MIKROCOMPUTER-SYSTEME SMS

SPRINT™

The Universal Programmer for your Personal Computer

OPERATION MANUAL

October 1989

(Updated February 1990)

SPRINT EXPERT and **SPRINT^{plus}** are registered trademarks.

Im Morgental 13
8994 Hergatz-Schwarzenberg
West Germany
Telephone (49) 07522 5018
Fax (49) 07522 8929

PAL (tm) is a registered trademark of Advanced Micro Devices Inc.

CONDITIONS OF SALE

SMS GmbH and PDC Ltd grant to the purchaser a single user, non-transferable, non-exclusive license for the use of this **SPRINT** software and hardware. The licensee may not transfer, assign or sub-license either the software, manuals or hardware of **SPRINT**. This license is granted on an 'AS IS' basis. Neither SMS GmbH nor PDC Ltd make any warranties, either express or implied, regarding the application or operation of **SPRINT**. Under no condition will either SMS GmbH or PDC Ltd be held liable for any direct, indirect, special or consequential damages caused or alleged to be caused directly or indirectly by **SPRINT**, including but not limited to loss of business, anticipation of profits, interruption of services, injury or physical damage. The liability for all effects of devices programmed by **SPRINT** rests completely with the licensee. Any disputes will be settled by the courts of West Germany.

TABLE OF CONTENTS

2. SPRINT PLD FUNCTIONS

3. SOCKET PODS AND ADAPTERS

4. PLDASM

5. SPRINT UN-ASSEMBLER OPERATION

APPENDICES

A. TECHNICAL NOTES

B. INSTALLATION

C. USER DOCUMENTATION

INTRODUCTION

SPRINT OVERVIEW

Congratulations on purchasing this **SPRINT** programming tool. You will now experience the ease of use and flexibility that has made the **SPRINT** family the leading 'personal programmer' for professional users. With the **SPRINT** family, we have achieved professional quality combined with user upgradability at an affordable price. Our **SPRINT**$^{\textbf{plus}}$ system is the original design, with a 28 pin socket for most devices, and optional 32 and 40 pin sockets for higher density circuits. **SPRINT EXPERT** is our newest system with a 40 pin socket, and exchangeable socket concept for higher pincount and SMD devices. Both **SPRINT** systems support PLCC, SOIC and LCC devices via industry standard adapters.

SPRINT is a convenient programming tool for all kinds of programmable devices, including PROMs and EPROMs (Erasable Programmable Read Only Memories), Flash EPROMs, EEPROMs (Electrically Erasable Programmable Read Only Memories), PALs (Programmable Array Logic), EPLDs (Erasable Programmable Logic Devices), Microcontrollers, and more. All popular technologies can be programmed, including bipolar, NMOS, CMOS, ECL, and even GaAs. **SPRINT** uses the popular IBM and IBM-compatible desktop computers as its 'host' system for data and algorithm storage, while actual programming is under control of the hardware pulse generators and D/A converters of the **SPRINT** system. **SPRINT** is very user-friendly, no special knowledge of programmable devices is required. All operations are menu driven and displayed in clear, easy to understand English text. In addition to the programming ability of **SPRINT**, we have added dozens of additional software features, including a Macro assembler for PLDs and EPLDs, an un-assembler, vector test support with vector editor, EPROM editors, multiple disk I/O modes and more.

One key advantage of **SPRINT** is the ability to add new devices and features without modifying the hardware. Software updates are released on floppy disk three times each year. Each system sold can be updated for 6 months after the date of purchase with new devices and features at no cost. After 6 months, the **SPRINT** Software Maintenance Plan (SMP) automatically distributes 3 updates each year to the user.

We are responsive to your needs. Please write to us if you would like to have any devices added to the list of presently supported chips. We add devices as soon as they become available to us, with the priority set by feedback from our many customers.

Because the list of supported devices is growing rapidly, this Manual contains no device list. All selections are menu driven with vendor names and part numbers. You may create your own list from the device menu onto the printer with the '*' key, or to a disk file with the '+' key.

SPRINT SYSTEM COMPONENTS

The **SPRINT** package consists of three basic functional units:

1) An IBM Compatible Plug-in Board

The **SPRINT** plug-in board contains all the hardware logic that is required to interface between the IBM compatible PC/XT/AT computer and the device socket. The **SPRINT** board produces all the necessary voltages that are required to program the supported logic devices.

The **SPRINT** board should be left permanently installed in the user's IBM compatible PC/XT/AT computer, it does not interfere with normal operation. Chapter 5 describes the installation of **SPRINT** in any IBM PC/XT/AT or compatible computer.

