives the ISE/08 unit. IDBG08 runs on the host computer and it communicates with the ISE/08 unit. Optionally, IDBG08 can also communicate with a terminal connected to ISE/08. The ISE/08 unit communicates with the IDBG08 program (running on the host) only while the ISE/08 unit is running the monitor program (in monitor mode), not while it is running the user's program (in emulation mode).

IDBG08 software is available for VAX/11 (VMS) and STARPLEX II hosts.



TL/R5290-1

FIGURE 1. ISE/16 Block Diagram.



TL/R5290-2

FIGURE 2. ISE/08 Controls and Indicators.

Table 1. ISE/08 Control and Indicator Functions

Control/Indicator	Function
NMI Switch	When pressed, <HANG-CLEAR> occurs. <HANG-CLEAR> restores control to ISE monitor.
RESET Switch	When pressed, resets the ISE hardware.
POWER ON	Indicates power to ISE.
MONITOR RUN	Indicates ISE monitor is running.
DIAGNOSTIC RUN	Indicates ISE diagnostics are running.
DIAGNOSTIC FAIL	Indicates failure during diagnostic tests.
HANG-CLEAR REQUEST	Indicates CPU has stopped executing instructions.

Table 2. Status Pod Signal Description

Status Pod Label	ISE Function
1-WHT-USRCLK-U	IS0 (input sync 0)
2-BLK-GND	Common Ground
3-BRN-EXTO-U	EXT0 (external input 0)
4-RED-EXT1	EXT1 (external input 1)
5-ORN-EXT2	EXT2 (external input 2)
6-YEL-EXT3	EXT3 (external input 3)
7-GRN-EXT4	EXT4 (external input 4)
8-BLU-EXT5	EXT5 (external input 5)
9-VIO-EXT6	EXT6 (external input 6)
10-GRY-EXT7	EXT7 (external input 7)
11-WHT-USEBRK/U	IS1 (input sync 1)
TBRUN	Multi-Processor Sync
BK SYNCH/-U	DO (output sync)
TR SYNCH/-U	TO (trace sync)
GND	Common Ground
TSYNC31/	Not Used
TSYNC21	Not Used
GND	Common Ground

IDBG08, The ISE/08 Debugger

IDBG08 is user compatible with the standard non-ISE NS16000 Cross-Software Debugger, IDBG08. Compatibility minimizes learning time for users of the various development tools. IDBG08 fully supports all the power debugging and emulation facilities provided by the ISE/08 hardware, and supplements these features with a very powerful software-based program debugging environment.

The basic debugging features of IDBG08 are:

(1) Both high-level and assembly languages are supported.

(2) Breakpoints can be set at the source code level, even when using high-level languages.

(3) Variables can be accessed by their source code names, i.e., IDBG08 is symbolic in nature.

(4) Procedure parameters and local variables are easily displayed.

(5) Structured data types and pointers are easily displayed.

(6) Both command and history files are supported.

(7) Memory can be displayed in many different ways, including a disassembly mode displaying memory as 16032 instructions.

(8) All the emulation and debug facilities provided by the ISE/08 hardware are supported.

The ISE Monitor

When the ISE/08 unit is not running an emulation program, it is running a program called the ISE monitor. The monitor communicates with IDBG08 and it provides a command protocol that allows the host complete control of the ISE/08 hardware.

The monitor is invisible to the user, who normally communicates with the system via the friendly IDBG08 program.

Optional Terminal Feature

As an option, the ISE monitor communciates with a terminal connected to the ISE/08 unit. This terminal also serves as a terminal for the host. Thus the ISE/08 unit and the user's terminal share one RS232 port from the host.

Operation with the optional terminal is called Transparent Mode; operation without the optional terminal is called Stand-Aside Mode.

Conversion Kit for NS16008 In-System Emulation (Available December 1983)

An optional conversion kit is available for those who wish to do NS16008 development work. Contained in this kit are the following: ISE/08 Emulator Pod, ISE/08 Symbolic Debugger (IDBG08), ISE/08 Monitor Firmware, and ISE/08 Manual. Thus, because the ISE Support Box can be used for either ISE/16 or ISE/08 development work, a user wishing to do NS16008 development work but who already has an ISE/08 unit can purchase this conversion kit (in comparison to the purchase of an entire ISE/08 unit).

ISE/08 Operation

Human Interface

ISE/08 is easy to learn and easy to use. The software includes a complete on-line manual. Invoking the "HELP" command gives a summary of all ISE/08 commands, an individual command, or an individual command's parameters. This feature helps the user get his work done quickly with less frustration.

Real-Time Emulation

The ISE/08 unit has its own CPU, and TCU components. These components are connected to the target system via cables, and they perform the same functions, with close to the same timing characteristics as they would if mounted in the target system. ISE/08 does not add wait states in its operation.

Emulation memory, resident in ISE/08, can be used in lieu of target system memory. This feature is implemented by the mapping capabilities. ISE/08 can run and debug programs, without a working target system.

User target memory address space (whether it exists or not) can be mapped onto the ISE/08 emulation memory. A memory read or write operation to an address mapped onto emulation memory is performed on emulation memory only and not on the target system's memory.

Memory from the entire 24-bit physical address space of the CPU or can be mapped onto emulation memory if the following restrictions are observed:

(1) Up to four, non-contiguous segments can be defined.

(2) The address range mapped by a segment must lie within an integral 128K byte division of the address space, e.g. 00000 to h'1FFFF, or h'20000 to h'3FFFF.

(3) The address range mapped by a segment must start at the beginning of an integral 2K byte division of the address space, and end at the end of one such division e.g. h'00 to h'FFF, or h'2800 to h'37FF.

(4) The total memory space mapped by all segments must not exceed 30K bytes.

Associated with the emulation memory mapping scheme is a capability for write protection/detection. Any 2K byte block within any of the four 128K byte segments selected can be protected. A write operation to a protected memory segment causes an IM (Illegal Map) event to occur. Write operations to protected memory are inhibited only if they occur on emulation memory. They are not inhibited if they occur in target system memory.

Related commands:

MC — Map Create
MP — Map Print

Note: For the syntax of these, and other commands listed in this section, refer to the IDBG08 Command Summary.

