# iUP-200A/iUP-201A UNIVERSAL PROM PROGRAMMERS

## MAJOR iUP-200A/iUP-201A FEATURES:

- Support for all Intel PROM families through multiple-device personality modules, which may also be used with the Intel personal development system (iPDS™).
- Serial interface to all Intellec® development systems.
- Powerful PROM programming software (iPPS).
- iUP system self-tests plus device integrity checks.
- Support for new personality modules that provide state of the art fast programming algorithms, the inteligent Identifier™, and a security bit.

## ADDITIONAL iUP-201A FEATURES:

- Off-line editing, device duplication, and PROM memory locking.
- 32K-byte iUP RAM.
- 24-character alphanumeric display.
- Full hexadecimal plus 12-function keypad.

The iUP-200A and iUP-201A universal programmers program and verify data in all the Intel programmable ROMs (PROMs). They can also program the PROM memory portions of Intel's single-chip microcomputer and peripheral devices. When used with any Intellec® development system, the iUP-200A and iUP-201A universal programmers provide on-line programming and verification using the Intel PROM programming software (iPPS). In addition, the iUP-201A universal programmer supports off-line, stand-alone program editing, PROM duplication, and PROM memory locking. The iUP-200A universal programmer is expandable to an iUP-201A model.






March, 1984
Order Number:210319-003

## FUNCTIONAL DESCRIPTION

The iUP-200A universal programmer operates in on-line mode. The iUP-201A universal programmer operates in both on-line and off-line mode.

### On-line System Hardware

The iUP-200A and iUP-201A universal programmers are free-standing units that, when connected to any Intel development system having at least 64K bytes of host memory, provide on-line PROM programming and verification of Intel programmable devices. In addition, the universal programmer can read the contents of the ROM versions of these devices.

The universal programmer communicates with the host through a standard RS-232C serial data link. A serial converter is needed when using the MDS 800 as a host system. (Serial converters are available from other manufacturers.)

Each universal programmer contains an 8085 CPU, selectable power supply, 4.3K bytes of static RAM, a programmable timer, an interface for personality modules, an interface for the host system, and 12K bytes of programmed EPROM. The iUP-201A also has a keyboard and display. The programmed EPROM contains the firmware needed for all universal programmer editing and control functions.

A personality module adapts the universal programmer to a family of PROM devices; it contains all the hardware and firmware necessary to program either a family of Intel PROMs or a single Intel device. The user inserts the personality module into the universal programmer front panel. The personality module comes ready to use; no additional sockets or adapters are required.

Figure 1 shows the iUP-200A on-line system configuration, and Figure 2 shows the on-line system data flow.

### On-line System Software

The Intel PROM programming software (iPPS) is included with both the iUP-200A and iUP-201A models of the universal programmer. Created to run on any Intellec development system, the iPPS software provides user control through an easy-to-use interactive interface. The iPPS software performs the following functions to make PROM programming quick and easy:

- Reads PROMs and ROMs
- Programs PROMs directly or from a file

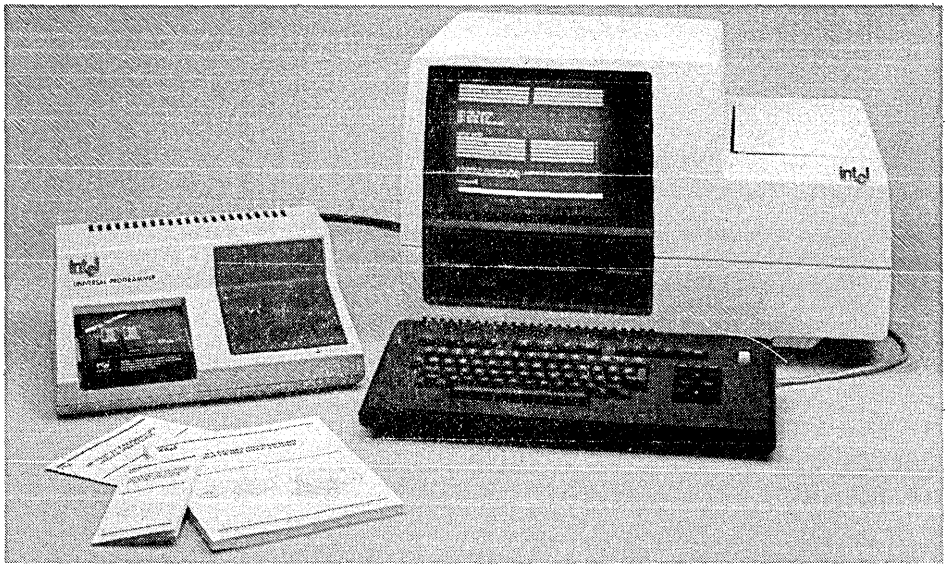


**Figure 1 On-Line System Configuration**

Order Number: 210319-003

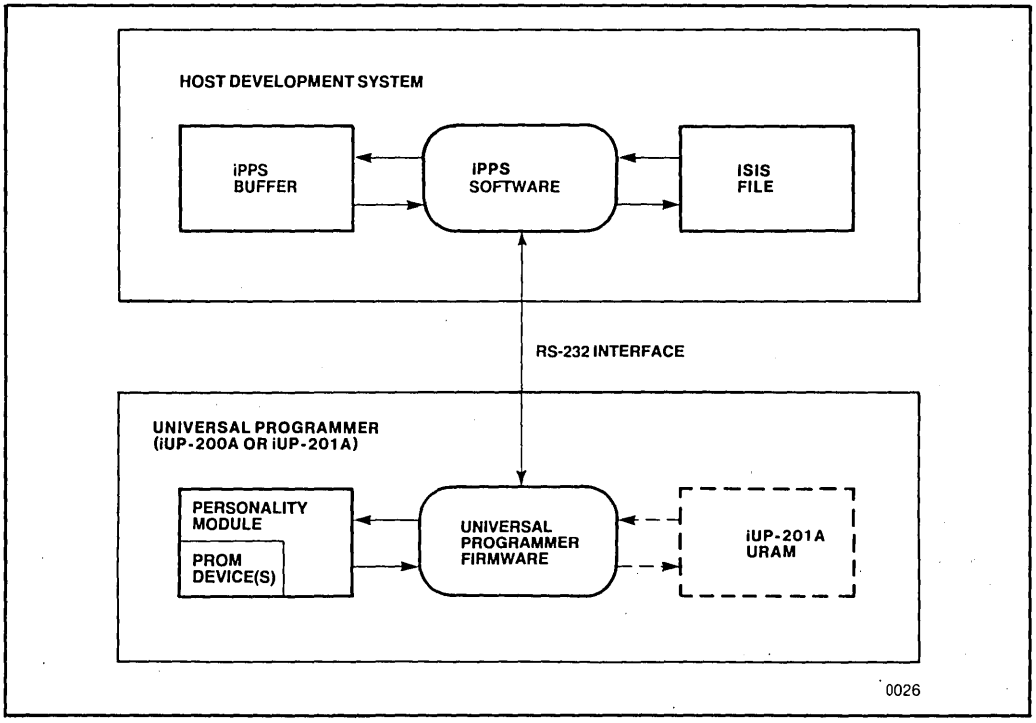



**Figure 2 On-Line System Data Flow**

- Verifies PROM data with buffer data
- Locks EPROM memory from unauthorized access (on devices which support this feature)
- Prints PROM contents on the network or development system printer
- Performs interactive formatting operations such as interleaving, nibble swapping, bit reversal, and block moves
- Programs multiple PROMs from the source file, prompting the user to insert new PROMs
- Uses a buffer to change PROM contents

All iPPS commands, as well as program address and data information, are entered through the development system ASCII keyboard and displayed on the system CRT. Table 1 summarizes the iPPS commands.

The iPPS software lets the user load programs into a PROM from Intellec system memory or directly from a disk file. Access to the disk lets the user create and manipulate data in a virtual buffer with an address range up to 16M. This large block of data can be formatted into different PROM word sizes for program storage into several different PROM types. In addition, a program stored in the target PROM, the Intellec system memory, or a system disk file can be interleaved with a second program and entered into a specific target PROM or PROMs.

The iPPS software supports data manipulation in the following Intel formats: 8080 hexadecimal ASCII, 8080 absolute object, 8086 hexadecimal ASCII, 8086 absolute object, and 80286 absolute object. Addresses and data can be displayed in binary, octal, decimal, or hexadecimal. The user can easily change default data formats as well as number bases.

The user invokes the iPPS software from the ISIS operating system (Intellec 800, Series II, and Series III, versions V3.4 and later; Series IV, versions V1.0 and later). The software can be run under control of ISIS submit files, thereby freeing the user from repetitious command entry.

Order Number: 210319-003

**Table 1 iPPS Command Summary**

| Command | Description |
|---|---|
| PROGRAM CONTROL GROUP | CONTROLS EXECUTION OF THE iPPS SOFTWARE. |
| EXIT | Exits the iPPS software and returns control to the ISIS operating system. |
| <ESC> | Terminates the current command. |
| REPEAT | Repeats the previous command. |
| ALTER | Edits and re-executes the previous command. |
| UTILITY GROUP | DISPLAYS USER INFORMATION AND STATUS AND SETS DEFAULT VALUES. |
| DISPLAY | Displays PROM, buffer, or file data on the console. |
| PRINT | Prints PROM, buffer, or file data on the local printer. |
| QUEUE | Prints PROM, buffer, or file data on the network spooled printer. |
| HELP | Displays user assistance information. |
| MAP | Displays buffer structure and status. |
| BLANKCHECK | Checks for unprogrammed PROMs. |
| OVERLAY | Checks whether non-blank PROMs can be programmed. |
| TYPE | Selects the PROM type. |
| INITIALIZE | Initializes the default number base and file type. |
| WORKFILES | Specifies the drive device for temporary work files. |
| BUFFER GROUP | EDITS, MODIFIES, AND VERIFIES DATA IN THE BUFFER. |
| SUBSTITUTE | Examines and modifies buffer data. |
| LOADDATA | Loads a section of the buffer with a constant. |
| VERIFY | Verifies data in the PROM with buffer data. |
| FORMATTING GROUP | REARRANGES DATA FROM THE PROM, BUFFER, OR FILE. |
| FORMAT | Formats and interleaves buffer, PROM, or file data. |
| COPY GROUP | COPIES DATA FROM ONE DEVICE TO ANOTHER. |
| COPY (file to PROM) | Programs the PROM with data in a file on disk. |
| COPY (PROM to file) | Saves PROM data in a file on disk. |
| COPY (buffer to PROM) | Programs the PROM with data from the buffer. |
| COPY (PROM to buffer) | Loads the buffer with data in the PROM. |
| COPY (buffer to file) | Saves the contents of the buffer in a file on disk. |
| COPY (file to buffer) | Loads the buffer from a file on disk. |
| COPY (file to URAM) | Loads file data into the iUP RAM (iUP-201A model only). |
| COPY (URAM to file) | Saves iUP URAM data in a file on disk (iUP-201A model only). |
| COPY (buffer to URAM) | Loads the buffer into the iUP URAM (iUP-201A model only). |
| COPY (URAM to buffer) | Loads iUP URAM data into the buffer (iUP-201A model only). |
| SECURITY GROUP | LOCKS SELECTED DEVICES TO PREVENT UNAUTHORIZED ACCESS. |
| KEYLOCK | Locks the PROM from unauthorized access. |

## System Expansion

The iUP-200A universal programmer can be easily upgraded (by the user) to an iUP-201A universal programmer for off-line operation. The upgrade kit (iUP-PAK-A) is available from Intel or your local Intel distributor.

## Off-line System

The iUP-201A universal programmer has all the on-line features of the iUP-200A universal programmer plus off-line editing, PROM duplication, program verification, and locking of PROM memory independent of the host system. The iUP-201A universal programmer also accepts Intel hexadecimal programs developed on non-Intel development systems. Just a few keystrokes download the program into the iUP RAM for editing and loading into a PROM.

Off-line commands are entered using the off-line command keys summarized in Table 2.

In addition to the hardware components included as part of the iUP-200A, the iUP-201A contains a 24-character alphanumeric display, full hexadecimal 12-function keypad, and 32K bytes of iUP RAM. Figure 3 illustrates the iUP-201A keyboard and display.

The two logical devices accessible during off-line operation are the PROM device and the iUP RAM. A typical operation is copying the data from a PROM (or ROM) into the iUP RAM, modifying this data in iUP RAM, and programming the modified data back into a PROM device. The address range of the iUP RAM is automatically determined by the universal programmer when PROM type selection is made. Figure 4 shows the off-line system data flow.