2) Floppy Disk Based Software

The standard **SPRINT** package contains two floppy disks with the programs and utilities that enable you to use **SPRINT**. The programs are all menu driven and easy to use. The programs on the floppy disk directly control the hardware logic on the **SPRINT** plug-in board and socket pod to produce the correct programming voltages for the selected device. The list of devices supported by **SPRINT** is continually being updated, a printer list is quickly out of date. A full list of devices supported by your software version may be obtained from within the programming utilities. Please turn to Sections 1 and 2 of this Manual for more details.

3) **An External Device Socket Unit (POD) With a Zero-Insertion Force Dual Width Socket.**

a) **SPRINT EXPERT**

The **SPRINT EXPERT** external hardware consists of two parts. The bottom unit (POD BASE) contains the actual pin driver logic, this keeps the signal length to a minimum and allows 100% SW programmable pins. The socket is located in the upper unit (POD TOP) and is exchangeable for other types of sockets. Examples of TOPs include

a) 40 pin DIP socket,
b) 4 x 32 pin DIP E/E/PROM gang/set module,
c) 20/28/32 pin PLCC/LCC module,
d) 44 pin PLCC/LCC socket,
e) 68 pin PLCC/LCC socket,
f) various PGA sockets

Other modules will also be available. The base is connected to the IBM type plug-in board via the '**SPRINTbus**'. The cable length of 90 cm allows the computer to be on the floor or otherwise located a distance away from the programming socket. All pinouts are selected by relays inside of the POD.

b) **SPRINT^{plus}**

SPRINT^{plus} uses a 28 pin external socket in a small box located outside of the computer. It is connected to the computer by a flat ribbon cable.

Attention! This cable must not be made longer or errors will occur.

The socket is used by **SPRINT** to provide the connection to the programmable device. This box is called the 'POD'. The standard POD supports devices in DIP packages with up to 28 pins. Other PODs support 32 and 40 pin EPROMs. In addition, various socket adaptors are available, supporting a variety of special devices and packages that do not fit into the 28 pin DIP socket. PODs for 32 and 40 pin devices as well as adapters for Lcc and PLcc circuits are also available.

NOTE: Before reading or programming any device with SPRINT^{plus}, be sure to set the 'Filter' switches on the socket pod to match the diagram shown on the screen. If SPRINT^{plus} can detect an incorrect filter setting, prompt the user to correct the setting. In any case, failure to set the switches correctly may result in damaged devices.

STARTING SPRINT

The user should follow the instructions detailed in the Installation Section of this Manual in order to ensure correct installation (and eventual calibration) of the **SPRINT** Programming Module.

With the **SPRINT** board installed according to the installation instructions, power-on your computer. If using a hard disk, select the **SPRINT** directory. If using a floppy based system, insert your copy of the **SPRINT PROM** disk into the floppy disk drive. Enter the command:

A>**SPRINT**<cr>

The **SPRINT** master menu as shown below will appear.

```
1  =  Logic devices (PAL, EPLD, etc)
2  =  Memory devices (PROM, EPROM, etc)
3  =  PLD Assembler
4  =  PLD Un-Assembler
5  =  TEST Utility
Q  =  Exit

      Enter Selection
```

You may now select the program you wish to run. Enter a '1' for PALs, and EPLDs. Enter a '2' for EEPROMs, EPROMs, PROMs and microcontrollers. You may also start the programs without the master menu by typing 'PAL', 'PROM', 'PLDASM', 'UNASM', or 'TEST' directly at the MSDOS prompt, e.g.:

A>PROM<cr>

This will immediately start the EPROM programming utility without the need for entering the main menu.

Sections 1 through 4 of this Manual describe the use of these specific programming utilities in detail. All the utilities are menu driven. The user is prompted with clear instructions for action. All keys typed by the user may be in upper or lower case letters.

The Manual is divided into sections on the utilities supplied with **SPRINT**.

- For EPROMs, EEPROMs, PROMs, microcontrollers, and other memory devices, the 'PROM' program should be used as described in Chapter 1.
- For programmable logic devices: 'PLDs', 'EPLDs', and 'PALs', the 'PAL' program should be used as described in Chapter 2.
- All special adapters, including instructions for **SPRINT EXPERT** with the TOP432 unit, and **SPRINT^{plus}** with 32 and 40 pin EPROMs (MEGA32 and MEGA40 adapters) are described in Chapter 3.
- The PLDASM utility is provided as a tool for creating JEDEC files from Boolean equations for the 'PAL' utility; details on the operation of the **SPRINT** PLDASM Macro Assembler may be found in Chapter 4.
- The PLD Un-Assembler is described in Chapter 5.
- For periodic checkout and calibration of the hardware components of **SPRINT**, the program 'TEST' is provided (see Installation Section, Appendix B of this Manual for details).