Generalized Events

To provide a versatile way of observing and controlling the significant state changes on the microprocessor, ISE/08 allows the use and definition of "events". In general, a simple event is a breakpoint, a bus change, or a significant observation. An event can also be a logical combination of simple events (an Event-Expression).

Simple Event Definition

The simple events are:

- Breakpoints
- Latched Breakpoints
- Counter Done
- Status Pod Inputs
- Illegal Map
- Trace Done

Breakpoint Events

ISE/08 provides three common breakpoint events, named A, B, and C. The breakpoint event can be used in two ways:

(1) Execution Breakpoint — occurs just prior to execution of an instruction fetched from a specified address.

(2) Memory Reference Breakpoint — occurs on a match when sampling:

- Address Bits
- Data Bits
- External Status Bits
- User/Supervisor Pin
- Data Direction Pin
- And where any of the above options or bits can be masked.

Either virtual or physical addresses can be sampled.

ISE/08 also provides a range breakpoint event, R. The range breakpoint occurs on any read or write operation to an address in a specified address range.

All breakpoints can cause emulation to stop immediately. Also, if used with the No Stop (/NS) option, breakpoints can be combined with other events to cause a variety of action.

Latched Breakpoint Events, Counted Events

Latched breakpoint events, named LA, LB, LC, occur at some time after a cycle where the corresponding breakpoint event (A, B, or C) has taken place. The occurrence of a latched breakpoint event remains asserted until cleared.

Events, instructions, memory cycles, and clock cycles can be counted with the breakpoint counter (up to 12 bits). Upon reaching a certain count provided by the Define Counter (DC) command, the Counter Done (CD) event takes place.

Other Simple Events

The other simple events available are:

(1) IS0, IS1 — Status Pod Input Sync 0 and Input Sync 1.

(2) IM — Write operation to write-protected address.

(3) TD — End of trace.

Related commands:

BC — Breakpoint Create
BD — Breakpoint Delete
DP — Breakpoint Print

Event-Expressions

An event-expression is a Boolean expression made up of simple events, i.e., a logical combination of simple events. This allows the user to generate many different event combinations, tailored to system activity of particular interest to the user. These generalized events are used by many ISE/08 commands such as stop, trace, event counting, etc. Event-expressions provide creative and flexible debugging procedures.

Event-expressions can be evaluated as either logically true or logicaliy false. Valid logic operations for event-expressions are: Negation (NOT), AND, and OR.

Stopping Execution on Events

A common debugging activity is to stop emulation on the occurrence of an event of interest. Stopping emulation puts ISE/08 in the monitor mode so the user can examine and alter the state of the CPU, memory, and ISE/08 functions. Emulation can be stopped on either simple events or event-expressions.

Related commands:

DS — Define Stop
BS — Breakpoint Create

Flexible Tracing

ISE/08 maintains a 255-entry trace memory. Trace memory captures bus activity in one of two trace modes. The trace modes are:

- Program Flow Trace
- Memory Bus Trace

Program Flow Trace

The Program Flow Trace mode captures the CPU Program Counter address of 255 non-sequential instructions. This mode also maintains a count of sequential instructions executed between each non-sequential instruction stored in the trace memory.

Memory Bus Trace

The Memory Bus Trace mode captures a summary of system parameters during 255 memory bus access cycles. The following parameters are captured:

- Address bus contents
- Data bus contents
- CPU Status (data transfer, non-sequential fetch, interrupt acknowledge, etc.)
- Status Pod External Inputs
- States of the Following CPU Pins:
  PFSC — Program Flow Status (start of instruction)
  UNS — User/Not Supervisor
  NDDIN — Not Data In

A tracing event can qualify the memory bus tracing mode. This event allows the user to reduce the number of events captured.

When enabled, tracing in either mode continues until a specified terminating event occurs. The actual end of tracing can be delayed after the terminating event by a count of 1 to 255. This allows trace data to be captured before, after, or around the terminating event.

Execution Timer

The execution timer is a 24-bit counter with an overflow flag that may be used to count events, instruction cycles, memory cycles, or clock cycles. The timer may be programmed to start and stop counting on specific events. This permits using the execution timer to determine relative timing differences between various events. One use of this feature is to measure software or hardware performance.

Related Commands:

DE — Define Execution Timer
LD — List Definitions

Event Trigger for External Test Equipment

ISE/08 events can trigger external test equipment, such as oscilloscopes and logic analyzers. This test equipment can be used in conjunction with ISE/08 debugging features to solve system timing problems. Two external trigger sources are provided:

- General Event (or Event-Expression)
- Trace Trigger Event (i.e., an event that causes an entry into trace memory.)

The external trigger signals are available at two status pod outputs:

- BKSYNCH/-U (General Event)
- TRSYNCH/-U (Trace Trigger Event)

Related Commands:

DO — Define Output Sync Command

ISE/08 Timing Options

ISE/08 includes the following timing options:

- Sampling time can be set to sample either virtual or physical addresses
- Status Pod external lines can be sampled at either data valid or address valid times
- The emulation clock frequency can be set to one of the following frequencies:
  1.5 MHz
  3.0 MHz
  6.0 MHz
  Target Board Frequency

Note: Selection of target board frequency may require synchronous and/or asynchronous delay compensation. Refer to ISE/08 User's Manual, Chapter 6, for details.

Related Commands:

SO — Select Options

Self-Test Diagnostics

At power-up or reset, ISE/08 runs a diagnostic program to verify ISE software integrity and proper hardware function.

Required User-Supplied Equipment

For use under VAX/11 systems:

- Valid DEC VAX/11 configuration, with available RS232 port.
- VMS Operating System, Version 3.0 or later.
- NSX-08 Cross Software Package, or NS-ASM-08 NS16000 Cross Assembler Package.

For use with STARPLEX II systems:

- STARPLEX II Development System.
- STARPLEX II Operating System, Version G or later.
- SFW-90-A010 NS16000 Cross-Assembler Package.

For use with a system that has a Berkeley 4.1 based UNIX^{TM} Operating System:
[Contact Marketing for Availability Information.]

- Valid computer system with an available RS232 port.
- Appropriate cross software package.
  [Contact Marketing for further information.]

Specifications

Environmental	Operating Temperature +10°C to +40°C Storage Temperature −20°C to +65°C
Power	3A @ 115 V_{AC}, 50/60 Hz, single phase 1.5A @ 220 V_{AC}, 50/60 Hz, single phase Approximately 1170 BTU.