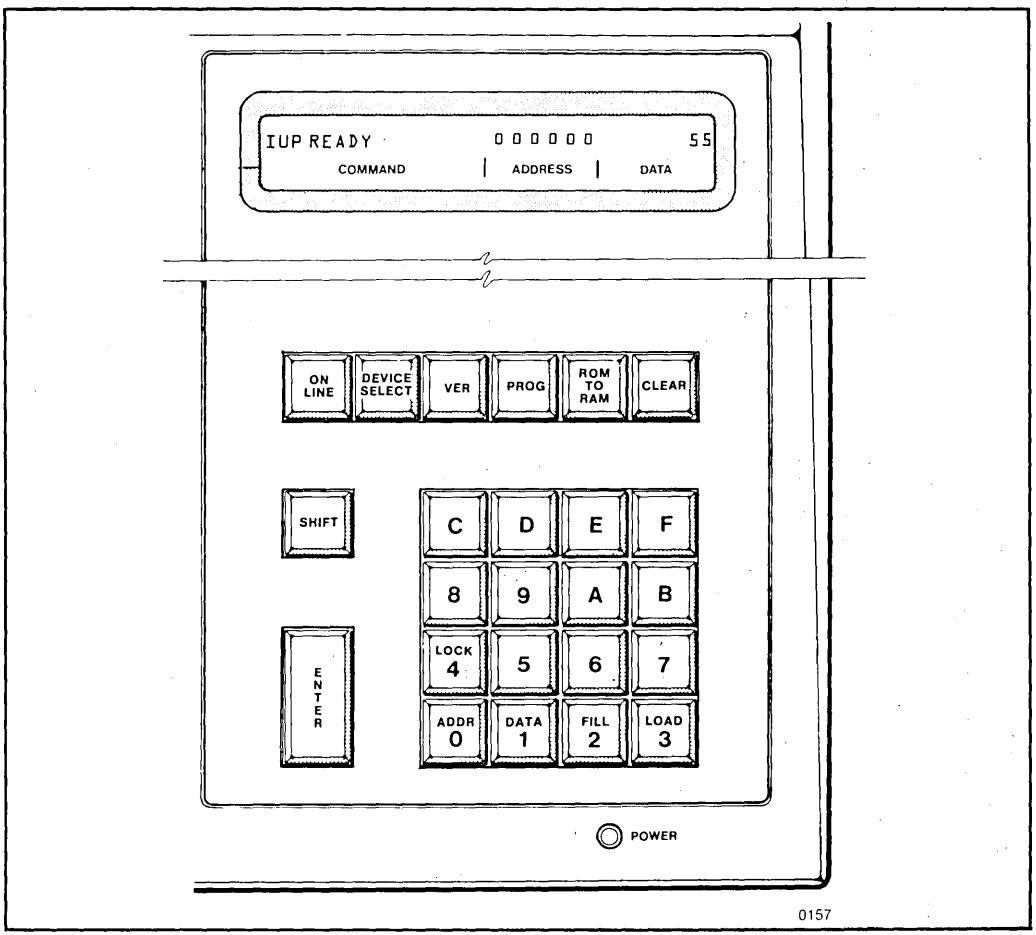



**Figure 3 iUP-201A Keyboard and Display**

Order Number: 210319-003

**Table 2 Off-Line Command Keys Summary**

| Key | Function |
|---|---|
| ON LINE | Selects either on-line or off-line operation. Wher on-line, all other function keys are disabled. |
| DEVICE SELECT | Selects the PROM type when using a personality module able to program multiple PROM devices. |
| VER | Verifies the contents of the installed PROM device with the contents of the iUP RAM. The universal programmer display indicates the address and the XOR of any mismatches. |
| PROG | Performs a device blank check and then programs the target PROM with data from the iUP RAM. If the blank check fails, pressing PROG again performs an overlay check to verify that non-blank PROMs can be programmed. |
| ROM TO RAM | Loads the iUP RAM with the data from the PROM device installed in the personality module. |
| CLEAR | Terminates the current off-line function, clears a user entry, or restores the display after an error. |
| ENTER | Transfers information from the universal programmer display (addresses, data, or baud rate) into the iUP RAM. |
| SHIFT + ADDR 0 | Selects an address field for display. |
| SHIFT + DATA 1 | Selects a data field for keypad editing and entry. |
| SHIFT + FILL 2 | Loads a contiguous section of iUP RAM locations with a constant. |
| SHIFT + LOAD 3 | Downloads Intel hexadecimal data from any development system which has an RS-232C port. |
| SHIFT + LOCK 4 | Locks a PROM from unauthorized access. |

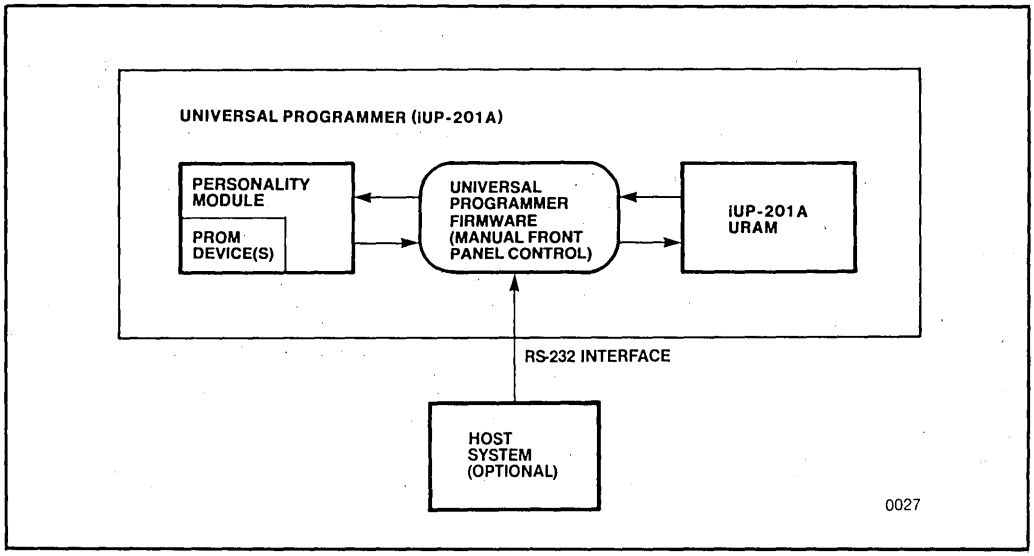



**Figure 4 Off-Line System Data Flow**

## SYSTEM DIAGNOSTICS

Both the iUP-200A and iUP-201A universal programmers include self-contained system diagnostics that verify system operation and aid the user in fault isolation. Diagnostics are performed on the power supply, CPU internal firmware ROM, internal RAM, timer, the iUP-201A keyboard, and the iUP RAM. In addition, tests are made on any personality module installed in the programmer the first time the module is accessed. The personality module tests include the power select circuitry and up to 4K of module firmware. Straight-forward messages are provided on the development system display in on-line mode and on the iUP-201A display in off-line mode.

## PERSONALITY MODULES

A personality module is the interface between the iUP-200A/iUP-201A universal programmer (or an iPDS system) and a selected PROM (or ROM). Personality modules contain all the hardware and firmware for reading and programming a family of Intel devices. Each personality module is a single molded unit inserted into the front panel of the universal programmer. No additional adapters or sockets are needed. Table 3 lists the available personality modules.

Each personality module connects to the universal programmer through a 41-pin connector. Module firmware is uploaded into the iUP RAM and executed by the internal 8085A processor.

**Table 3 iUP Personality Modules**

| Personality Module | PROM Type | PROMs and ROMs Supported |
|---|---|---|
| iUP-Fast 27/K | EPROM | 2764, 2764A, 27128, 27256 |
| iUP-F27/128 | $E^2$/EPROM | 2716, 2732, 2732A, 2764, 27128, 2815, 2816 |
| iUP-F87/51A | Microcontroller | 8748, 8748H, 8048, 8749H, 8048H, 8049, 8049H, 8050H, 8751, 8751H, 8051 |
| iUP-F87/44A | Peripheral | 8741A, 8041A, 8742, 8042, 8744H, 8044AH, 8755A |

Order Number: 210319-003

The personality module firmware contains routines necessary to read and program a family of PROMs. In addition, the personality module sends specific information about the selected PROM to the universal programmer to help perform PROM device integrity checks.

LEDs on each personality module indicate operational status. On some personality modules a column of LEDs indicate which PROM device type the user has selected. On some personality modules an LED below the socket indicates which socket is to be used. A red indicator light tells the user when power is being supplied to the selected device. Figure 5 shows the personality modules supported on the universal programmer.

In addition to the testing done by the iUP system self-tests, each personality module contains diagnostic firmware that performs selected PROM tests and indicates status. These tests are performed in both on-line and off-line modes. The PROM installation test verifies that the device is installed in the module correctly and that the ZIF socket is closed. The PROM blank check determines whether a device is blank. The universal programmer automatically determines whether the blank state is all zeros or all ones. The overlay check (performed when a PROM is not blank) determines which bits are programmed, compares those bits against the program to be loaded, and allows programming to continue if they match. As with the system self-tests, straight-forward messages are provided. The user can invoke all of the PROM device integrity checks except the installation test (which occurs automatically any time an operation is selected).

Figure 6 illustrates a typical testing sequence.