DEFAULT PATH AND FILE EXTENSIONS

The default path for all of **SPRINT** software is the path from which the program was called. This path will be used to display the directory when inputting data. The default extension (file type) depends on the program called; the defaults are:

Program	Default Extension
PLDASM	.PLD
PROM	.COM
PAL	.JED
UNASM	.JED

The default file extension and the default path may be changed by entering the desired default when starting **SPRINT**. The software will extract the path and file type from the sample file description on the command line. For example :

PROM \hexdata*.hex

will change the default to all files with the file extension '.hex' in the subdirectory 'hexdata'. In this manner, another program (like ISDATA LOG/iC) can start one of the **SPRINT** programs with any directory or file type being used initially during the Input data command.

If **SPRINT** has been executed previously, a file with the extension '$$$$.CNF' will have been created or updated in the current directory. This file contains the drive, directory and file extension parameters, the file name, as well as other user options, that were set up by the user in the previous session. This simplifies the use of **SPRINT** in development environments by maintaining an automatic setup state each time the software is started. For best use of this feature, place **SPRINT** files in a directory '/SPRINT' and add '/SPRINT' to your PATH command.

The .CNF files that are created by the different utilities are:

JED$$$$.CNF	For the PLD programming utility
COM$$$$.CNF	For the PROM programming utilities
PLD$$$$.CNF	For the PLD assembler

Please refer to the Input Command Function in this Manual for more details.

SCROLL CONTROL - MORE ?

All commands support controlled scrolling of the screen. When **SPRINT** has filled a screen with data, the question 'More?' is displayed. Answering '.' or 'n' will terminate the command, any other key will allow an additional 20 lines of text to be displayed.

BATCH MODE

Operating **SPRINT** from a batch file allows many special, customer defined applications. For example, a serial number could be entered into the memory buffer to allow EPROMs to be serially numbered. A user could embed **SPRINT** into a completely different application using the batch mode. In effect, all keystrokes from the keyboard can be replaced by a file on the disk. For our examples we will call this file 'FILEB'. With the batch mode a device can be selected, a file read-in, and a device programmed - all automatically. All **SPRINT** programming modules support batch mode. For example, with PROM, batch mode is selected by starting PROM as shown below:

> PROM -bFILEB

'FILEB' ends either when no more data is in the file, or when a 'Q' character is read while in the main menu. It is also possible to start the **SPRINT** drivers with a default batch file. If a file JED$$$$.CNB or COM$$$$.CNB is found on the disk when starting PAL or PROM, then this file will be read as a batch file. The user can use any editor to create this file. Note that the ^J key (0x0A) is the same as the cursor key. Function keys F1 to F9 are represented as ~1 to ~9.

EXAMPLE BATCH FILE : blank check Xilinx 1736:

~8abq

If any error occurs during batch mode, a 'N' is given as the response any internal error control questions, for example 'skip blank check [N]' will have a N response in batch mode. When the **SPRINT** software exits to the calling program, there will be an return code with the following values:

0	no error, normal termination
1	general error (not blank, programming fail, etc)
2	batch file error (not found etc)
3	Device reversed or pin continuity error
0x20-F	(TOP432 only) indicates which EPROM socket failed

FILE INPUT MODE

All SPRINT programs use the same file input format. When a file is to be read, the system will display on the screen, the files in the default directory with the default extension (see defaults above). The highlight bar may be positioned to any entry on the screen by using the cursor keys. An entry is selected by typing <return>.

The input screen shows the current drive, current directory, file type option, and those files that match the type selected. Actual files are displayed with an extension (in yellow color on a color monitor). Selecting a file will take the user to the next menu. Other entries on the screen with non-matching extensions are directories (purple on a color monitor). The entry '..' refers to the root directory; selecting that entry will change the current directory to the root directory. Choosing any other directory entry will select the chosen directory for display.

It is also possible to directly type in any file name instead of using the cursor keys (useful in batch mode). The file extension must be entered, followed by a return. For example, a 2764 binary file on the root directory of drive b could be:

B:\T2764.COM

NOTE:	**When the default drive, directory or file extension is modified in the input command, the new default values are stored in a configuration file ($$$$.CNF). This configuration file is read (if it exists on the disk in the current default directory) to determine the default parameters that were used in the previous session of the PROM utility, the Input command will use these same parameters as default in the current session by referencing the configuration file.**

DEVICE REVERSAL AND CONTINUITY TESTING

SPRINT EXPERT includes the additional feature of device socket testing. Before any operation that applies voltage to a device is started, the connection of the device to the socket is first tested. A backwards or offset device will cause an error message, as will a device with a bent pin. A device with more pins than expected will also cause an error. The operator can correct the error and try again. Note that certain devices do not permit reversal and pin testing.

Should **SPRINT EXPERT** report a pin continuity error, the user will have the option to abort the operation with <ESC> or to retry the operation with 'Y', or to ignore the message and proceed with '#'. We strongly recommend not to continue if this error message occurs.

I/O ADDRESSING CONFLICTS

If for any reason **SPRINT** does not operate correctly in the system, please check first if another board in the system is using the same address as the **SPRINT** card. See the installation section for information on the default port addresses, and how to change them.