Physical

ISE Support Box —	Height: 4.125 in. (10.5 cm) Width: 19.0 in. (48.3 cm) Depth: 17.5 in. (44.5 cm)
Emulation Pod —	Height: 2.25 in. (6.4 cm) Width: 9.25 in. (23.5 cm) Depth: 14.0 in. (35.6 cm)
TTL Status Pod —	Height: 1.0 in. (2.5 cm) Width: 3.125 in. (7.9 cm) Depth: 6.125 in. (15.6 cm)
Cable Lengths —	ISE Support Box to Emulation Pod: 4.0 ft. (1.22M) ISE Support Box to TTL Status Pod: 6.0 ft. (1.83M) Emulation Pod to Target Board: 1.0 ft. (0.30M)

Electrical

Operating Frequency —	User selectable to one of the following: 1.5MHz 3.0MHz 6.0MHz Target Board Frequency **Note:** Selection of target board frequency may require synchronous and/or asynchronous delay compensation. Refer to ISE/08 User's Manual, Chapter 6, for details.
Target Interface Electrical Characteristics —	See Tables 3 through 5.

Order Information

Complete ISE/08 Units

NS-ISE-08	ISE/08 (NS16008), 115 V_{AC} for VAX/11 (VMS) Computer System.
SPM-90-A1608	ISE/16 (NS16008), 115 V_{AC} for STARPLEX II Development Systems.
NS-SYS-2008	ISE/16 (NS16008), 115 V_{AC} for UNIX OS based operating systems. [Contact Marketing for Availability Information.]

Conversion Kits to Allow for ISE/16 Emulation

[Contain ISE/16 Emulator Pod, ISE Debugger (IDBG16), appropriate ISE/16 monitor firmware, and ISE/16 manual.]

AEE-90-A1632	ISE/08 to ISE/16 kit for STARPLEX II use.
AEE-ISE-16	ISE/08 to ISE/16 kit for VAX/11 (VMS) use.
AEE-ISENIX-16	ISE/08 to ISE/16 kit for UNIX based OS systems use. [Contact Marketing for Availability Information.]

Documentation

TBD	ISE/08 User's Manual (Included with NS-ISE-08, and SPM-90-A1608.)

Documentation Conventions

The following documentation conventions are used in describing the IDBG08 commands and parameters. Upper-case and lower-case letters are used in these conventions; any combination of upper-case and lower-case letters may actually be used when entering commands.

UPPER-CASE letters show the command letters, parameters and options. The names must be entered exactly as shown.

Spaces and blanks have been added for readability. When actually entering commands, spaces and blanks may only appear between the command and its parameters and between the parameters and the local radix.

< > — angle brackets enclose descriptive names (in lower-case) for user-supplied parameters/options.

{ } — braces enclose more than one item out of which one, and only one, must be used. The items are separated from each other by a logical OR sign "|".

[ ] — brackets enclose optional item(s).

| — logical OR sign separates items out of which one, and only one, may be used.

... — three consecutive periods indicate optional repetition of the preceding item.

Table 3. Electrical Characteristics for TCU Interface

Signal Name	Interface Device	Input And/Or Output Current		Propagation Delay Time T_{pd}*
		I_{OH}	I_{OL}	
OUTGOING SIGNALS:				
NTSO	74S244	15mA	64mA	14.6ns
CTTL	74S244	15mA	64mA	14.6ns
FCLK	74S244	15mA	64mA	14.6ns
NDBE	74S244	15mA	64mA	14.6ns
NRD	74S244	15mA	64mA	14.6ns
NWR	74S244	15mA	64mA	14.6ns
NRST	74S244	15mA	64mA	14.6ns
RDY	74S244	15mA	64mA	14.6ns
		I_{IH}	I_{IL}	
INCOMING SIGNALS:				
NPER	74S244	50μA	400μA	14.6ns
NCWAIT	74S244	50μA	400μA	14.6ns
NWAIT1	74S244	50μA	400μA	14.6ns
NWAIT2	74S244	50μA	400μA	14.6ns
NWAIT3	74S244	50μA	400μA	14.6ns
NWAIT4	74S244	50μA	400μA	14.6ns
XCTL1	74S244	50μA	400μA	14.6ns
NCEN	74S244	50μA	400μA	14.6ns
NRST1	74S244	50μA	400μA	14.6ns

*Interface device, plus cable.

Table 4. Electrical Characteristics for CPU Interface

Signal Name	Interface Device	Input And/Or Output Current				Propagation Delay TimeT_{pd}*
		I_{OH}	I_{OL}	I_{IH}	I_{IL}	
BIDIRECTIONAL SIGNALS:						
NSPC	none	—	—	—	—	1.4ns
A15	74S244	15mA	64mA	—	—	14.6ns
A14	74S244	15mA	64mA	—	—	14.6ns
A13	74S244	15mA	64mA	—	—	14.6ns
A12	74S244	15mA	64mA	—	—	14.6ns
A11	74S244	15mA	64mA	—	—	14.6ns
A10	74S244	15mA	64mA	—	—	14.6ns
A09	74S244	15mA	64mA	—	—	14.6ns
A08	74S244	15mA	64mA	—	—	14.6ns
AD07	74S244	15mA	64mA	—	—	14.6ns
AD06	74S244	15mA	64mA	—	—	14.6ns
AD05	74S244	15mA	64mA	—	—	14.6ns
AD04	74S244	15mA	64mA	—	—	14.6ns
AD03	74S244	15mA	64mA	—	—	14.6ns
AD02	74S244	15mA	64mA	—	—	14.6ns
AD01	74S244	15mA	64mA	—	—	14.6ns
AD00	74S244	15mA	64mA	—	—	14.6ns
OUTGOING SIGNALS:						
A23	74S244	15mA	64mA	—	—	14.6ns
NILO	74S244	15mA	64mA	—	—	14.6ns
ST0	74S244	15mA	64mA	—	—	14.6ns
ST1	74S244	15mA	64mA	—	—	14.6ns
ST2	74S244	15mA	64mA	—	—	14.6ns
ST3	74S244	15mA	64mA	—	—	14.6ns
NPFS	74S244	15mA	64mA	—	—	14.6ns
NDDIN	74S244	15mA	64mA	—	—	14.6ns
NADS	74S244	15mA	64mA	—	—	14.6ns
UNS	74S244	15mA	64mA	—	—	14.6ns
HHLDA	74S244	15mA	64mA	—	—	14.6ns
A22	74S244	15mA	64mA	—	—	14.6ns
A21	74S244	15mA	64mA	—	—	14.6ns
A20	74S244	15mA	64mA	—	—	14.6ns
A19	74S244	15mA	64mA	—	—	14.6ns
A18	74S244	15mA	64mA	—	—	14.6ns
A17	74S244	15mA	64mA	—	—	14.6ns
A16	74S244	15mA	64mA	—	—	14.6ns
INCOMING SIGNALS:						
TSYSPWR	1N4002	—	—	—	—	—
NINT	74S244	—	—	50µA	400µA	14.6ns
NNMI	74S244	—	—	50µA	400µA	14.6ns
NHOLD	74S244	—	—	50µA	400µA	14.6ns