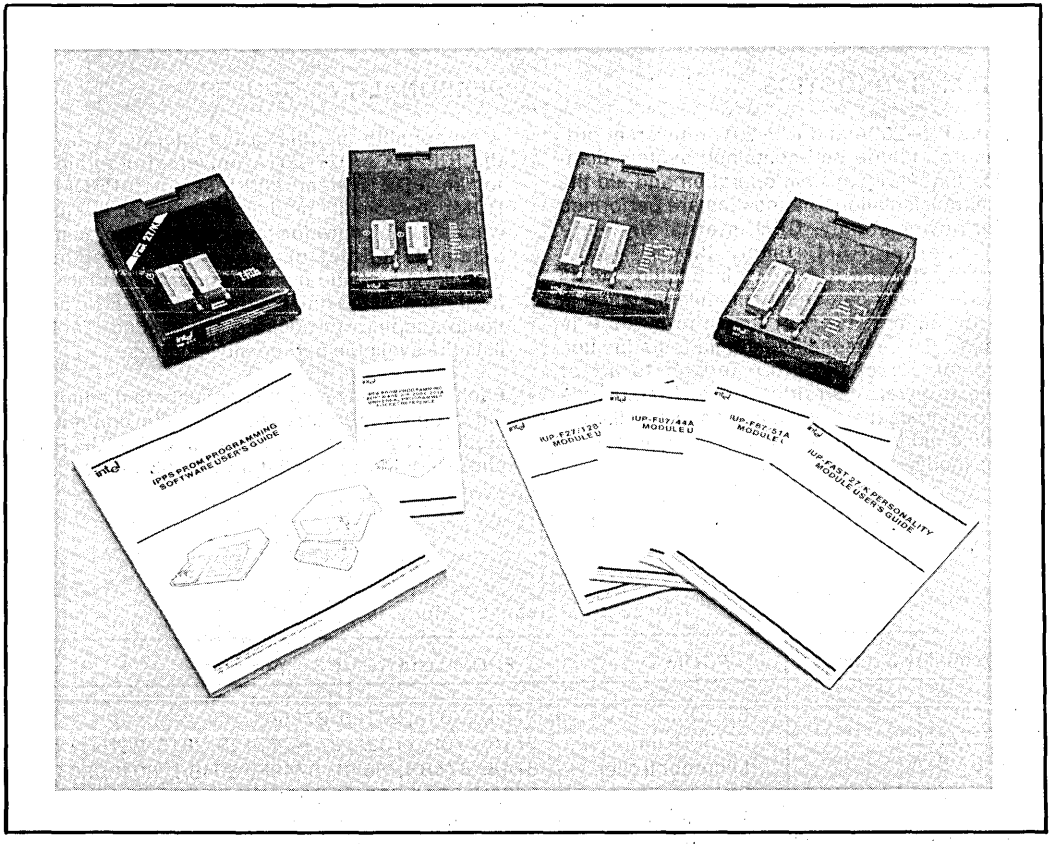



**Figure 5 Personality Modules**

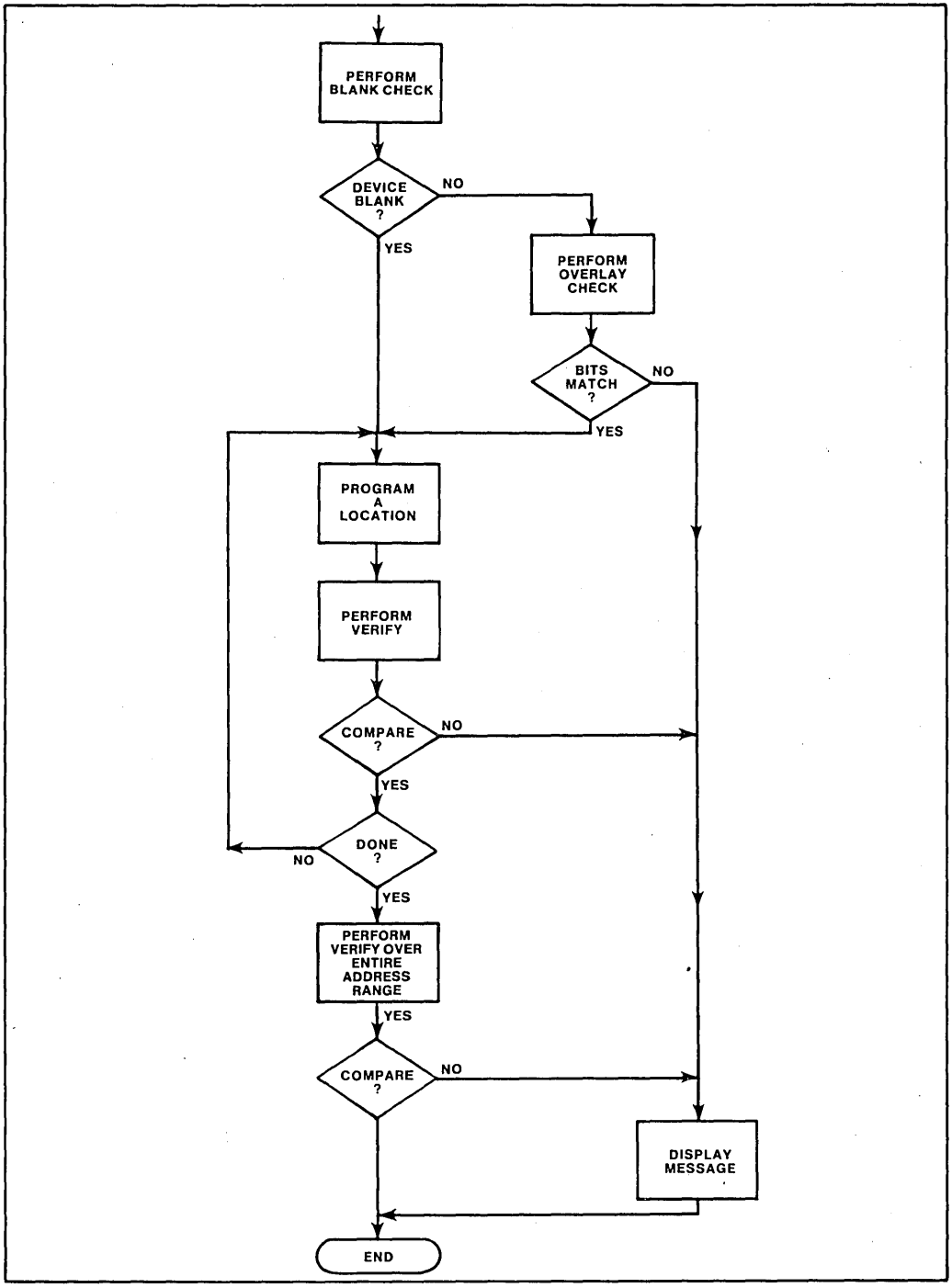



**Figure 6 PROM Testing Sequence**

Order Number: 210319-003

## iUP-200A/iUP-201A SPECIFICATIONS

### Control Processor

Intel 8085A microprocessor
6.144 MHz clock rate

### Memory

RAM — 4.3 bytes static
ROM — 12K bytes EPROM

### Interfaces

Keyboard — 16-character hexadecimal and 12-function keypad (iUP-201A model only)
Display — 24-character alphanumeric (iUP-201A model only)

### Software

Monitor — system controller in pre-programmed EPROM
iPPS — Intel PROM programming software on supplied diskette

### Physical Characteristics

Depth — 15 inches (38.1 cm)
Width — 15 inches (38.1 cm)
Height — 6 inches (15.2 cm)
Weight — 15 pounds (6.9 kg)

### Electrical Characteristics

Selectable 100, 120, 200, or 240 Vac ± 10%; 50-60 Hz
Maximum power consumption — 80 watts

### Environmental Characteristics

Reading temperature — 10°C to 40°C
Programming temperature — 25°C ± 5°
Operating humidity — 10% to 85% relative humidity

### Reference Material

**164852** — *iUP-200A/201A Universal Programmer User's Guide.*
**164861** — *iPPS PROM Programming Software User's Guide.*
**164853** — *iPPS PROM Programming Software/iUP-200A/201A Universal Programmer Pocket Reference.*

## PERSONALITY MODULE SPECIFICATIONS

### Memory

EPROM — up to 4K bytes

### Physical Characteristics

Width — 5.5 inches (1.4 cm)
Height — 1.6 inches (4.1 cm)
Depth — 7.0 inches (17.8 cm)
Weight — 1 pound (.45 kg)

### Electrical Characteristics

Maximum power consumption (module) — 7.5 watts
Maximum power consumption (device) — 2.5 watts
Maximum power consumption (total from iUP) — 10 watts

### Environmental Characteristics

Reading temperature — 10°C to 40°C
Programming temperature — 25°C ± 5°
Operating humidity — 10% to 85% relative humidity

### Reference Material

Appropriate personality module user's guide:

**164376** — *iUP-Fast 27/K Personality Module User's Guide.*
**162848** — *IUP-F27/128 Personality Module User's Guide.*
**164855** — *iUP-F87/51A Personality Module User's Guide.*
**164853** — *iUP-F87/44A Personality Module User's Guide.*

## ORDERING INFORMATION

| Part number | Description | | |
|---|---|---|---|
| iUP-200A | Intel on-line universal programmer | iUP-Fast 27/K* | EPROM personality module |
| iUP-201A | Intel on-line/off-line universal programmer | iUP-F27/128 | EPROM and E²PROM personality module |

Order Number: 210319-003

| | |
|---|---|
| iUP-F87/51A | Microcontroller personality module |
| iUP-F87/44A | Peripheral personality module |
| iUP-200/201 U1 Upgrade Kit | Upgrades an iUP-200/201 universal programmer to an iUP-200A/201A universal programmer |
| iUP-PAK-A Upgrade Kit | Upgrades an iUP-200A universal programmer to an iUP-201A universal programmer |

*The iUP-Fast 27/K personality module can be used only with an iUP-200A/201A universal programmer or an iUP-200/iUP-201 universal programmer upgraded to an A with the iUP-200/201 U1 upgrade kit. If used in an iPDS, this personality module requires version 1.4 or later of the iPPS-iPDS software. All iPDS-140 units shipped after June 1984 will contain this software.

# PROM PROGRAMMING PERSONALITY MODULES

**MAJOR PERSONALITY MODULE FEATURES:**

- **Adapts an iUP-200A/iUP-201A Universal Programmer or Intel Personal Development System (iPDS™) to a family of PROM devices.**
- **Comes ready to use.**
- **Includes the Fast 27/K personality module that programs Intel's latest PROM devices in one tenth the time.**
- **Supports multiple PROM device types.**

Personality modules custom-fit the iUP-200A/iUP-201A Universal Programmer or the iPDS™ system to a family of PROM devices. Each personality module comes ready to use—just plug it into a Universal Programmer or an iPDS system and begin reading or programming parts. The personality modules can be used off-line or controlled from a host or iPDS system using Intel's powerful PROM programming software (iPPS). Selected personality modules support the latest PROM programming features such as the int$_e$ligent Programming™ algorithms (reduce programming time up to a factor of 10), the int$_e$ligent Identifier™ (automatically selects the correct int$_e$ligent Programming algorithm), and the security bit function (protects PROM memory from unauthorized access).







May, 1984
ORDER NUMBER: 280003-001

## PERSONALITY MODULE DESCRIPTION

The personality module adapts the universal programmer or the iPDS system to a specific family of PROM devices; it contains all the hardware and firmware necessary to read and program a family of Intel PROMs. The personality module comes ready to use; the user merely inserts the personality module into the universal programmer front panel or the side door of the iPDS chassis. No additional sockets or adapters are required.

As Table 1 shows, each personality module supports a different family of PROM devices.

Each personality module connects to the universal programmer/iPDS system through a 41-pin connector. LEDs on the personality module indicate its operational status. A column of LEDs or a hexadecimal display indicates which PROM device type the user has selected. On some personality modules, an LED below the socket indicates which socket is to be used. A red indicator light tells the user when power is applied to the selected device.

After specifying the PROM device type, the user inserts the PROM to be programmed or read in the socket on the personality module. The personality module checks for correct PROM installation. In addition, each personality module contains diagnostic firmware that performs the following selected PROM tests and indicates status.

- The PROM installation test verifies that the device is installed in the module correctly and that the ZIF socket is closed.
- The PROM blank check determines whether a device is blank. The universal programmer/iPDS system automatically determines whether the blank state is all zeros or all ones.
- The overlay check (performed when a PROM is not blank) determines which bits are programmed, compares those bits with the program to be loaded, and allows programming to continue if they match.

The user can invoke all the PROM device integrity checks except the installation test (which occurs automatically any time an operation is selected).

## PROM PROGRAMMERS

The personality modules are used with either the universal programmer or the iPDS system. Both the iUP-200A and iUP-201A models of the universal programmer program PROM devices in on-line mode. The iPPS software which controls on-line programming runs on the host system. The iUP-201A universal programmer adds an additional feature: off-line programming directly from the universal programmer's keyboard. Figure 1 shows an iUP-201A universal programmer with a personality module inserted.

The iPDS system features stand-alone on-line programming controlled by the iPDS-iPPS software which runs on the iPDS system. The iPDS system operates in on-line mode only. Figure 2 shows an iPDS system with a personality module inserted.

Table 2 compares the features of the universal programmer with the features of the iPDS system.

**Table 1. Personality Modules**

| Personality Module | PROM Type | PROMs and ROMs Supported |
|---|---|---|
| iUP-Fast 27/K | EPROM | 2764, 2764A, 27128, 27256 |
| iUP-F27/128 | E²PROM/EPROM | 2716, 2732, 2732A, 2764, 27128, 2815, and 2816 |
| iUP-F87/51A | Microcontroller | 8748, 8748H, 8048, 8749H, 8048H, 8049, 8049H, 8050H, 8751, 8751H, 8051 |
| iUP-F87/44A | Peripheral | 8741A, 8041A, 8742, 8042, 8744H, 8044AH, 8755A |