1. SPRINT PROM FUNCTIONS

STARTING PROM

The **SPRINT** PROM software controls the **SPRINT** hardware in order to program data into PROM based devices. Start PROM either from the main **SPRINT** menu, or directly from DOS. A menu of PROM manufacturers and vendors will appear on the screen. Next to each vendor in the menu is a single letter or function key vendor code as shown below:

```
SPRINT PROM device type selection

    A = Auto EPROM ID
    B = AMD
    C = AMD-MMI
    D = ATMEL
....
    F7 = Xilinx

Enter Vendor Code
```

The user is then prompted for the key code of vendor of the device that is to be programmed. At this point, the user should type the single key corresponding to the vendor of the device to be programmed. For example, entering a 'D' following this prompt would select Atmel, 'F7' would select Xilinx, etc. The <Esc> key returns operation to the main menu.

AUTOMATIC DEVICE SELECTION

In order to auto-configure the PROM programming utility to match the EPROM currently inserted in the socket, the user may optionally type an 'A'. This will cause the PROM utility to attempt to read the device in the socket using 'silicon signature', determine its type and to configure itself for that device. If any error occurs in attempting to read the device in the socket, due to no device in the socket or a device which does not support silicon signature, no auto-configuring will take place. Note - silicon signature must only be used with EPROMs, and: some EPROM vendors have used the same ID codes. Always confirm the Auto ID selection.

SPRINT EXPERT automatically determines the number of pins in the device and applies the correct Vcc, Gnd and selection voltages.

SPRINT^{plus}, 28 pin POD, should have switches 1 and 2 off during selection. Mega32 switch 1 should be on for 28 pin devices, and off for 32 pin devices. Check the filter switch settings after selection is complete.

BUILT-IN 'WALL CHART'

The user may optionally make a printout of all the PROM and EPROM devices supported by the version of the PROM utility that is being used. This is done by typing a '*' instead of the vendor code. If the printout should go to the disk instead of the printer, the '+' key can be pressed.

During printing of a device list, the option exists to use the 132 column mode of the printer. This function works with IBM compatible (or EPSON type) printers only. The 132 column mode can optionally be turned off at the end of the print cycle as well. The printout will show the complete list of devices supported by **SPRINT**. Some devices may be marked 'requires SMP'. You can access these devices by purchasing an update from your **SPRINT** distributor.

FAMILY AND DEVICE SELECTION

After the selection of the vendor is complete, a second menu will appear on the screen. This menu lists all the devices from the selected vendor that are currently supported by the **SPRINT** PROM programming utility. This menu also has a single key code for each device supported. The device to be programmed should be selected in the same way as the vendor was selected in the previous menu. For example, if Cypress was the selected Vendor and the device to be programmed was a CY7C291, the user would have typed 'B' for Cypress in the first menu and 'G' for CY7C291 in response to the second menu. Following is an example of a device table as it might appear on the screen

```
PROM device table for vendor = Cypress
     A = CY7C225              B = CY7C235
     C = CY7C245              D = CY7C251
     ...
     O = CY7C291A             P = CY7C293A

     Enter Selection
```

Non-supported device selections are ignored. The working size of the memory buffer is automatically set to the device size selected. The first time any device is selected, the memory buffer is initialized to the value set by the 'default buffer contents flag' (see Options). The default buffer size is 32 Kbyte; larger memory buffers are allocated to devices that require more than 32 Kbyte to store the entire device contents. Attention: if your PC does not have enough memory installed, an error message will be displayed. A checksum of the active area in the memory buffer is calculated and displayed on the main menu screen. For large devices, like the 27020, checksum calculation may take a few seconds. If the device size is larger than 2 megabits, **SPRINT** will use the disk as storage. **SPRINT** does not require any EMS or EXPANDED memory.

Some registered PROM devices have, in addition to the data array, additional reset and enable functions. **SPRINT**^{plus} automatically adjusts the total memory size to include these extra functions. Also, for those microcontrollers that support a security fuse, an extra byte is assigned to control this function. These extra features, such as security fuses, initialize bytes, enable options etc, are always set to zero when a device is selected.

The <Esc> key will cause PROM to display the vendor selection menu again.

MAIN MENU

Following the device selection, the main PROM programming utility menu is displayed on the screen. This menu is composed of a list of commands. Command selection is performed with a single key stroke (the letter corresponding to the command in the menu). Commands may be selected with either upper case or lower case letters: **SPRINT** will understand both. The function of each command is described in the following paragraphs of this chapter. In the top right-hand corner of the screen the vendor and device that have been chosen are displayed. If this is does not correspond to the vendor or the device that is to be programmed, then the user may change the device type and the vendor by executing the command 'C' (Change device type), see below.