*Interface device, plus cable.

IDBG08 Command Summary

The following is a comprehensive list of the IDBG08 commands. Commands are in alphabetical order. See the ISE/08 User's Manual for a detailed description of each command.

Command	Syntax	Function
Begin	B [<file>] [/NL] [/NI] [/R] [/Z]	(if no switches) Loads the program <file> into target board memory, and initializes registers. /NL — No Load /NI — No Initialize /R — Reset /Z — Zero-Fill data areas
Breakpoint Create	BC [A,|B,|C,] <address> [/NS]	Creates execution breakpoint A,B, or C at specified <address>. /NS — No Stop
	BC [A,|B,|C,] {<address> | <mask>} <breakpoint-options> [/NS]	Creates memory reference breakpoint A, B, or C at address specified by <address> or <mask> and with specified <breakpoint-options>. /NS — No Stop
	BC R, <address-range> [/NS]	Creates range breakpoint R at specified <address-range>. /NS — No Stop
Breakpoint Delete	BD [A|B|C|R]	Deletes specified breakpoints.
Breakpoint Revive	BR [A|B|C|R]	Revives specified breakpoint.
Breakpoint Print	BP [A|B|C|R]	Prints address and conditions of specified breakpoints.
Command File	@ {<file> | <n>}	Executes command <file> or debugger string sequence beginning at <n>.
Debugger String	$ <n> = [<string>]	Sets debugger string <n> to <string>.
Define Counter	DC <n> [/B = <event-expression>] [/C = {<event-expression> |I|M|C}]	
	DC/R	Defines set up for ISE counter. <n> — Number of counts /B — Begin event /C — Counter type /R — Reset
Define Execution-Timer	DE [/B = <event-expression>] [/E = < event-expression>] [/C = {<event-expression> |I|M|C}]	
	DE/R	Defines set up for ISE execution timer. /B — Begin event /E — End event /C — Count type /R — Reset
Define Output Sync	DO <event-expression>	Defines output sync event.
Define Stop	DS <event-expression>	Defines stop event.

IDBG08 Command Summary (Continued)

Command	Syntax	Function
Define Trace	DT [/E = <event-expression> [/D = <n>]] [/P | /M [= { <address> | <mask>}] <qualify-options>]]]	Defines the end, delay, and trace mode parameters for trace. /E — End event /D — Delay count /P — Program Flow mode /M — Memory Bus mode /R — Reset
Disassemble	D <address-range> [/I = <n>] [/NA]	Disassemble instructions in <address-range>. /NA — No Address /I — Number of Instructions to be disassembled.
Go	G [/F = <address>] [[/T =] <breakpoint>]	Starts execution of the program at the current PC address of from the <address>. Execution continues until <breakpoint>.
Help	H [<string>]	Displays general help or command syntax or parameter syntax.
In	I {<address> | <register>} [<radix>] <value1> [<value2>]	Checks that contents of <address> or <register> are in the range specified by <value1> and <value2 >, inclusively.
List Calls	LC [<n>]	Lists first <n> entries in call chain.
List Definitions	LD [/T|/E|/C|/O|/S]	Lists current definitions. /T — Trace definition /E — Execution timer definition /C — Counter definition /O — Output sync definition /S — Stop definition
List Files	LF [<line> [/<file>]]	Lists lines in <file>.
List Information	LI	Lists current IDBG16 status.
List Modules	LM	Lists modules in current program.
List Procedures	LP	Lists procedures in current program.
List Strings	LS	Lists current debugger string values.
List Trace	LT [<n> |*|/A|/J]	Lists nine trace entries centered around entry <n> or around the trigger point (*). /A — All entries /J — Jumps only
Map Create	MC [<address-range>] [/S = {A|B|C|D}] [/M = <n> |/NM] [/P|/NP]	Creates segment assignment and mapping for special <address-range>. /S — Segment assignment /M — Mapping to ISE block <n> /NM — No Mapping /P — Write Protection /NP — No Protection

IDBG08 Command Summary (Continued)

Command	Syntax	Function
Map Print	MP [/S = {A|B|C|D}]	Prints current mapping for specified segment.
Memory Fill	MF <address-range> [<radix>] <value>	Fills memory at <address-range> with <value>.
Memory Move	MM <address-range>,<address> [<radix>]	Moves memory from <address-range> to <address>.
Memory Search	MS <address-range> [<radix>] <value>	Searches for <value> in <address-range>.
On	O {FAIL|RESET|NMI|EXIT [({A|B|C|R|S|HC})] {@ <n> |/O}	Sets IDBG16 response on condition: @ <n> — Executes debugger string sequence on condition beginning with $ <n>, /O — Response Off, restores normal response.
Print	P [<address-range> |<register-range>] [<radix>]	Prints contents of <address-range> or <registers>.
Print Address	PA <address>	Prints absolute address and module area associated with <address>.
Protection Create	PC <address-range> [/UW|/UR|/SW|SR] [/V|/NV] [/R|/NR] [/M|/NM] [/T = <gn>] [/P]	Creates protection/translation for pages specified by <address-range>.
Potection Print	PP [<address-range>]	Prints protection level status for pages specified by <address-range>.
Quit	Q [/S]	Terminates session. /S — Save IDBG16 status in IDBG16.IND
Repeat	<cr>	Repeats previous command.
Replace	R {<address> | <register>} [/NV] [<radix>] [<value>]	Replaces contents of <address> or <register> with <value>. /NV — No Verify
Select Echo	SE [/O]	Selects echo mode. /O — Echo Off.
Select Full	SF [/O]	Selects full symbolic PC. /O — Full Off.
Select History	SH {<file> [/F] |/O}	Selects history file <file>: /F — Full history (with responses) /O — History Off
Select Link	SL <file> [,<file>] [/L]	Selects communications channel(s): /L — List communications
Select Module	SM <module>	Selects module.