280003-001

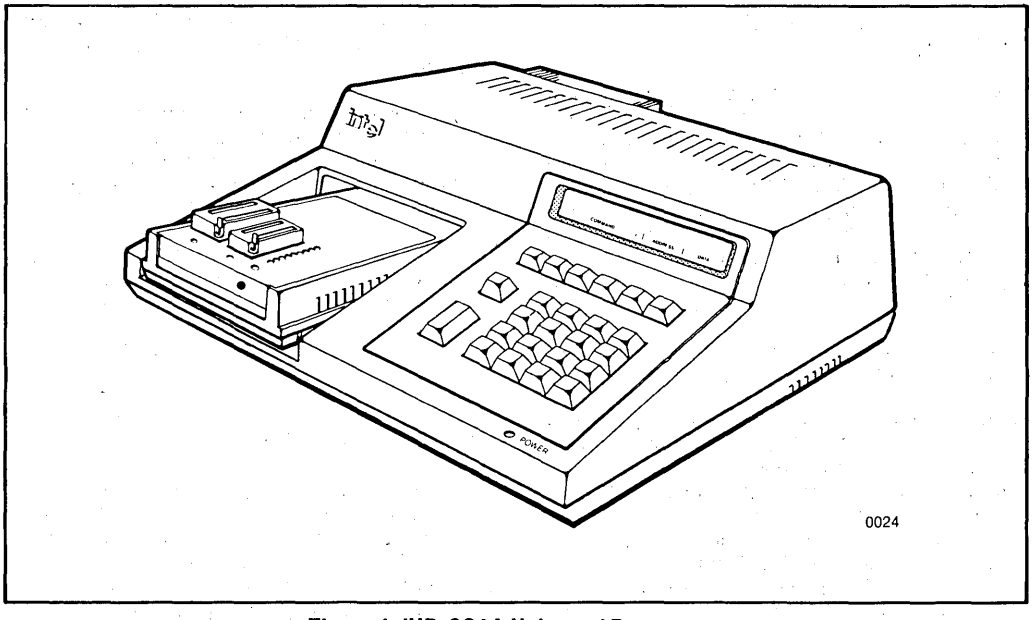



**Figure 1. iUP-201A Universal Programmer**

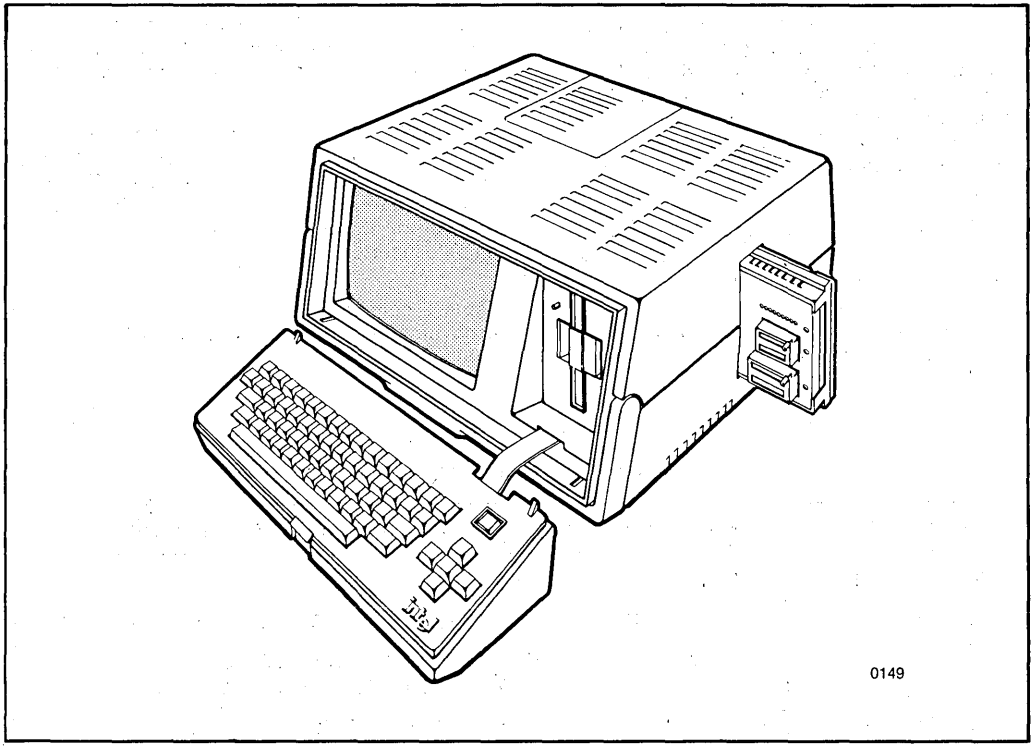



**Figure 2. iPDS™ System**

280003-001

Table 2. PROM Programmers

| Features | iUP-200A Universal Programmer | iUP-201A Universal Programmer | iPDS™ System |
|---|---|---|---|
| Function | PROM programmer | PROM programmer | Development system and PROM programmer |
| Operating mode | On-line mode | On-line mode and off-line mode | On-line mode |
| Configuration | Requires host system running iPPS software | Requires host system in on-line mode; stand-alone in off-line mode | Stand-alone plugged into iPDS system |
| Data display | On CRT of host system terminal | On built-in single-line display in stand-alone mode | On iPDS CRT |
| Input keyboard | From host system terminal | Built-in keyboard | From iPDS Keyboard |

## THE iPPS SOFTWARE

The iPPS software, included with both the iUP-200A and iUP-201A models of the universal programmer and with the iPDS system, brings increased flexibility to PROM programming. The iPPS software provides user control through an easy-to-use interactive interface and performs the following functions to make PROM programming quick and easy:

- Reads PROMs and ROMs.
- Programs PROMs directly or from a file.
- Verifies PROM data with buffer data.
- Locks EPROM memory from unauthorized access (on devices which support this feature).
- Prints PROM contents on the network printer (universal programmer only) or the development system printer.
- Performs interactive formatting operations such as interleaving, nibble swapping, bit reversal, and block moves.
- Programs multiple PROMs from the source file, prompting the user to insert new PROMs.
- Uses a buffer to change PROM contents.

With the iPPS software the user can load programs into a PROM from system memory or directly from a disk file. Access to the disk lets the user create and manipulate data in a virtual buffer. This block of data can be formatted into different PROM word sizes for program storage into several different PROM types. In addition, a program stored in the target PROM, the system memory, or a system disk file can be interleaved with a second program and entered into a specific target PROM or PROMs.

The iPPS software supports data manipulation in the following Intel formats: 8080 hexadecimal ASCII, 8080 absolute object, 8086 hexadecimal ASCII, 8086 absolute object, and 80286 absolute object. Addresses and data can be displayed in binary, octal, decimal, or hexadecimal. The user can easily change default data formats as well as number bases.

The user invokes the iPPS software from the ISIS operating system (Intellec 800, Series II, and Series III, ISIS versions V3.4 and later; Series IV, ISIS versions V1.0 and later; iPDS system, ISIS versions 1.4 and later). The software can be run under control of ISIS submit files, thereby freeing the user from repetitious command entry.

Note that the universal programmer and the iPDS system each has its own version of the iPPS software. To distinguish between them, the iPPS software for the iPDS system is called iPPS-iPDS software.

## PERSONALITY MODULE FEATURES

The personality modules described in the following sections allow a universal programmer/iPDS system to program a wide range of PROM devices, each with its unique needs and requirements: PROMs, EPROMs, E²PROMs, microcontrollers, and microprocessor peripherals.

280003-001

Note that the user needs one of the following configurations to use the Fast 27/K personality module or to use the security bit function on the iUP-F87/51A and iUP-F87/44A personality modules:

iPDS system

Intel PROM programming software (iPPS-iPDS), version 1.4 or later

iPDS-140 EMV/PROM adapter option

universal programmer

- on-line — Intel PROM programming software (iPPS), version 1.4 or later; model 200A or 201A
- off-line — model 201A

The user can easily update an iUP-200/201 universal programmer to an iUP-200A/201A universal programmer with the iUP-200/201 U1 upgrade kit.

## The iUP-Fast 27/K Personality Module

The iUP-Fast 27/K personality module lets the user program, read, and verify the contents of Intel's newest 64K and 256K EPROMs. This personality module supports the int$_e$ligent Programming algorithms and the int$_e$ligent Identifier. The int$_e$ligent Identifier lets the personality module interrogate the PROM device in the program/master socket. It determines whether the type selected matches the type of PROM device installed and then selects the proper int$_e$ligent Programming algorithm. The int$_e$ligent Programming algorithms reduce programming time up to a factor of 10.

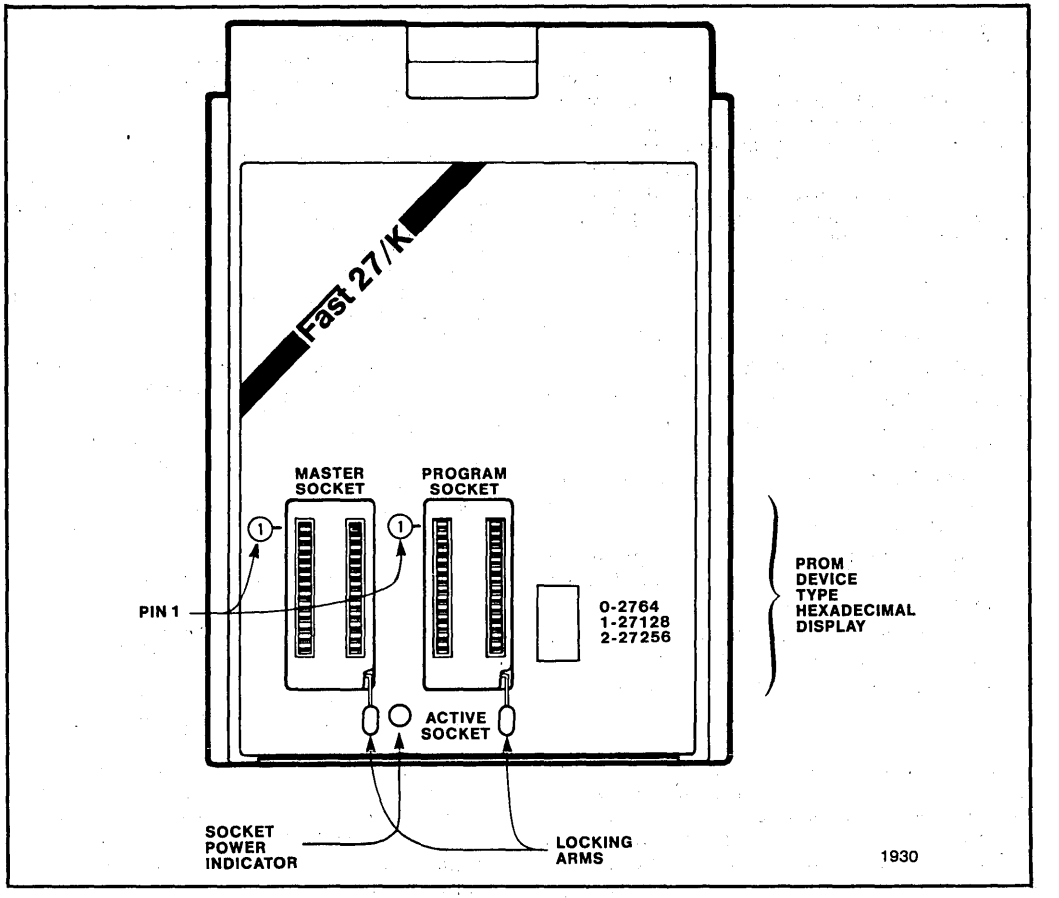



**Figure 3. iUP-Fast 27/K Personality Module**

280003-001

The iUP-Fast 27K personality module supports the following PROM devices:

2764 2764A 27128 27256

As shown in Figure 3, the iUP-Fast 27/K personality module contains two 28-pin sockets, a hexadecimal display (0 through F), and a red LED that indicates when power is being applied to a socket. The program socket holds the device being programmed. The master socket will be used in future upgrades. The hexadecimal display shows the PROM device type selected.

## The iUP-F27/128 Personality Module

The iUP-F27/128 personality module lets the user program, read, and verify the contents of a wide variety of PROM devices, including some of Intel's most popular PROM devices. This personality module supports the following PROM devices:

2716 2732 2732A 2764 27128 2815 2816

As shown in Figure 4, the iUP-F27/128 personality module contains two sockets: one for 24-pin PROM devices and the other for 28-pin PROM devices. The user can use only one socket at a time. An LED below the socket indicates the correct socket to use based on the PROM device type selected, and a row of green LEDs on the right side of the personality module indicate which PROM type is selected. The ACTIVE SOCKET LED indicates when power is being applied to the PROM device and when the universal programmer/iPDS system is accessing the selected socket.

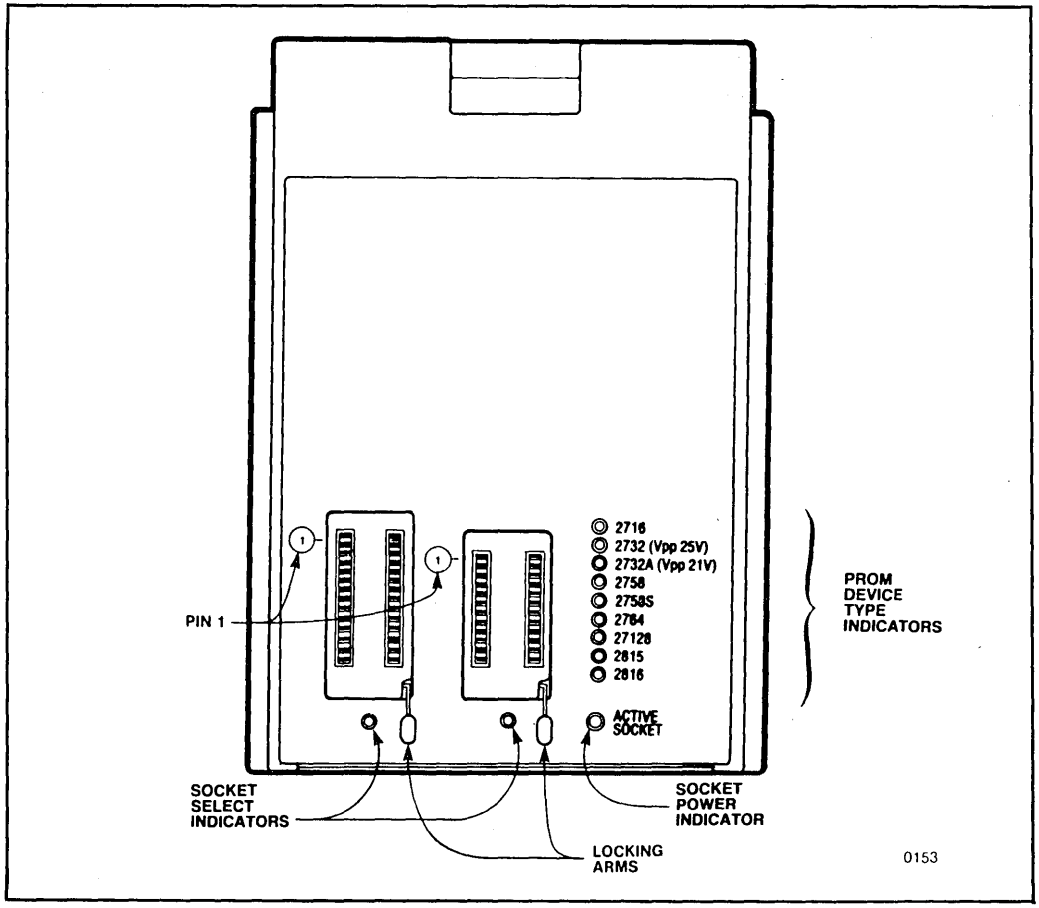



**Figure 4. iUP-F27/128 Personality Module**

280003-001

## The iUP-F87/51A Personality Module

The iUP-F87/51A personality module lets the user program EPROM microcontrollers and read the memory contents of ROM microcontrollers. This personality module supports the security bit function on the 8751H microcontroller. The KEYLOCK command locks the 8751H EPROM memory from unauthorized access by setting the security bit (which cannot be unlocked without erasing the device). As a safety precaution, the KEYLOCK command requires user verification before locking the security bit.