COMMAND LINE OPTIONS

In addition to the directory and file type option described in the introduction (Default Path and File Extensions), PROM has additional command line options :

> PROM [-S] [-V] [-Z] [-Bfilename] [\Path\Filetype]

a) -V Enables verification of programmed devices at Vcc $\pm$ 10%. The default is verification at 5.0 V.

b) -Z Forces the buffer memory to zero at initialization and before reading files. The default forces the buffer memory to all ones.

c) -B Causes the contents of file 'filename' to replace all keyboard entries until end of file, quit or an error is reached.

d) -G (EXPERT) Multi base systems (Dual, Quad etc) will be activated in GANG mode - all base units must have the same TOP, all devices in all TOPs must be the same type. Programming will program all in parallel.

e) -L (DUAL) Force selection of the left hand POD

f) -Sn (QUAD, OCTAL) Force selection of base position #n, 1 is right-most position, then 2, 3, 4 etc.

PROM PROGRAMMING COMMANDS

A SET ADDRESS RANGE

Sometimes it is necessary to program a set of bytes in a PROM without programming the entire device. The set address range command 'A' allows the user to work on and affect any portion of the entire address range of the device without influencing other areas. The user will be prompted for the start and end addresses of the range to be programmed. The PROM programming utility will check on the values entered to ensure that values entered do not correspond to an address range larger than the total address range of the device.

During blank check, program and verify, only the range selected will be addressed. The added versatility of being able to specify the start address of the device in memory to be anywhere within the full address space of the complete memory buffer allows smaller PROMs to be combined together into one larger PROM. See the 'Read' command for more details on the combine function. If a range other than that which is normal for the device is set, a box will appear on the screen to display the currently active range.

NOTE:	**The checksum display on the screen refers to the entire device, not the range selected.**

B BLANK CHECK

This command verifies that the device is empty. The device in the socket is read and the contents are compared to the expected 'erased' condition. The device type selected must match the physical device inserted in the socket. Some devices require special blank check sequences, **SPRINT** automatically uses these sequences if required.

If the device is not blank, the message 'PROM not blank' will be displayed. Verufy or EDIT can be used to examine the contents of the PROM.

EEPROM and Flash EPROMs require no blank check before programming, they will be automatically erased as part of the programming cycle. The 'B' command for these devices is not significant.

C CHANGE DEVICE TYPE

The active device type can be changed using this command. The user is prompted by a menu with the list of currently available vendors and devices as described in the introduction to this chapter. After selection, **SPRINT** returns to the main menu.

When the device type is changed, the active memory buffer area is automatically changed to accommodate larger or smaller devices as necessary; however, data previously stored in the memory buffer remain unchanged until either a device is read, or data is read from the disk.

Pressing the Escape key 'ESC', at any time during the change device phase will return the user to the previous menu without affecting the currently selected device.

D DOS COMMAND

This allows MSDOS commands to be executed without leaving the **SPRINT** environment. Note that the MSDOS 'COMMAND.COM' file must be either in the **SPRINT** directory, or locatable via the DOS PATH command for this to work. If large EPROMs are select, there may not be enough room to load the MSDOS command file. Once loaded, any normal DOS command can be entered, in the same manner as in DOS itself. For example:

```
DIR *.COM
```

will list all the files with the extension or file type ".COM" (MSDOS binary files). **SPRINT** will remain in the DOS command mode until 'EXIT' is typed on the keyboard. On completion of the DOS command, the user is returned to the main PROM utility menu.

E EDIT MEMORY CONTENTS

Selection of this command allows the user to examine and modify the contents of the memory buffer. This Screen oriented editor displays 256 bytes (100 HEX) from the buffer at any one time. Data may be input in HEX or ASCII format.

After typing 'E' for Edit, the user will be prompted to enter the address to be edited. The user should enter the HEX address, followed by a carriage return. The screen will then display a 256 byte page of data from the memory buffer including the selected address, the cursor will be positioned to the selected location. Cursor control keys may be used to select any address location in the memory.

The Help screen lists extra data control functions in addition to the cursor movement keys. Help is displayed with the F1 key.

In order to enter TEXT mode, the user may type the ' key. This allows the entry of ASCII text data directly. ESC returns from TEXT mode to HEX mode.

Data may be entered while in HEX mode by simply typing in the hex data. The ASCII equivalent will be displayed at the end of each line.

The F2 key is used to directly jump to any location in memory (GOTO). After pressing F2, the user will be prompted to enter the destination address.

The F3 key is used to copy blocks of memory. After pressing F3, the user will be prompted for the start address, number of bytes (in Hex) and destination address.

The F4 key will swap all even and odd bytes in memory.

The Insert key will enable the insert mode. Normally data entered overwrites data in memory. With insert mode enabled, all locations after the location pointed to will be copied up one byte.

The '.' key or the 'ESC' key will exit the editor, with a new checksum to be calculated, and a return to the main menu. For very large devices, this can cause some seconds of delay. The correct checksum is displayed in a box to the right of the main menu screen.