IDBG08 Command Summary (Continued)

Command	Syntax	Function
Select Options	SO [/AS = {NADS|NPAV} [/XS = {D|A}] [/EC = {10|5|2.5|U}] [/MC = {10|EC}] [/L = {C|NC}] [/T = {0|1|N|A}]	Selects current ISE operation options. /AS — Address Sample time /XS — External Sample time /EC — Emulator Clock frequency /MC — Monitor Clock frequency /L — Latched Clear or No-Clear /T — Translation
Select Procedure	SP [<n> | <procedure>:]	Selects procedure.
Select Radix	SR <radix>	Select global radix.
Step	S [<gn>]	Executes <gn> machine instructions (Assembly programs) or one Pascal statement (Pascal programs). <gn> illegal in Pascal.
Step Call	SC	Executes until a call or return.
Step Down	SD	Executes one instruction inside a procedure; skips over call instructions.
Step Instruction	SI [<gn>]	Executes <gn> machine instructions.
Step Until	SU {<address> | <register>} [<radix>] <value1> [<value2>]	Executes instructions until contents of <address> or <register> within the range specified by <value1> and <value2>.
Step While	SW {<address> | <register>} [<radix>] [<value1>] [<value2>]	Executes instructions while contents of <address> or <register> are within the range specified by <value1> and <value2>.

Parameter Summary

The following is a comprehensive list of command parameters. See the ISE/16 User's Manual for a detailed description of each.

Command	Syntax	Function
Number (<n>)	<digits>	An unsigned, decimal number in the range 0 to 32767.
General Number (gn)	[H'|Q'|O'|D'] [–] <digits>	A signed, octal, decimal, or hexadecimal number in the range of -2^{31} to $2^{31}-1$.
Mask	M'{0|1|X|_}...	An unsigned number which consists of binary digits, "don't care" bits, and optional underscores. A mask represents the value of address, data, status, or external bits.
Name	<letters, numbers, underscores, tildes>	A combination of letters, digits, underscores, and tildes which does not start with a digit.
Module	<name>	The name of a module in the program.
Procedure	<name> [# <n>]	The name of a procedure in the selected module. # <n> specifies the <n>th procedure having <name> in the selected module.
Symbol	<name>	The name of a variable in the selected module or procedure.
Register	<register-name> [% <n>]	One of the registers shown in Table 3. % <n> specifies the field starting at the <n>th bit in <register>.
Register-Range	{•CPU_ | •MMU_ | •FPU_ | •PSR_ | •MSR_ | •FSR_ | <register>}	Specifies all CPU, MMU, or FPU registers or all PSR, MSR, and FSR fields.

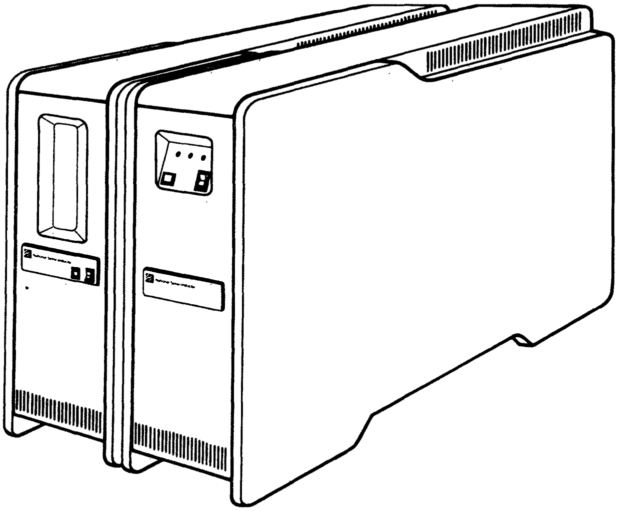
Parameter Summary (Continued)

Command	Syntax	Function
Address	<basic-address> [+<abs> | –<abs> | % <abs> | ∧ | • <field> | <indexing>]...	A byte or bit address. The address consists of a basic address and any combination of optional operators. +<abs> — Adds <abs> to address. –<abs> — Subtracts <abs> from address. %<abs> — Takes <abs>th bit at address. ∧ — Takes contents as address. • <field> — Address is address of a field in a Pascal record. <indexing> — Address is address of an array element.
Address-Range	{<address1>..<address2> | < address> ! <n> | <address>}	The range of addresses from <address1> to <address2>, from <address> to <address> + (<n> – 1)*(<current radix>), or from <address> to itself.
Radix	[%] <n> <base>	Specifies the length and type of input/output in IDBG16 commands. [%] specifies length. If % is specified, length is in bits. <n> must be within the range 1 to 256. <base> specifies type and may be binary (B), decimal (D), octal (O), hexadecimal (H), hexadecimal dump (H), floating-point (F), logical (L), ASCII (A), Pascal set (S), or Pascal string (G).
Value		A value to be entered or displayed after issuing a Print, Replace, Step, Memory, or In command. Syntax is defined by current radix.
File		The name of any file in the host system. File syntax is host dependent.
Event	{IS0|IS1|IM|TD|CD|A|B|C|LA|LB|LC|R}	The name of an ISE response to a specific set of run-time conditions.
Event-Expression	([′] <event> [*[′] <event>...[+ [′] <event> [*<event>]...]...]) (1) (0)	A Boolean expression consisting of one or more <events>s and the logical NOT (′), AND (*), and OR (+) operators. (1) — always true. (2) — always false.