The iUP-F87/51A personality module supports the following PROM devices:

| | | | | |
|---|---|---|---|---|
| 8748 | 8748H | 8048 | 8048H | 8749H |
| 8049 | 8050H | 8751 | 8751H | 8051 |

As shown in Figure 5, the iUP-F87/51A personality module has two sockets for inserting applicable PROM devices: one for the MCS®-48 family of devices and the other for the MCS-51 family of PROM devices. An LED below the socket indicates the correct socket to use based on the PROM device type selected. One of the green

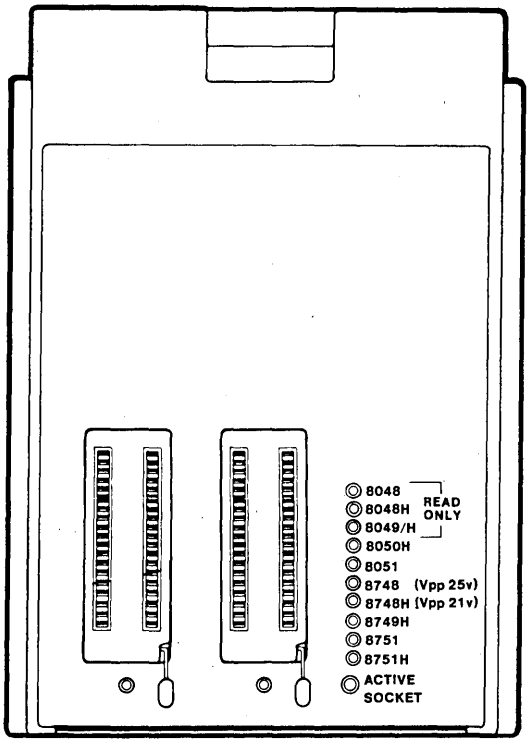



**Figure 5. iUP-F87/51A Personality Module**

280003-001

LEDs on the right side of the personality module lights to indicate the PROM type selected. The ACTIVE SOCKET LED lights when power is applied to the PROM device and when the universal programmer/iPDS system is accessing the selected socket.

## The iUP-F87/44A Personality Module

The iUP-F87/44A personality module lets the user program EPROM versions of the 8044 family of microcontroller/serial interface units and read the memory contents of ROM versions. This personality module supports the security bit function on the 8744H microcontroller. The KEYLOCK command locks the 8744H EPROM memory from unauthorized access by setting the security bit (which cannot be cleared without erasing the device). As a safety precaution, the KEYLOCK command requires user verification before setting the security bit.

The iUP-F87/44A personality module supports the following PROM devices:

| | | | |
|---|---|---|---|
| 8741A | 8041A | 8742 | 8042 |
| 8744H | 8044AH | 8755A | |

As shown in Figure 6, the iUP-F87/44A personality module has two sockets for inserting applicable PROM devices: one for the 8741A, 8742, and 8755A PROM devices and the other for the 8744H PROM device. An LED below each socket indicates the correct socket to use based on the PROM device type selected. One of the green LEDs on the right side of the personality module lights to indicate the PROM type selected. The

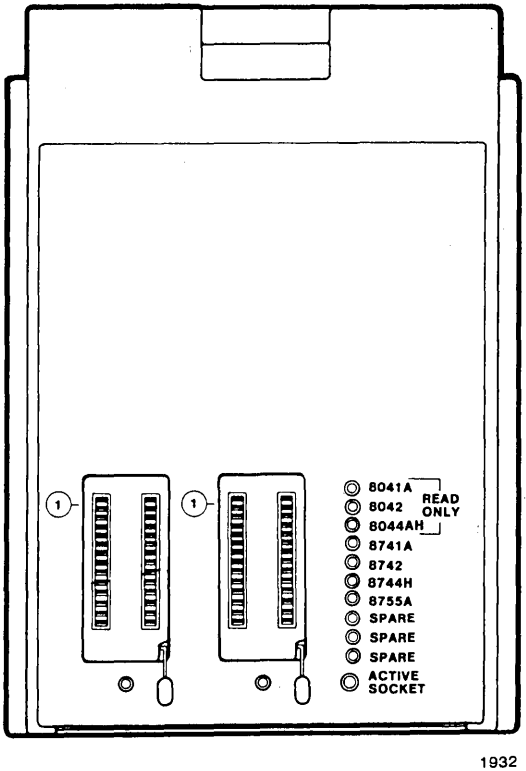



**Figure 6. iUP-F87/44A Personality Module**

280003-001

ACTIVE SOCKET LED lights when power is applied to the PROM device and when the universal programmer/iPDS system is accessing the selected socket.

## PROM PROGRAMMING EXAMPLE

The personality module is the interface that lets the user perform a wide variety of PROM programming, data display, and data editing operations. One of the most popular applications is copying data from a master PROM into a blank PROM. Table 3 outlines and compares the steps for both on-line and off-line copying. Notice the easy-to-use, English-language approach of the iPPS commands, which may be shortened to the first letter for faster entry.

The on-line example assumes that the universal programmer/iPDS system has been powered on and is under control of the ISIS software and that the iPPS software has been initialized. The off-line example assumes that the iUP-201A universal programmer has been powered on and initialized.

## PERSONALITY MODULE SPECIFICATIONS

### Memory

EPROM — up to 4K bytes

### Physical Characteristics

Width — 5.5 inches (1.4 cm)
Height — 1.6 inches ( 4.1 cm)
Depth — 7.0 inches (17.8 cm)
Weight — 1 pound (.45 kg)

### Electrical Characteristics

Maximum power consumption (module) —7.5 watts
Maximum power consumption (device) —2.5 watts
Maximum power consumption (total from PROM programmer) — 10 watts

### Environmental Characteristics

Reading temperature 10°C to 40°C
Programming temperature 25°C ± 5°
Operating humidity 10%-85% relative humidity

## DOCUMENTATION

Appropriate personality module user's guide:

164376 — *iUP-FAST 27/K Personality Module User's Guide*
162848 — *iUP-F27/128 Personality Module User's Guide*
164855 — *iUP-F87/51A Personality Module User's Guide*
164854 — *iUP-F87/44A Personality Module User's Guide*

**Table 3. Typical PROM Programming Sequence**

| Action | On-line Command | Off-line Function Key |
|---|---|---|
| 1. Select PROM type. | TYPE | DEVICE SELECT |
| 2. Install the PROM to be copied (the master PROM) in the personality module. | | |
| 3. Copy the contents of the master PROM to the buffer. | COPY PROM TO BUFFER | ROM TO RAM |
| 4. Verify that the copy was correct. | VERIFY | VER |
| 5. Remove the master PROM; install a blank PROM | | |
| 6. Copy the buffer to the blank PROM. | COPY BUFFER TO PROM | PROG |

280003-001

## ORDERING INFORMATION

| Part number | Description |
|---|---|
| iUP-Fast 27/K* | EPROM personality module |
| iUP-F27/128 | EPROM and $E^2$PROM personality module |
| iUP-F87/51A* | Microcontroller personality module |
| iUP-F87/44A* | Peripheral personality module |

*The iUP-Fast 27/K personality module and the security bit function on the iUP-F87/51A and iUP-F87/44A personality modules can be used with an iUP-200A/201A universal programmer; or an iUP-200/iUP-201 universal programmer upgraded to an A with the iUP-200/201 U1 upgrade kit; or an iPDS system, using version 1.4 or later of the iPPS-iPDS software (iPDS-140 units shipped after June 1984 contain this software).

intel

APPLICATION NOTE

AP-179

May 1984

# PROM Programming With the Intel Personal Development System (iPDS™)

**FRED MOSEDALE**
DSHO TECHNICAL PUBLICATIONS





Order Number 280015-001

Intel Corporation makes no warranty for the use of its products and assumes no responsibility for any errors which may appear in this document nor does it make a commitment to update the information contained herein.

Intel retains the right to make changes to these specifications at any time, without notice.

Contact your local sales office to obtain the latest specifications before placing your order.

The following are trademarks of Intel Corporation and may only be used to identify Intel Products:

BXP, CREDIT, i, ICE, $I^2$ICE, ICS, iDBP, iDIS, iLBX, $i_m$, iMMX, Insite, INTEL, $int_e$l, Intelevision, Intellec, $int_e$ligent Identifier™, $int_e$lIBOS, $int_e$ligent Programming™, Intellink, iOSP, iPDS, iRMS, iSBC, iSBX, iSDM, iSXM, Library Manager, MCS, Megachassis, Micromainframe, MULTIBUS, Multichannel™ Plug-A-Bubble, MULTIMODULE, PROMPT, Ripplemode, RMX/80, RUPI, System 2000, and UPI, and the combination of ICE, iCS, iRMX, iSBC, MCS, or UPI and a numerical suffix.

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

* MULTIBUS is a patented Intel bus.

Additional copies of this manual or other Intel literature may be obtained from:

Intel Corporation
Literature Department
3065 Bowers Avenue
Santa Clara, CA 95051

## INTRODUCTION

Programmable read-only memory (PROM) devices play a significant role in microprocessor-based products. How can PROM programming devices perform to best serve the needs of those who develop and service such products?

This application note first provides a general answer to this question; then, it proceeds to describe the features and use of PROM programming hardware and software available for the Intel Personal Development System (iPDS™). The description explains how the iPDS system provides the wide range of capabilities needed by those who use PROM programming devices. The description also highlights the iPDS system's ability to program some of Intel's newest EPROMs.

## PROMS IN THE LIFE CYCLE OF A PRODUCT

### Memory Options: A Review

Before PROM programming needs are discussed, it is important to briefly review memory options available to designers.

Microprocessor-based products need memory to store instructions and data used in controlling their operations. In order to maximize product operating speeds, designers must use memory that can be accessed quickly. Both random access memory (RAM) devices and read-only memory (ROM) devices offer designers quick access, but RAM devices are volatile—their contents are erased when system power is turned off. ROM devices are nonvolatile; thus, designers use ROMs to store programs and data that will not change during the operation of the product.

After a product's program code data has been debugged, you can transfer the code to programmable ROMs (PROMs) or masked ROMs. (The most flexible PROMs are $E^2$PROMs and EPROMs; $E^2$PROMs are electrically erasable and EPROMs can be erased by ultraviolet light.) PROMs are programmed using a relatively simple procedure; by contrast, masked ROMs can only be programmed in a manufacturing environment. Thus, masked ROMs provide less flexibility but are used because they may be more cost-effective in large volumes. However, because the price of PROMs is falling and because inventories with erasable PROMs can be reprogrammed when product programs are changed, erasable PROMs are also attractive for large volumes.

### Desirable PROM Programming Features

Once your product's software is debugged, you can load the software into the product's PROMs (so that it becomes the product's firmware). However, usually in the development of a product, the initial programming of PROM devices is not the last operation involving the PROMs. Even if the software is debugged, once it is loaded into the PROMs, you may discover new bugs in the program that you failed to detect before the program was committed to PROMs. So, there may soon be a need to erase the PROMs (if they are EPROMs or $E^2$PROMs) and reprogram them.

During product development and servicing, you will also sometimes need to accomplish the following tasks:

- Check the contents of a PROM.
- Use one PROM to program other PROMs that will be used in other prototype systems.
- Update earlier firmware versions with later versions.

Consider in more detail the (P)ROM-related needs that can arise for you during the product's life cycle, that is, between the time when the product's software has been loaded into PROMs and the time when the product is phased out. There are two basic sets of needs, those having to do with displaying (P)ROM contents and those having to do with programming PROMS.

#### DISPLAYING AND PRINTING NEEDS

For a variety of reasons, you may need to examine what is stored in a (P)ROM. For example, you may suspect that a (P)ROM's program is in error; or you may have incomplete documentation on what was programmed into a (P)ROM. Thus, you will want to be able to display the contents on a video display terminal and to have a printer print out the contents. You will want to be able to choose the display base (binary, octal, decimal, or hexadecimal) and whether to display the contents as ASCII characters.