F FILL MEMORY

A range of buffer memory can be initialized to any value with this command. After 'F' is entered, the user will be prompted for the hex start address, end address, and one byte of data. The addresses and data are separated by commas. After carriage return, the range of memory will be initialized to the value entered. Fill will always change at least one byte on execution.

H HANDLER INTERFACE

The optional handler interface is controlled by this command. The number of devices to be programmed with the pattern in memory is entered after the prompt. Then the 'P' key is entered. **SPRINT** will cycle through the number of devices selected, counting only the good units. The **SPRINT** handler interface kit is required. **SPRINT EXPERT** is strongly recommended due to the device reversal and pin continuity features.

I INPUT DISK DATA INTO MEMORY

Loads the data from the file selected into the memory buffer. After 'I' for Input is entered, the system will display the files in the default directory with the default extension. The default extension is '*.COM' unless otherwise specified in the original command line or the configuration file (COM$$$$.CNF, see the section Default Path and File Extensions).

See the Introduction Section for information on the use of the **SPRINT** file input mode.

After a filename is selected, the user will be prompted to enter various options about the file. The keyboard cursor up and down keys are used to point to the option required, and the value can be typed-in. The file type, Binary or Hex may be selected (default is Binary). The byte/word extraction can be specified (default is All). The starting address of the data in the file on disk can be entered, and the starting address of the data in the buffer memory can be entered (default is the 'bottom' address defined by the 'A' address range command. A return without changing any parameters starts the load process. The 'Auto' option default for the start address allows SPRINT to find the most likely starting address for Intel HEX files with physical starting addresses in segments other than segment 0. Inserting a 0 in this field will force the physical address 0.

There are 9 different input options for the data:

A	All for 8 bit data
E	Even for even bytes of 16 bit data
O	Odd for odd bytes of 16 bit data
0	Byte 0 of 32 bit data
1	Byte 1 of 32 bit data
2	Byte 2 of 32 bit data
3	Byte 3 of 32 bit data
U	The last 2 bytes of 32 bit data
L	The first 2 bytes of 32 bit data

The user can select any one of these options.

Before any data is read-in, if the start address is equal to 0000, the memory buffer will be reset to either 00 or FF, depending on the buffer default option. See the 'O' command for more detail. Any extra bytes (security, initialize, reset control etc) will be set to 00. Note that these extra bytes are located after the actual memory data.

If the start address is not equal to 0000, the existing data in the memory buffer is not changed. This is done to allow a files to be merged and then be programmed into devices.

Data will then be read from the specified file using the options selected, until either the 'TOP' of the memory buffer or the end of the file is reached. The number of bytes read from the file will be displayed on the screen. After reading, a new checksum value will be calculated. For very large devices, this can cause some seconds of delay. The correct checksum is displayed on the main menu screen.

Nine different file types are supported,

a) Binary Format

If this format is selected, the data in the file will be read until either the end of file or the top of the memory buffer is reached. A non-zero start address offset in the data file on the disk may be specified, the default is zero. Finally a destination address offset for the data in the memory buffer can be selected. The default for the destination address offset is the beginning of the programming range (see Set Range command 'A').

b) HEX Space

This is a string of bytes in hex format, separated by a space character. Returns and line feeds are ignored. An example of the format is shown below, it is also on the distribution floppy disk (file "DEMO.HEX"). HEX space files are terminated by an ASCII ETX (HEX 04) character.

STX 12 23 34 45 56 67 78 89 90 01 12 23 34 45 56 67 78 89 12 23 34 45 56 67 78 89 90 ETX

c) Intel HEX, standard and extended
Tektronix HEX, standard and extended
Motorola S Record, 16, 24 and 32 address modes.

These formats are supported according to each manufacturer's definition. A start address of the data in the file may be selected. With Intel and Tektronix HEX formats, both 16 and 20 bit addresses are supported, with Motorola S Records, 16, 24 and 32 bit addresses are supported.

SPRINT automatically determines if the HEX file is SPACE, Intel, Tek or Motorola.

L LIST MEMORY CONTENTS

This command displays in tabular form the contents of the memory buffer. When 'L' for List is entered, the user is prompted for the start and end addresses. A comma separates these addresses. If the user wishes the display to go to the printer, a ', * ' is typed after the end address. Memory is displayed in HEX, 16 (HEX 10) bytes per line, from the start of the address range to the end of the address range. A pause is inserted after each full screen to allow the user to examine the contents of the memory buffer. The next screen of data can be displayed by entering a 'Y' or return at the end of each page. The continuation of memory display may be terminated by typing 'N' or '.' at the end of the currently display page. The ASCII equivalent of the HEX data is also displayed at the end of each line, for data values between 32 and 127 decimal.