National Semiconductor

PRELIMINARY
May 1983

SYS16™ Multi-User Development System for the NS16000™ Microprocessor Family



TL/F/5266-1

- **GENIX™ enhanced Berkeley 4.1 bsd UNIX operating system**
- **Time-shared support for up to eight users**
- **NS16032 Microprocessor Family based**
- **Demand-Paged Virtual Memory (DPVM) support**
- **Easy to use, proven programming environment**
- **1.25 MB RAM, expandable to 3.25 MB**
- **20 MB Hard Disk, expandable to 140 MB**
- **Streamer Tape backup, with 20 MB cartridges**
- **C and Pascal High Level Language Compilers**
- **NS16000 assembler**
- **Supports emulation of NS16000 Microprocessor Family**

Product Overview

The SYS16 is a multi-user development system which provides powerful software and hardware tools for the development of applications using National Semiconductor's NS16000 Microprocessor Family components.

Based on the NS16032 16-bit Microprocessor, SYS16 gives the designer access to an assembler, high level language compliers and real-time In-System Emulation (ISE™) tools. Total development support is provided for up to eight users, on a time-shared basis.

The SYS16 includes two main modules: the Processor module, which houses most of the electronics and the Disk-Tape module, which houses the hard disk and streamer tape back-up.

Optional disk drive modules may be added to increase system capacity. Disk drive modules contain two drives of 20 MB each.

One terminal is provided with the system. Additional terminals may be added to the system as the demand warrants. Emulation and software development work may be performed concurrently. Shared resources of the hard disk and user-supplied printer lowers the system's cost per user.

National's GENIX Operating System is an enhanced version of Berkeley 4.1 bsd UNIX. These enhancements have been added to fully utilize the advanced architecture of the NS16032 Micrprocessor Family.

SYS16™, ISE™, GENIX™ and NS16000™ are trademarks of National Semiconductor Corp.
UNIX™ is a trademark of Bell Labs

Hardware Description

Processor Module

This six-slot module houses most of the electronics for the SYS16. Standard configuration includes boards installed in four of the slots, with the remaining two available for additional Random Access Memory boards. The bus is an extended-CPU National proprietary bus designed for fast interface between the boards. The four boards provided in the standard configuration are the CPU, Serial I/O, Memory, and Disk-Tape Controller.

The CPU board is based on the NS16032 microprocessor family and includes the CPU plus the NS16082 Memory Management Unit, the NS16201 Timing Control Unit, the NS16081 Floating-Point Unit and the NS16202 Interrupt Control Unit. The CPU board also contains diagnostic firmware, one parallel I/O port, one GPIB IEEE-488 port, one RS232 port, and 256 kB of RAM.

The Serial I/O board contains logic supporting eight RS232 ports.

The memory board contains 1 MB of RAM with error checking and correction. Access time is 400 ns. Additional memory boards may be added to the system, up to a total of 3.25 MB.

The Disk-Tape Controller board contains the necessary electronics to control the disk drives and the streamer tape.

Disc-Tape Module

This module houses an 8-inch Winchester hard disk with a capacity of 20 MB (17.8 MB formatted). It also contains a ¼″ streamer tape for backup and ready access for Operating System and other software updates. The tape cartridge has a 20 MB capacity.

Disc only Module

Additional hard disk memory may be added to the system. Disk-only modules are available which house 40 MB each. A total of 3 modules may be added for a total of 140 MB.

Hardware Support:

Parallel Printer Interface: Centronics interface is provided to support both 700 and 300 series printers.

Prom Programming: support is provided for Data I/O System 19.

Software Description

The SYS16 includes the GENIX operating system, an enhanced version of Berkeley 4.1 bsd UNIX. These enhancements allow GENIX to fully support the features of the NS16000 family, providing an advanced, proven programming environment.

The GENIX operating system is a time-shared, demand-paged system with protected address spaces, supporting from one to eight users. It is completely compatible with the NSXC16 cross software package.

Included are a C compiler, based on Berkeley's portable C compiler, NS1600 assembler, linker, libraries, utilities, loader, editor, and debugger. Virtually all of the utilities that make UNIX a powerful operating system are provided.

A Pascal compiler is available as an option.

Physical Specifications

The standard SYS16 consists of the Processor Module, Disk-Tape Module, one terminal, the required interconnect cables, and supporting manuals.

Processor Module: this is a rectangular floor mounted unit with front mounted controls and indicators, and rear mounted I/O connections.

Height:	24 inches
Width:	7.5 inches
Depth:	27 inches
Color:	beige side panels with grey inner frame, and black front and rear
Weight:	38 pounds

Disk-Tape Module: this unit is physically similar to the processor module, with the exception of the Weight, which is 48 pounds.

Terminal: DEC VT100 compatible.

Environmental:

Altitude:	25,000 ft. non-operating 15,000 ft. operating
Temperature:	−20°C to 65°C non-operating 10°C to 40°C operating
Humidity:	5% to 80% max wet bulb 32°C minimum dew point 2°C

Electrical:

Processor Module:

FCC:	Class A
AC Voltage:	90–130 or 180–260 V_{AC}; 47–63 Hz
Fusing:	6A-Domestic 3A-European

Disk-Tape Module: same as processor module

Odering Information

Systems:

NS-SYS-1001:	Full system: Processor Module, Disk-Tape Module, one Terminal, GENIX operating system, cables, and manuals.
NS-SYS-1001E:	same as above configured for European power.

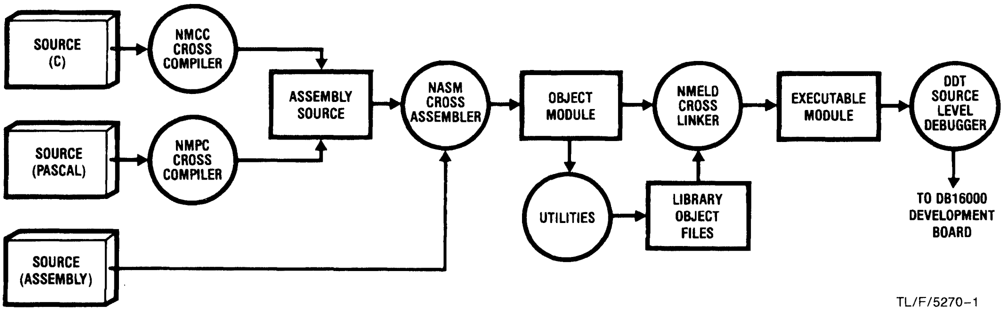
Accessories:

NS-SYS-2001:	Disk Drive Expansion Module with 40 MB
NS-SYS-2001E:	same as above configured for European power
NS-SYS-2002:	1 MB RAM Expansion Board
NS-SYS-2003:	Terminal
NS-SYS-2003E:	Terminal with European power configuration
NS-SYS-2004:	ISE for 16032
NS-SYS-2004E:	ISE for 16032 with European power configuration
NS-SYS-2005:	20 MB Streamer Tape Cartridge
NS-SYS-2006:	Hardware manual
NS-SYS-2007:	Software manual
NS-SYS-2008:	ISE for 16008
NS-SYS-2008E:	ISE for 16008 with European power configuration

Software:

NS-SYS-3001:	Pascal software

NSXC16™ Cross Software Package



TL/F/5270-1

- **Advanced software development environment**
- **C compiler**
- **Pascal compiler with extensions**
- **Powerful assembler supporting the full NS16000™ architecture**
- **Superior interactive debugger with efficient command interface**
- **Runs under DEC VAX11 with Berkeley 4.1 bsd UNIX O.S.**

Product Overview

NSXC16 is a comprehensive cross software package that supports an advanced software development environment for the NS16000 family. It is designed to run on DEC's VAX11 series with the Berkeley 4.1bsd UNIX operating system. It is compatible with all language tools of the SYS16'™ GENIX™ operating system.

All NSXC16 language tools are modeled after, and integrated with the SYS16 GENIX language tools. Included are a C compiler, an optional Pascal compiler, NS16000 assembler, linker, libraries, utilities, and an interactive debugger. The NSXC16 provides a full complement of tools to make the generation of NS16000 code an easy task. Programs thus developed can be downloaded via serial port to the DB16000 development board or ISE/16™ for execution and debug.

Components

nmcc—C Compiler

Designed to be compatible with the portable C compiler (pcc) of the Berkeley 4.1bsd UNIX system. The C compiler accepts compatible C source and generates NS16000 assembly language code. Designed to fully utilize the NS16000 architecture.

nmpc—Pascal Compiler (OPTIONAL)

ANSI standard with modular software extensions. Accepts compatible Pascal source and generates NS16000 assembly language code. Extensions include features such as import/export in support of full modularity. Designed to fully utilize the NS16000 architecture.

nasm—NS16000 Assembler

The assembler produces NS16000 object code in extended UNIX a.out format. It accepts complex expressions, external symbolic references, and external address arithmetic.

nmeld—Linker

Modules generated by the assembler can be linked by nmeld with the supplied libraries or user-generated ones to produce executable files.

include, libc.a, libpc.a—Libraries

The libraries contain standard UNIX include files, the C library, and the Pascal library.

nar, nnm, nranlib, nsize, nstrip—Utilities

Utilities provide the necessary tools to construct user defined libraries and to facilitate performance improvement.

ddt—Interactive Debugger

The interactive debugger allows remote debugging at the assembly language source level. It communicates with the DB16000 monitor via a serial link allowing execution and debugging on the board. Instruction may be displayed symbolically and breakpoints set by instruction. Single-stepping is possible at the machine instruction level, the procedure level, or when a register address value match occurs. ddt supports debugging in physical address space, supervisor virtual address space, and user virtual address space.

cu16—Remote Communication Utility

cu16 provides communication between the host system and DB16000 board. It is used interactively to download programs from the host system to the DB16000 board.

monitor—DB16000 Monitor

The DB16000 monitor is provided in source form and allows NS16000 customers to modify the monitor to suit their target system.

nburn—EPROM Programmer

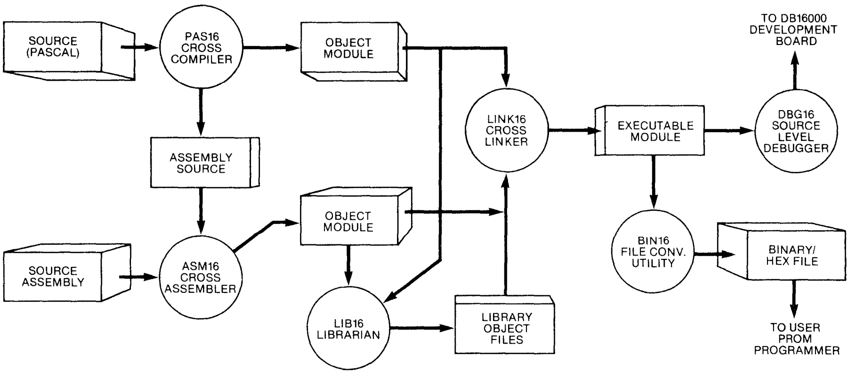
Nburn is used for the programming of EPROMs.

Ordering Information

NS-XC-16	NSXC16 software package on, 16000 bpi magnetic tape
NS-XC-PAS	nmpc Pascal compiler option

National Semiconductor

NSX16 Cross Software Package



NSX16 Cross Software Package

- **Runs under STARPLEX II™ operating system and DEC VAX/VMS operating system**
- **Compatible with ANSI standard PASCAL**
- **Supports NS16081 floating point unit**
- **PASCAL run-time support environment for DB16000 development board**
- **PASCAL compiler produces NS16000 code directly**
- **High-level symbolic debugger allows debug at source level**

Product Overview

NSX16 is a comprehensive software development package that includes all the components necessary to produce NS16000 native code. Intended as a support package to facilitate the development of software for NS16000-based systems, NSX16 has been designed to run initially on two hardware configurations. These are National Semiconductor's STARPLEX II operating system and Digital Equipment's VAX11 series running the VMS operating system.

Consisting of a PASCAL compiler, NS16000 cross-assembler, linker, librarian, and source-level debugger, NSX16 provides the full ensemble of tools to make the generation of NS16000 code an easy task. Code thus developed may then be downloaded via a serial port to the DB16000 development board for execution and debug.

NSX16 consists of the following components:

— PAS16, the PASCAL Cross-Compiler
Note: Not available for STARPLEX II.
— RTS16, the Run-Time Support Package
Note: Not available for STARPLEX II.
— ASM16, NS16000 Cross-Assembler
— LINK16, the Cross Linker
— LIB16, the Librarian
— BIN16, the File Conversion Utility
— DBG16, the Source-Level Symbolic Debugger

PAS16

Designed to be compatible with the ANSI standard, with listed extensions and restrictions, the PASCAL cross-compiler is capable of accepting compatible PASCAL source and generating NS16000 code. Extensions include features such as IMPORT/EXPORT in support of full modularity and FAST variables for code optimization. Also included is the run-time support environment for the DB16000 development board.