#### PROGRAMMING NEEDS

When you program PROMs, you need a programming device that can program a variety or PROMS and one that offers flexibility and ease of programming. The following list describes PROM programming needs.

- **Need for simple operation** — You want a programming device that satisfies all of the following needs and is simple to operate. You do not want to have to refer to a manual every time you wish to program a PROM.

- **Need to program a wide variety of PROMs** — For greatest flexibility, you want a programming device that can program the various kinds of PROMs that are available. For example, you will want to be able to program microcontrollers with EPROMs, and you will want to be able to program from the small inexpensive 16K and 32K PROMS to the latest 256K PROMS with int$_e$ligent Programming™ algorithms to speed programming. You will also want to be able to program those PROMs that use the new lower programming voltage (12.5 V).

- **Need to be upgradeable** — A PROM programming device should be designed so that it can be upgraded to program PROMs that will be available in the future. Without upgradeability, the device will soon be out-of-date.

- **Need to check PROM contents before programming** — If the PROM you will be programming is blank, it can of course be programmed. Even if it has some bits set at the time of programming, if the same bits must also be set for the program, the PROM can be used. Thus, ideally, a PROM programming device will determine whether the PROM is blank, and if not, determine whether the bits already set are compatible with the program to be loaded into the PROM.

- **Need to recognize file formats** — When a PROM is programmed using an object file generated by a compiler or assembler, the PROM programming device must be able to extract the data that is to be loaded into the PROM from the larger file structure. For greatest flexibility, you will need a PROM programming device that recognizes the file structures generated by compilers and assemblers that you will use when developing program code for the PROMs.

- **Need to support a variety of source program options** — There are three sources you may wish to use for PROM data: an already-programmed PROM, a software development system, or a disk. For greatest programming flexibility, you will want a PROM programming device that makes all three kinds of sources available.

- **Need to support data manipulation and modification** — You may wish to modify a source file for your PROM program. For example, you may discover an error in the source file, or you may realize that for your new processor system, the PROM data must first be 2's complemented. For a variety of reasons, your PROM programming will be much more flexible if the PROM programming device offers a buffer for temporary storage and PROM programming software that can manipulate the source program data in a variety of ways.

- **Need for variety in loading program code into PROMs** — If your product has a 16-bit microprocessor and you are using 8-bit PROMs to store the firmware, you will need to interleave the 16-bit code between two 8-bit PROMs. A PROM programming device with interleaving capability will speed such programming. You may want other kinds of flexibility when programming.

- **Need to transfer long programs that will not fit into one PROM** — Long programs may exceed the storage capacity of the PROMs chosen for your product. You need a programming device that can format the program so that it can be stored in successive PROMs.

- **Need to verify programming** — When programming is finished, you will want to check that the PROM is indeed correctly programmed. A defect in the PROM could corrupt the intended firmware. Checking would involve comparing the source with the programmed PROM.

- **Need to compare buffer with PROM** — If you are interrupted when programming PROMs or if you have not labeled PROMs that you did program, you may forget whether a particular PROM was programmed. In such cases, you will want to compare the particular PROM with the program stored in the buffer of the programming device. Comparison will prevent you from having to reprogram an already-programmed PROM.

- **Need to lock microcontrollers from unauthorized access** — Some advanced microcontrollers can be locked to prevent unauthorized access. To take advantage of this security feature, you need to be able to control the locking of the microcontrollers.
- **Need to automate routine PROM programming tasks** — To speed programming, you will want a programming device that can automate routine programming functions. Automation will not only speed programming, it will also release personnel for other work.

### DESIRABLE PROM PROGRAMMING FEATURES: A SUMMARY

In summary, if PROMs (and ROMs) are incorporated in your product, you will have the greatest flexibility if your PROM programming device has the following features. (Of course, for ROMs only reading tasks are needed.)

- Is easy-to-use.
- Can program a wide variety of PROMs.
- Is upgradeable.
- Can display (P)ROM contents in ASCII characters and in a variety of bases.
- Can enable a printer to print out (P)ROM contents in ASCII characters and in a variety of bases.
- Can check for blank PROMs.
- If PROM is not blank, can check PROM contents for compatibility with program.
- Can recognize file formats of your development system object files.
- Supports transfers of program code from development systems, disks, and other PROMs to the new PROM.
- Provides temporary storage and software for manipulating Programming data before loading it into the PROM.
- Supports variety in how data is loaded into PROMs, e.g.:
  - Interleaving 16-bit data into 8-bit PROMs
  - Segmenting long programs so that resulting program segments fit into successive PROMs
- Can verify the accuracy of copying.
- Can compare programming buffer with PROM contents.
- Can lock microcontrollers from unauthorized access.
- Can automate routine PROM programming tasks.

The Intel Personal Development System (iPDS) with the PROM programming option meets these needs. The following sections describe the iPDS PROM programming hardware and software and show how this system can perform all of these tasks for a variety of PROMs.

## THE iPDS™ PROM PROGRAMMING SYSTEM

The iPDS system supports integrated hardware and software development; it provides a complete set of software development tools and in-circuit emulators for hardware debugging and hardware-software integration. With its optional PROM programming hardware and software, the iPDS system also supports PROM programming.

Three components comprise the iPDS PROM programming system: the iPDS system (with the ISIS-PDS operating system and the plug-in module adapter board), the PROM programming modules, and the PROM programming software. Each of these components is described briefly in the following sections.

### iPDS™ System

To perform PROM programming tasks, the iPDS system must use its ISIS-PDS operating system and the plug-in module adapter board. The PROM programming software (iPPS-PDS) runs under the ISIS-PDS operating system.

The adapter board allows you to use both the PROM programming personality modules and emulation modules. It provides the interface between the modules and the iPDS system.

Figure 1 shows the iPDS system with a PROM programming personality module plugged into its side.

280015-001

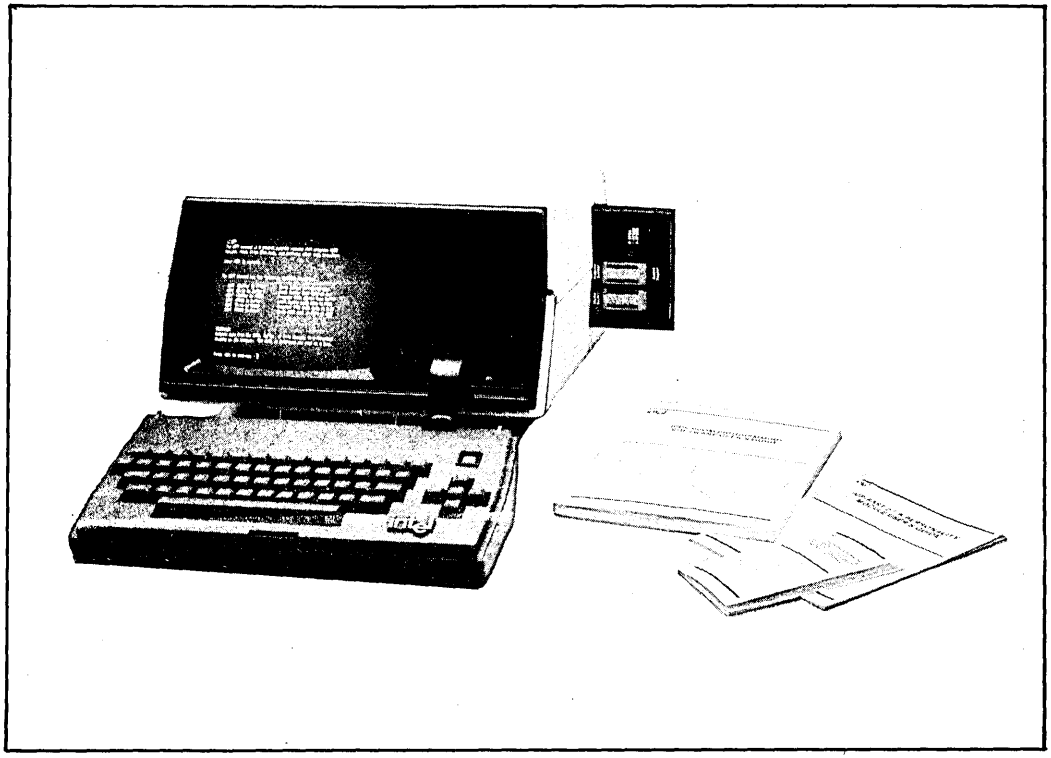


**Figure 1 iPDS™ System with PROM Programming Personality Module**

## PROM Programming Personality Modules

A personality module is the interface between the iPDS system and a selected PROM. Personality modules contain all the hardware and firmware for reading and programming a family of Intel devices. Each personality module is a single molded unit inserted into the side panel of the iPDS unit. No additional adapters or sockets are needed. Table 1 lists the available personality modules, and Figure 2 shows the four modules.

**Table 1 PROM Programming Personality Modules**

| PERSONALITY MODULE | PROM TYPE PROGRAMMED | PROMs AND ROMs SUPPORTED |
|---|---|---|
| iUP-Fast 27/K | EPROM | 2764, 2764A, 27128, 27256, and provisions for future PROMs |
| iUP-F27/128 | $E^2$/EPROM | 2716, 2732, 2732A, 2764, 27128, 2815, and 2816 |
| iUP-F87/51A | Microcontroller | 8748, 8748H, 8048, 8749H, 8048H, 8049, 8049H, 8050H, 8751, 8751H, 8051 |
| iUP-F87/44A | Peripheral | 8741A, 8041A, 8742, 8042, 8744H, 8044AH, 8755A |



280015-001


**Figure 2 PROM Programming Personality Modules**

Each personality module connects to the iPDS system through a 41-pin connector. Module firmware is uploaded into the iPDS system and executed by the iPDS system. The personality module firmware contains routines needed to read and program a family of PROMs. In addition, the personality module sends specific information about the selected PROM to the iPDS system, such as information about the PROM size and its blank state.

LEDs on each personality module indicate its operational status. On some personality modules a column of LEDs or a hexadecimal display indicates which PROM device type the user has selected. On some personality modules with more than one socket, an LED below each socket indicates the socket to be used. In addition, a red indicator light tells the user when power is being supplied to the selected device.

The personality module firmware performs selected PROM tests and indicates status:

- The PROM installation test verifies that the device is installed in the module correctly and that the ZIF socket is closed.
- The PROM blank check determines whether the device is blank. The iPDS system automatically determines whether the blank state for the particular device is defined as all zeros or all ones.
- The overlay check (performed when a PROM is not blank) determines which bits are programmed, compares those bits against the program to be loaded, and allows programming to continue if they match.

Easy-to-read status messages are also provided. The user can invoke all of the PROM device integrity checks except the installation test (which occurs automatically any time an operation is selected). The following sections describe specific features of the three personality modules that program the newer Intel PROMs.

280015-001

### iUP-F87/44A AND iUP-F87/51A PERSONALITY MODULES: SPECIAL FEATURE

Each of these personality modules supports the security bit function on one member of the microcontroller family it can program. The iUP-F87/44A module supports the function on the 8744H microcontroller, and the iUP-F87/51A supports the function on the 8751H microcontroller. The KEYLOCK command locks the 8744H (or the 8751H) EPROM memory from unauthorized access by setting the security bit; the microcontroller cannot be unlocked without erasing the EPROM. As a safety precaution, the KEYLOCK command requires user verification before it sets the security bit.

### iUP-FAST 27/K PERSONALITY MODULE: SPECIAL FEATURES

The iUP-Fast 27/K personality module supports the int$_e$ligent Identifier™ and the int$_e$ligent Programming algorithms. The int$_e$ligent Identifier is used to check the PROM installed in the personality module socket to determine whether it matches the type selected; then the int$_e$ligent Identifier is used to select the proper int$_e$ligent Programming algorithm. The int$_e$ligent Programming algorithms reduce PROM programming time by as much as a factor of ten. This module has provision for support of future EPROMS and E$^2$PROMs via simple plug-in updates.

### The int$_e$ligent Programming™ Algorithm

Using the capabilities of the iPDS PROM programming equipment and employing a new kind of algorithm that recognizes differences among EPROM cells, you can dramatically reduce programming time for the newest high-density EPROMs. As a bonus, the technique helps ensure that EPROMs receive adequate programming — in terms of memory-cell charge — to maintain long-term reliability.

Reducing programming time and costs for EPROMs has become increasingly important because the chips have become a cost-effective, easy-to-use alternative to masked ROM in high-volume applications requiring code flexibility or simplified inventory — a major switch from EPROMs' original small-volume prototyping applications. And, volume usage makes EPROM programming a significant manufacturing consideration.

The conventional programming procedure for most EPROMs uses a nominal 50-msec pulse per EPROM byte, resulting in a total programming time of approximately 1.5 minutes for a 16K-bit chip. With the introduction of the 2764 (64K bits) and devices with even higher densities, however, programming times have increased. A 256K-bit EPROM, for example, requires 24 minutes for programming using the conventional programming method.