O DISPLAY PROM OPTIONS

This command will display on the screen, the programming parameters **SPRINT^{plus}** will use based on the device selected. It can be used to compare the voltages selected by **SPRINT^{plus}** with the voltage recommended in the data sheet of the device. Any discrepancies found in the programming voltage selection should be notified to your **SPRINT^{plus}** supplier immediately. All other parameters displayed can be useful information about the device selected, and the operation mode of **SPRINT**. The user can change any parameter shown (at his own risk!!). These parameters will not be saved once the program is exited (except screen display mode). The status displayed includes:

Installed pod type	
Hardware model	
Device size in decimal bytes	
Type of device	
Vpp voltage, modifiable in 100 mv steps	a
Vcc voltage during programming, 50 mv steps	a
Vendor ID (if defined) in HEX	
Vendor device ID (if defined)	
Width of first programming pulse	a
Maximum number of programming attempts	a
Width of last programming pulse	a
Type of programming method	
Number of special feature bytes	
Vcc voltage during verify	t
Screen Display mode, use slow if your screen has 'snow'	t
Default buffer memory contents, 00 or FF hex	t
Sound BEEP at the end of the programming operation	t

To modify a parameter (a), enter the new value while the cursor is on the line. To toggle a flag (t), press the 'SPACE' key. The modified parameters are only used during this session, and are not stored to the disk.

P PROGRAM DEVICE

When 'P' for Program is entered, the device in the socket can be programmed using the data stored in the memory buffer. The device type selected by the user must match the physical device in the socket. The user is prompted to confirm the programming of the device by typing a 'Y', any other key will abort the command. This has been implemented so that accidental programming of devices may be avoided. **SPRINT**$^{\textbf{plus}}$ users should verify the Filter switch settings.

After confirmation that programming of the device is to be performed, non-erasable devices will be checked for erasure (Blank check). If the device is not empty, **SPRINT** will test if the pattern in memory can be overprogrammed on top of the existing pattern in the device. If so, the user will be prompted to 'overprogram'. Entering 'Y' will allow **SPRINT** to program on top of the existing pattern. Erasable devices will be erased automatically before programming.

Then the device will be programmed using the algorithm defined by the manufacturer. Intelligent, Flashrite, Rapid, Quickpulse, Quickpro, Page Mode and all other fast programming methods are used to keep the programming time to a minimum. The programming algorithm used is automatically determined by the vendor/device type chosen. For devices with long programming times, the number of 512 byte blocks remaining to be programmed is displayed to indicate that the programming operation is processing. This is useful in large devices not using QuickPulse type algorithms. This 'ACTIVITY' display is not shown when QuickPulse devices are programmed. When the 'ACTIVITY' display is shown, programming can be interrupted with the <ESC> key.

Extra PROM and microcontroller features like Reset, Synchronous Enable and Security Locations are automatically programmed in devices which have them; these features are controlled by extra data bytes. See the PROM notes section for specific details and examples. The extra bytes in the memory buffer may be set using the buffer edit utility (see above), and can be stored to the disk with the 'W' command.

After programming, the contents of the device will be verified against the contents of the memory buffer to confirm the successful programming of the device. Following a correct verify operation, when everything has been successfully completed, the message 'PASS' appears on the screen.

If there are any errors, they will be shown on the screen, and the blinking message 'FAIL' will appear. Otherwise the message 'PASS' will be displayed. The number of devices successfully programmed with the same contents will be displayed in a box to the right of the PASS/FAIL box.

Q QUIT

Returns the user to the **SPRINT** menu or the operating system if the menu program **SPRINT** was not used.

READ DEVICE TO MEMORY

This command reads the contents of the device in the socket pod into the memory buffer. Note that after data has been read into memory, the same size or smaller device type may be selected without altering any data stored.

When registered PROMs are selected, the Reset Byte and Programmable Synchronous Enables (if any) are automatically read into the buffer memory for correct duplication or storage.

After reading in the data, a new checksum value will be calculated. For very large devices, this can cause some seconds of delay. The correct checksum is displayed on the main menu screen.

With the 'A' command (= set address range), an address range larger than the physical device may be defined. This allows the user to read in several small EPROMs, at successively higher contiguous memory addresses in the memory buffer. This feature enables the user to combine the data contained in a number of small EPROMs and to program this data into one larger EPROM. The editing buffer size must be initialized first by selecting the larger device, then the smaller one. The EPROM in the socket can be read into any portion of this editing buffer. When a larger device is selected with the Change command, this editing buffer is not cleared.

Example: To combine the contents of four 2764's and to program this data into a single 27256 the following procedure may be performed.

a) Select the vendor and device for the 2764 devices that you wish to read.
b) Select the address range 0 - 1fff (A command)
c) Read in the lowest addressed 2764 (R command)
d) Select the address range 2000 - 3fff (A command)
e) Read in the 2764 with the next address (R command)
f) Select the address range 4000 - 5fff (A command)
g) Read in the 2764 with the next address (R command)
h) Select the address range 6000 - 7fff (A command)
i) Read in the highest addressed 2764 (R command)
j) Change the device type to 27256 (C command)
k) Insert the 27256 device into the socket and program it (P command)

V VERIFY DEVICE

Compares the contents of the device in the socket to the data in memory buffer. Any differences are displayed in clear text, with memory data, device data and address. All features of the device are verified, including extra reset and enable bits and architecture bits.