Note: PAS16 is not available for STARPLEX II Development Systems.

ASM16

The cross-assembler produces relocatable NS16000 object code. It accepts complex expressions, floating point scientific notation, external symbol references and can handle external address arithmetic.

LINK16

Modules generated by the cross-compiler or assembler are linked by LINK16 to produce executable modules. LINK16 is interactive, allowing the user to include additional files and libraries at link time whenever symbol matching is unsuccessful. LINK16 provides an extensive repertoire of directives to support complex system configurations. Directives can be entered from disk or directly from the console. LINK16 permits user control of RAM/ROM allocation.

LIB16

The librarian maps module characteristics and builds module libraries.

BIN16

The BIN16 program is a utility that converts 16000 executable files into a format acceptable for PROM programmers.

DBG16

DBG16 is an interactive symbolic debugger that allows debugging at the source level. It communicates with the DB16000 monitor via a serial link allowing execution and debug on the board. Source code may be displayed and breakpoints set by line numbers. Single-stepping is possible at the machine instruction level, the PASCAL statement level, the procedure level, or until a register/address value match occurs. DBG16 supports debugging of multimodule programs, drawing all information needed to support debug from source files and the output of the linker. It supports command files and output to a file to serve as a history file.

Supported Hardware

- STARPLEX II
  STARPLEX Operating System
  Revision G or later
- DEC VAX 11 Family
  VMS Operating System Vers 2.X or later

Shipping Package

VAX Version:
1600 bpi mag tape (9-track VMS copy format).
User and reference documentation.

STARPLEX II Version:
Compatible 8″ floppy diskette, standard soft sector format.
User and reference documentation.

Order Information

NSX-16	Cross Software Package, VAX/VMS version (Includes PAS16, RTS16 ASM16, LINK16, LIB16, BIN16 and DBG16)
NS-PAS-16	PASCAL Cross-Compiler, VAX/VMS version
NS-ASM-16	NS16000 Cross-Assembler, VAX/VMS version (Includes ASM16, LINK16, LIB16, BIN16 and DBG16)
SFW-90-A010	NS16000 Cross-Assembler, STARPLEX II version (Includes same as NS-ASM-16)

National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, California 95051
Tel: (408) 721-5000
TWX: (910) 339-9240

National Semiconductor
5955 Airport Road
Suite 206
Mississauga, Ontario
L4V1R9 Canada
Tel: (416) 678-2920
TWX: 610-492-8863

Electronica NSC de Mexico SA
Hegel No. 153-204
Mexico 5 D.F. Mexico
Tel: (905) 531-1689, 531-0569,
531-8204
Telex: 017-73550

NS Electronics Do Brasil
Avda Brigadeiro Faria Lima 830
8 Andar
01452 Sao Paulo, Brasil
Telex: 1121008 CABINE SAO PAULO
113193 INSBR BR

National Semiconductor GmbH
Furstenriederstraße Nr. 5
D-8000 München 21
West Germany
Tel.: (089) 5 60 12-0
Telex: 522772

National Semiconductor (UK), Ltd.
301 Harpur Centre
Horne Lane
Bedford MK40 1TR
United Kingdom
Tel: 0234-47147
Telex: 826 209

National Semiconductor Benelux
Ave. Charles Quint 545
B-1080 Bruxelles
Belgium
Tel: (02) 4661807
Telex: 61007

National Semiconductor (UK), Ltd.
1, Bianco Lunos Allè
DK-1868 Copenhagen V
Denmark
Tel: (01) 213211
Telex: 15179

National Semiconductor
Expansion 10000
28, Rue de la Redoute
F-92 260 Fontenay-aux-Roses
France
Tel: (01) 660-8140
Telex: 250956

National Semiconductor S.p.A.
Via Solferino 19
20121 Milano
Italy
Tel: (02) 345-2046/7/8/9
Telex: 332835

National Semiconductor AB
Box 2016
Stensätravägen 4/11 TR
S-12702 Skärholmen
Sweden
Tel: (08) 970190
Telex: 10731

National Semiconductor
Calle Nunez Morgado 9
(Esc. Dcha. 1-A)
E-Madrid 16
Spain
Tel: (01) 733-2954/733-2958
Telex: 46133

National Semiconductor Switzerland
Alte Winterthurerstrasse 53
Postfach 567
CH-8304 Wallisellen-Zürich
Tel: (01) 830-2727
Telex: 59000

National Semiconductor
Pasilanraitio 6C
SF-00240 Helsinki 24
Finland
Tel: (90) 14 03 44
Telex: 124854

NS Japan K.K.
POB 4152 Shinjuku Center Building
1-25-1 Nishishinjuku, Shinjuku-ku
Tokyo 160, Japan
Tel: (03) 349-0811
TWX: 232-2015 NSCJ-J

National Semiconductor Hong Kong, Ltd.
1st Floor,
Cheung Kong Electronic Bldg.
4 Hing Yip Street
Kwun Tong
Kowloon, Hong Kong
Tel: 3-899235
Telex: 43866 NSEHK HX
Cable: NATSEMI HX

NS Electronics Pty. Ltd.
Cnr. Stud Rd. & Mtn. Highway
Bayswater, Victoria 3153
Australia
Tel: 03-729-6333
Telex: AA32096

National Semiconductor PTE, Ltd.
10th Floor
Pub Building, Devonshire Wing
Somerset Road
Singapore 0923
Tel: 652 700047
Telex: NATSEMI RS 21402

National Semiconductor Far East, Ltd.
Taiwan Branch
P.O. Box 68-332 Taipei
3rd Floor, Apollo Bldg.
No. 218-7 Chung Hsiao E. Rd.
Sec. 4 Taipei Taiwan R.O.C.
Tel: 7310393-4, 7310465-6
Telex: 22837 NSTW
Cable: NSTW TAIPEI

National Semiconductor (HK) Ltd.
Korea Liaison Office
6th Floor, Kunwon Bldg.
No. 2, 1-GA Mookjung-Dong
Choong-Ku, Seoul, Korea
C.P.O. Box 7941 Seoul
Tel: 267-9473
Telex: K24942

DA-RRD20M63/Printed in U.S.A.