Most EPROM cells program in less than 45 msec, however. In fact, empirical data shows that very few cells require longer than 8 msec for programming. Therefore, a procedure that takes into account the characteristics of individual EPROM cells can significantly reduce a device's programming time.

Arbitrarily reducing programming time is risky, however, because a cell's ability to achieve and maintain its programmed state is a function of this time. What is needed, therefore, is a way to verify the level to which individual cells have been programmed. Such a way exists. By determining the charge stored in a cell compared to the minimum charge needed to program the cell to a detectable level, you can check for a program margin that ensures reliable EPROM operation.

Margin checking does not occur in conventional EPROM programming, however. Instead, each EPROM cell receives a 45- to 55-msec write pulse, and manufacturers attempt to ensure program margin by screening out EPROMs having bytes that do not program within 45 msec. This programming procedure is thus an open loop — no actual verification of margin occurs.

By contrast, the int$_e$ligent Programming algorithm guarantees reliability through the closed-loop technique of margin checking. This algorithm uses two different pulse types: initial and over-program. The algorithm first applies a 1-msec initial pulse to an EPROM. After the pulse, it checks the EPROM's output for the desired programmed value. If the output is incorrect, the algorithm repeats the pulse-and-check operation. When the output is correct, the algorithm supplies an over-program pulse; the length of this pulse depends on how many initial pulses were used and varies with the EPROM being programmed. This longer pulse helps ensure that the EPROM cell has an adequate programming margin for reliable operation.

280015-001

## Prom Programming Software (iPPS-PDS)

The iPPS-PDS software provides easy-to-use commands that allow you to load programs into a target PROM from another PROM, from iPDS system memory, or directly from a disk file.

The iPPS-PDS software also supports data manipulation in the following Intel formats: 8080 hexadecimal ASCII, 8080 absolute object, 8086 hexadecimal ASCII, 8086 absolute object, and 286 absolute object. Addresses and data can be displayed in binary, octal, decimal, or hexadecimal. You can easily change default data formats as well as number bases.

You invoke the iPPS-PDS software from the ISIS operating system. (The software can be run under control of ISIS submit files, thereby freeing you from repetitious command entry.)

An explanation of the iPPS-PDS software follows. It is divided into three main sections: the iPPS-PDS storage devices, iPPS-PDS commands, and invoking the iPPS-PDS. Also see the Appendix for iPDS PROM programming examples.

### iPPS-PDS STORAGE DEVICES

The iPPS-PDS software transfers data between any two of the three storage devices: PROM, buffer, and file. These devices are defined in the following three sections.

#### PROM Device

The PROM device is plugged into a socket on the personality module installed in the iPDS system. The iPPS-PDS software does not recognize the PROM device until you enter the TYPE command. The TYPE command automatically sets the appropriate buffer size according to the size of the PROM device specified.

#### Buffer Device

The buffer device is a section of development system memory that the iPPS-PDS software allocates and uses as a working area for temporary storage and for rearranging data. Its boundaries can exist anywhere in a virtual address range from 0 to 16777215 (0 to $2^{24}$-1).

When the iPPS-PDS software is initialized, the buffer starting address is set to 0 and the buffer ending address is set to 8K−1, providing an initial buffer size of 8K bytes (the default buffer size when no PROM type is specified). During subsequent iPPS-PDS operations, the size and boundaries can vary. Specific iPPS-PDS commands determine these variations. The most recent command that changed the lower boundary of the buffer determines the buffer starting address. The TYPE command affects both the size and location of the buffer. For example, the TYPE command always resets the buffer start address to 0. The most recent TYPE command controls the size of the buffer.

The iPDS system needs a virtual buffer when PROM size exceeds 8K. If the PROM size exceeds the 8K memory buffer space available on the development system, the iPPS-PDS software creates a virtual buffer area using temporary file space on disk.

Two temporary work files are used to create the virtual buffer. During subsequent virtual buffer operations, the iPPS-PDS software automatically swaps data in and out of development system memory from and to work files.

#### File Device

The file device is an ISIS file on a disk. It is specified within iPPS-PDS commands.

The data stored in the disk file is in one of the following Intel absolute formats: 8080 hexadecimal, 8080 object, 8086 hexadecimal, 8086 object, or 80286 object. The iPPS-PDS software can read any of these formats as input but writes data to a file in 8080 object, 8086 object, or 80286 object formats only. Basically, these files contain representations of blocks of memory data. Included with the data are addresses for the locations of the data. The data blocks are not necessarily in consecutive address order. The method used to create the file determines the order of the data.

The iPPS-PDS file device has address boundaries that exist in the virtual range from 0 to 16777215 (0 to $2^{24}$-1). These boundaries are determined as follows:

- The file's lowest address is the lowest address encountered while reading the file.
- The file's highest address is the highest address encountered while reading the file.

If the iPPS-PDS software creates the file (that is, if the file is a destination device in an iPPS-PDS command), the specific command issued determines these boundaries.

When you specify a particular address range to be read from a file, all sections in the address range that are not present in the file are written in a PROM destination device as the blank state of the currently selected PROM type. If the destination device is the buffer, the nonexistent sections in the file do not overwrite the corresponding sections in the buffer.

During the operation of commands that use the file device as a source, the iPPS-PDS software only reads the actual data from the file and ignores any other information in the file. For example, the file can contain special information used later for debugging. Since the iPPS-PDS software ignores this information, it will not appear in any new files generated. If the data is written back to the original file, the original file is deleted.

### iPPS-PDS COMMANDS

Each iPPS-PDS command consists of a keyword that identifies the command, followed by other keywords and associated parameters that are the arguments of the command. You enter all iPPS-PDS commands, as well as program address and data information, through the development system ASCII keyboard; the commands are displayed on the system CRT. Table 2 summarizes the iPPS-PDS commands.

**Table 2 iPPS-PDS Command Summary**

| COMMAND | DESCRIPTION |
|---|---|
| **PROGRAM CONTROL GROUP** | **CONTROLS EXECUTION OF THE iPPS-PDS SOFTWARE.** |
| EXIT | Exits the iPPS software and returns control to the ISIS operating system. |
| <ESC> | Terminates the current command. |
| REPEAT | Repeats the previous command. |
| ALTER | Edits and re-executes the previous command. |
| **UTILITY GROUP** | **DISPLAYS USER INFORMATION AND STATUS; SETS DEFAULT VALUES.** |
| DISPLAY | Displays PROM, buffer, or file data on the console. |
| PRINT | Prints PROM, buffer, or file data on the local printer. |
| HELP | Displays user assistance information. |
| MAP | Displays buffer structure and status. |
| BLANKCHECK | Checks for unprogrammed PROMs. |
| OVERLAY | Checks whether non-blank PROMs can be programmed. |
| TYPE | Selects the PROM type. |
| INITIALIZE | Initializes default number base and file type. |
| WORKFILES | Specifies the drive device for temporary work files. |
| **BUFFER GROUP** | **EDITS, MODIFIES, AND VERIFIES DATA IN BUFFER.** |
| SUBSTITUTE | Examines and modifies buffer data. |
| LOADDATA | Loads a section of buffer with a constant. |
| VERIFY | Verifies data in the PROM with buffer data. |
| **FORMATTING GROUP** | **REARRANGES DATA FROM PROM, BUFFER, OR FILE.** |
| FORMAT | Formats and interleaves buffer, PROM, or file data. |
| **COPY GROUP** | **COPIES DATA FROM ONE DEVICE TO ANOTHER.** |
| COPY (file to PROM) | Programs PROM with data in a file on disk. |
| COPY (PROM to file) | Saves PROM data in a file on disk. |
| COPY (buffer to PROM) | Programs PROM with data from the buffer. |
| COPY (PROM to buffer) | Loads the buffer with data in the PROM. |
| COPY (buffer to file) | Saves the contents of buffer in a file on disk. |
| COPY (file to buffer) | Loads the buffer from a file on disk. |
| **SECURITY GROUP** | **LOCKS SELECTED DEVICES; PREVENTS UNAUTHORIZED ACCESS.** |
| KEYLOCK | Locks the PROM from unauthorized access. |

280015-001

Once entered, a command line is verified for correct syntax and executed. If a syntax error is detected, the following error message is displayed:

--SYNTAX ERROR--*specific error.*

If you omit a required keyword, the iPPS-PDS software prompts for the keyword and its associated parameters. If the keyword is entered but its parameters are omitted, either a default value is assumed or an error message is displayed if there is no default. In certain commands, default keywords are also assumed.

You can enter complete iPPS-PDS keywords or any unique abbreviation (only the first character is required). For example, command keywords of C, CO, COP, and COPY are all interpreted as the COPY command.

The iPPS-PDS software accepts numeric entries in any one of four number bases: binary (Y), octal (O or Q), decimal (T), or hexadecimal (H). Numbers can be entered in any of these bases by appending the appropriate letter identifier to specify the base (e.g., 11111111Y, 377Q, 255T, FFH). An explicit number base identifier overrides the default number base, which is initially hexadecimal.

### INVOKING iPPS

There are two methods of invoking the iPPS-PDS software: command lines and submit files.

The command line for invoking the iPPS-PDS software (under V1.0 and later versions of the ISIS.PDS operating system) uses the following syntax:

[:F*n*:]IPPS

The symbol ":F*n*:" Specifies the drive on which the iPPS-PDS files are located. When you enter the iPPS-PDS command, the ISIS operating system loads and executes the iPPS-PDS software.

The iPPS-PDS software can also run under the control of a submit file. SUBMIT is an ISIS command that allows you to use a disk text file as input for further ISIS commands or as command inputs to utilities running under the ISIS operating system. Thus, a submit file can contain the ISIS command line to invoke the iPPS-PDS software and then a sequence of commands for the iPPS-PDS software itself.

## Summary: The iPDS™ System Meets PROM Programming Needs

Table 3 describes briefly how the iPDS system meets each of the needs identified earlier in this application note.

The iPDS system can be a complete intelligent PROM programmer — and, because the iPDS system is also a development system, it can provide an excellent means to off-load PROM programming from your current development system (just as the iPDS system allows you to off-load other 8-bit development tasks). In addition, with its state-of-the-art PROM programming capability, the iPDS system becomes an attractive solution to your complete development system needs.

**Table 3 iPDS™ Features Meet PROM Programming Needs**

| NEED | iPDS™ FEATURE |
|---|---|
| Be easy-to-use. | iPPS software and the PROM programming personality modules were designed to provide ease-of-use. |
| Program a wide variety of PROMs. | Personality modules each permit the programming of a family of PROMs or microcontrollers. |
| Be upgradeable. | New personality modules will be released as new PROM families appear. |
| Display (P)ROM contents in ASCII characters or in a variety of bases. | iPPS DISPLAY command displays (P)ROM (or buffer or file) contents in ASCII characters and in binary, octal, decimal, or hexadecimal. |
| Enable a printer to print out (P)ROM contents in ASCII characters and in a variety of bases. | iPPS PRINT command prints out (P)ROM (or file or buffer) contents in ASCII characters and in binary, octal, decimal, or hexadecimal. |
| Check for blank PROMs. | iPPS BLANKCHECK command checks for blank PROMS. |
| If PROM is not blank, check PROM contents for compatibility with program. | iPPS OVERLAY command checks PROM contents for compatibility with program. |
| Recognize file formats of development system object files. | iPPS command file switch allows you to indicate to the iPDS system which object file format is being used. |
| Support transfers of program code from development system, disks, and other PROMs to the new PROM. | iPPS COPY commands allow you to copy in either direction between the iPDS disk drive(s), PROMs, and the iPDS buffer storage. |
| Provide temporary storage and software for manipulating programming data before loading it into the PROM. | iPDS buffer provides temporary storage and the iPPS SUBSTITUTE and LOADDATA commands allow you to manipulate programming data before you load it into a PROM. |
| Load data into PROMs in a variety of formats, e.g.:<br>— interleaving 16-bit data into two 8-bit PROMs<br>— segmenting long programs so that resulting program segments fit into successive PROMs | iPPS FORMAT command allows you to format data in a variety of ways so that it can be loaded into PROMs in various sequences (including interleaving and segmenting). |
| Verify the accuracy of copying. | iPPS software automatically checks the accuracy of copying. |
| Compare programming buffer with PROM contents | iPPS VERIFY command compares buffer data with PROM data. |
| Control the security feature of advanced microcontrollers for unauthorized access. | iPPS KEYLOCK command locks advanced microcontrollers. |
| Automate routine PROM programming tasks. | ISIS SUBMIT files permit you to store frequently used command sequences. The files can then be activated with a single command. |

280015-001

# APPENDIX: PROM PROGRAMMING EXAMPLES

## APPENDIX: PROM PROGRAMMING EXAMPLES

Displaying (P)ROM contents and programming PROMs are easy tasks with the iPDS system. The following four examples show typical uses of the iPDS system's PROM programming capabilities:

- Examining the contents of a masked ROM
- Duplicating a PROM
- Interleaving a file between two PROMs
- Locking a microcontroller

## EXAMPLES

The examples assume that the iPDS system is under control of the iPPS-PDS software. The boldface characters shown on the iPDS screen displays indicate user entries. The key-in sequence below each screen display gives the actual entries that you must key in to obtain the screen display.

### Examining the Contents of a Masked ROM

The DISPLAY command lets you examine the contents of a PROM or a masked ROM.

```
PPS> DISPLAY PROM
000000: C3 40 00 20 20 44 20 2D 20 44 49 53 4B 00 20 20  .@. D - DISK.
000010: 47 20 2D 20 47 45 4E 45 52 41 4C 00 20 20 4B 20  G - GNENERAL.  K
000020: 2D 20 4B 45 59 42 4F 41 52 44 2F 43 52 54 00 FF  - KEYBOARD/CRT..
000030: FF FF FF FF FF FF FF FF C3 36 1C FF FF FF FF FF  .........6......
000040: F3 DB 80 E6 20 CA 03 08 3E 00 D3 D1 DB 80 E6 01  ........>.......
000050: C2 66 00 3E 4F D3 D0 3E 58 D3 D0 3E 89 D3 D0 3E  .f.>O..>X..>...>
000060: 99 D3 D0 C3 76 00 3E 4F D3 D0 3E 98 D3 D0 3E 8A  ....v.>O..>...>.
000070: D3 D0 3E 9C D3 D0 21 00 00 11 00 08 AF 47 7B B2  ..>...!......G{.
000080: CA 8A 00 78 86 23 18 C3 7D 00 78 FE 55 C2 8D 00  ...x.#..}.x.U...
000090: 3E 34 D3 E3 3E 1F D3 E0 3E 00 D3 E0 01 30 00 DB  >4..>...>....0..
0000A0: 80 E6 01 C2 A9 00 01 2C 00 3E 72 D3 E3 79 D3 E1  .......,.>r..y..
0000B0: 78 D3 E1 3E B2 D3 E3 3E 00 D3 E2 3E 16 D3 E2 D3  x..>...>...>....
0000C0: 10 3E 22 D3 60 D3 50 DB 80 E6 04 CA C7 00 DB 80  .>".`.P.........
0000D0: E6 04 C2 CE 00 AF D3 F0 D3 F0 D3 F0 D3 F1 3E A1  ..............>.
0000E0: D3 F8 3E 23 D3 60 3E C8 D3 E2 3E 00 D3 E2 D3 50  ..>#.`>...>....P
0000F0: 21 EF 00 2B 7C B5 C2 F3 00 DB 80 E6 04 C2 FD 00  !..+|...........
000100: 3E 00 D3 E2 3E 16 D3 E2 D3 50 DB 80 E6 04 CA 0A  >...>....P......
000110: 10 DB 80 E6 04 C2 11 01 3E 22 D3 60 D3 50 DB 80  ........>".`.P..
000120: E6 04 CA 1E 01 DB 80 E6 04 C2 25 01 21 00 40 11  ..........%.!.@.
ENTER <CR> TO CONTINUE $
ABORTED
PPS>
```

**Key-in Sequence**

**DISPLAY PROM**

RETURN

ESC

**Comments**

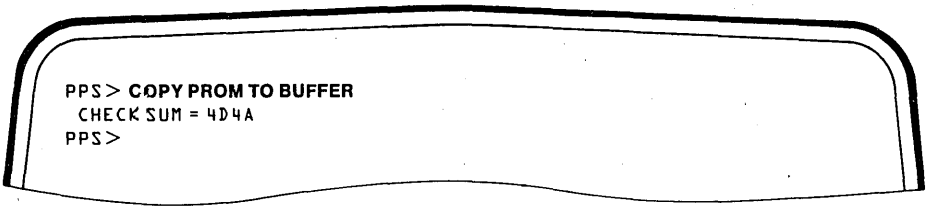
This example shows the data in the PROM in hexadecimal format, which is the default base in this example. Press the ESC key at any time to end the display. The "S" sign is the echo of the ESC key. You can also display the data in other number bases. Note the ASCII code displayed in the far right column.

280015-001

## Duplicating a PROM

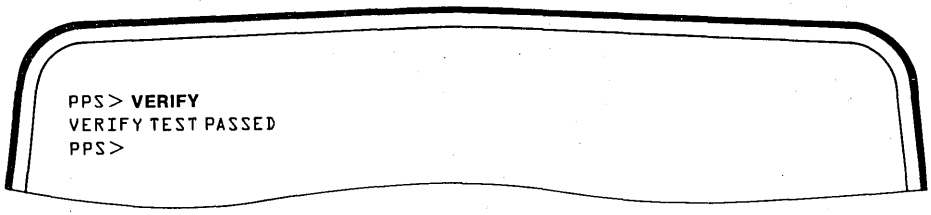
One frequently used application of iPDS PROM programming is copying data from a PROM into a buffer or file, then copying it into another PROM. You can perform this operation using the iPPS-PDS buffer (or an iPDS file for intermediate storage) and the iPPS-PDS COPY commands.

The following example illustrates a direct PROM-to-buffer-to-PROM duplication. If you wish to perform these examples, place the PROM in the PROM socket and reset the iPPS-PDS (using the TYPE command for your type of PROM). A 2716 EPROM that contains sample code is used in this example.




| Key-in Sequence | Comments |
|---|---|
| **COPY PROM TO BUFFER** RETURN | This command copies every memory location in the PROM to the buffer beginning at destination address 00H in the buffer. The checksum is the 2's complement of the 16-bit sum of all the bytes read. |

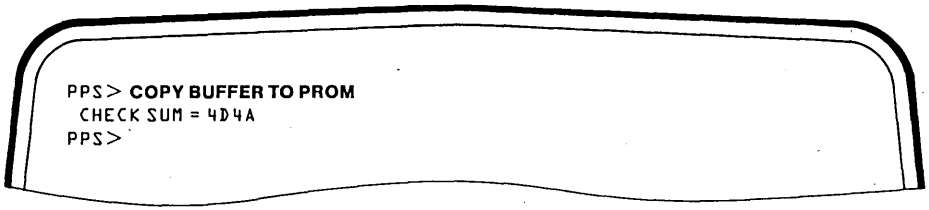
If you want to check the buffer to be sure the data now there matches the original data in the PROM, one command is all that is needed. Enter the VERIFY command, and if the buffer and PROM data match, you will be informed VERIFY TEST PASSED.




| Key-in Sequence | Comments |
|---|---|
| **VERIFY** RETURN | The data in the buffer matches the data in the PROM. |

280015-001

Now that you have verified that the data in the buffer matches the data in the PROM, you are ready to copy the buffer to a blank PROM. Remove the master PROM from the PROM socket and insert the blank PROM. Then use COPY again to copy the contents of the iPPS-PDS buffer to the blank PROM.




| Key-in Sequence | Comments |
|---|---|
| **COPY BUFFER TO PROM** RETURN | The display of the check-sum and the return of the iPPS prompt indicate that the PROM was successfully programmed. |

Note that for copying from the buffer to a PROM, you do not need to use the VERIFY command. The iPPS-PDS software automatically verifies the copying when you copy in this direction.

### Interleaving a File Between Two PROMS

It is often desirable to have code or data arranged in 16-bit words and stored on a pair of 8-bit PROMs. This is the case, for example, when working with an 8086 microprocessor that reads from and writes to memory on a 16-bit data bus. The data is interleaved between two PROMs, the odd (or low) bytes stored in one PROM and the even (or high) bytes stored in the other PROM. The FORMAT command handles this interleaving automatically.

In the following example, a file written in Intel 8086 hexadecimal format is interleaved into two PROM devices.

```
PPS>FORMAT DOUBLE.BYT (0,FFFH)
LOGICAL UNIT (BIT=1,NIBBLE=2,BYTE=3,N-BYTE=4)
LU = 3
INPUT BLOCK SIZE (N BYTES)
N = 2
OUTPUT BLOCK SIZE (N BYTES)
N = 1
INPUT BLOCK STRUCTURE:
NUMBER OF INPUT LOGICAL UNITS = 002

LSB
--------,
| 00 | 01 |
--------
NUMBER OF OUTPUT LOGICAL UNITS = 001
OUTPUT SPECIFICATION (<CR> TO EXIT):
*
```

**Key-in Sequence**

**FORMAT DOUBLE.BYT (0,FFFH)** RETURN

3 RETURN

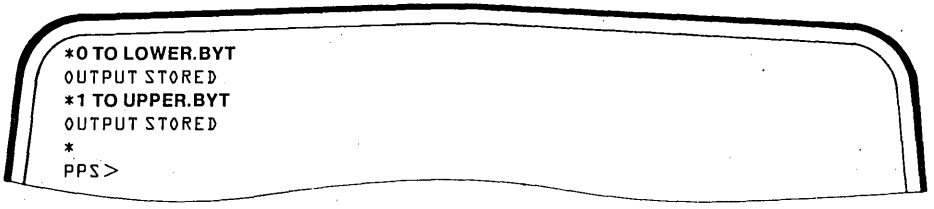
2 RETURN

1 RETURN

**Comments**

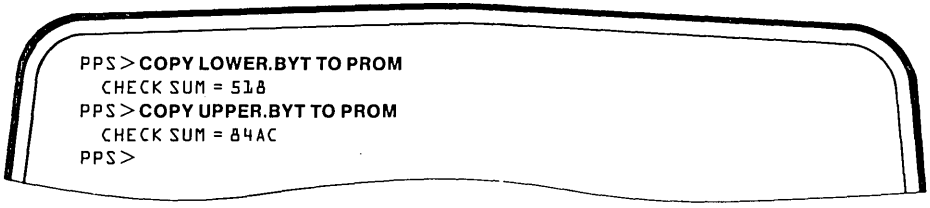
In this example, a file called DOUBLE.BYT is split into two files, with alternate bytes being loaded into alternate files. After establishing the FORMAT command and the file name with the first entry, the iPPS software prompts for the size of the logical unit that is going to be manipulated. Byte is selected as the logical unit. You are then prompted to set up the input block size (in this case two bytes) and the output block size (one byte). A diagram of the input block is displayed with the logical units labeled. The least significant bit in the input block is displayed with the logical units labeled. The least significant bit in the input block is shown on the left. The number of logical units in the output block is also displayed. You are then prompted with an asterisk (*) to enter the output specification.

280015-001




| Key-in Sequence | Comments |
|---|---|
| **0 TO LOWER.BYT** RETURN<br>**1 TO UPPER.BYT** RETURN<br>RETURN | Once the size of the logical unit, the input block size and the output block sizes have been established, you are prompted for the output specification (how you want the data in the file to be manipulated in terms of logical units). This example specified that the least significant byte in each input block be stored in a file titled LOWER.BYT in the default drive. The iPPS software then sorts through the DOUBLE.BYT file. Next it specifies that the most significant byte be stored in a file titled UPPER.BYT. The iPPS software then sorts through the DOUBLE.BYT file and copies every odd byte to the UPPER.BYT file. OUTPUT STORED is displayed after each output specification is implemented. You then have the option of entering another output specification. Pressing RETURN exits the FORMAT command and returns the iPPS prompt. |

You can use the two files created with this FORMAT operation to program two PROMs, which you can then install in parallel to provide 16-bit data words to a 16-bit microprocessor. To copy the files to the PROMs, use the COPY command as follows.

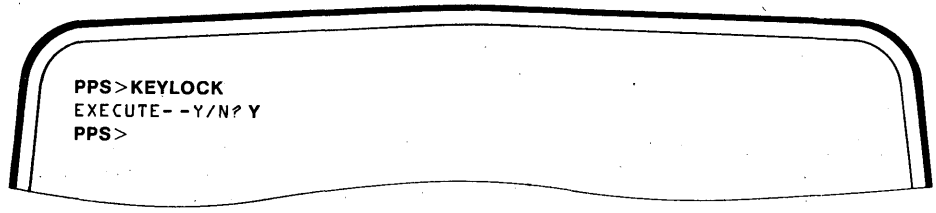



| Key-in Sequence | Comments |
|---|---|
| **COPY LOWER.BYT TO PROM** RETURN<br>**COPY UPPER.BYT TO PROM** RETURN | You must install a blank PROM in the personality module before entering each COPY command. |

280015-001

## Locking a Microcontroller

After programming a microcontroller, you can protect it from unauthorized access by locking it with the KEYLOCK command (the KEYLOCK command cannot be used with all EPROMs). The following example locks an 8751H microcontroller, which then cannot be unlocked without erasing it.




| Key-in Sequence | Comments |
|---|---|
| KEYLOCK RETURN | Entering Y locks the EPROM. If you enter N, the command terminates and EPROM remains unlocked. |
| Y RETURN | |

280015-001

# System Design Kits

7