Verify may be done at nominal Vcc, or Vcc $\pm$ 10%. See command line options or the 'O' command for this feature.

W WRITE BUFFER DATA TO DISK

Writes the data in the memory buffer onto the disk using the filename supplied. After entering 'W', the user will be prompted to enter a MSDOS style path. A drive label or path may optionally be entered. The default path is accepted if <Return> is entered without any other characters. The filename may then be entered.

The user may select either Binary [B] or Intel Hex [H] format to write the data.

When writing the buffer to disk, the number of bytes written corresponds to the size of the currently selected address range.

In case a file with the same name exists, the user will be prompted to confirm erasure of the previous file.

PROM ERROR MESSAGES

The following is a list of common error messages for the PROM utility.

1) **Disk Full, no data written.**

   The disk drive used for the write operation was too full to contain the entire file. The partially written file has been erased.

2) **PROM not blank**

   The PROM contains data. The data in the **SPRINT** memory buffer cannot be over-programmed into the device.

3) **O.K. to program on top of existing pattern?**

   PROM was not blank, the new pattern can be over-programmed onto the existing pattern. Enter 'Y' to commence programming.

4) **Skip blank check?**

   Device is not blank. Enter 'Y' to program anyway.

5) **Incompatible TOP POD**

   The currently installed TOP cannot support the operation attempted.

6) **Manufacturer xx Device yy not recognized**

   Auto ID could not identify the device in the socket. Either the device does not support silicon signature, or the switch settings (**SPRINT**$^{\textbf{plus}}$ only) were not correct for the size of the device.

PROM NOTES

1. Registered PROMs which have a programmable initialization byte, have this byte stored after the last data byte. The location in memory of this byte is demonstrated by these examples:

Device size	Type	initialize byte location (HEX)
1024	7c235, 27S35	400
2048	7c245, 27S45	800
8192	7c268, 7C269	2000

SPRINT automatically programs this data into the PROM during the programming cycle. The byte can be read in from a device or from a disk data file. When reading a device, **SPRINT** automatically determines the reset byte value.

2. Proms which have additional programmable features have the data stored in a control byte after the reset byte value. For example :

Device size	Type	Control byte location (HEX)
2048	7c245, 27S45	801
8192	7c268, 7c269	2001

SPRINT automatically programs this data into the PROM during the programming cycle. The byte can be read in from a device or from a disk data file. When reading a device, **SPRINT** automatically determines the control byte value.

3. Microcontroller programming is identical to PROM programmming, except that an additional byte contains a flag for the security fuse. This byte is located after the last data byte. If a device contains multiple fuses, the first fuse will be bit 0, the second bit 1, etc. Many microcontrollers require adapters (see Chapter 3 for adapter information).

4. (**SPRINT**$^{\textbf{plus}}$ only) When programming bipolar 16 and 18 pin PROMS, Vcc is supplied to these devices from the socket pin marked '<-20'. This means that either a jumper is required to connect this '20' pin to the device Vcc pin, or an intermediate socket can be inserted to make this connection.

5. The Cypress 7c245, 7c268 and 7c269 have programmable Enable and programmable SYNC bits. The control byte location contains the instructions for **SPRINT** for programming these extra features. The data stored in this location has the following meaning, a value of:

0 = async enable	7c245, 268, 269
1 = sync enable	7c245, 268, 269
2 = async initialize	7c269

6. EEPROM. The Xicor 28C256 has 1 extra byte as security fuse. If this byte equals 1, the write protect feature of this device will be set after programming.

M MANUAL PRODUCTION PROGRAMMING

IN **SPRINTplus**, and **SPRINT EXPERT** installations with a single pod installed, this command sets up a counter for programming a fixed number of devices. After entering the number of devices to program, the operator need only insert the device, and type any key to program. Programming stops after the unit count is reached.

In **SPRINT EXPERT DUAL** systems, the counter is first initialized, afterwards the active socket switches from right to left (SWAP mode). The currently active socket is shown by the LED pulsing. In this mode, one device can be programmed while the other socket is being loaded/unloaded. The operator need only type <space> to continue programming until the unit count is reached. This mode can double programming throughput.

The <ESC> key will turn off manual production mode. Failing devices are not counted.

T. GANG / SET PROGRAMMING OPTIONS

This command is enabled if either a MultiSyte system is detected at power-on, or if a TOP432 is detected.

The default condition when starting SPRINT EXPERT MultiSyte systems is for the right-most socket to be enabled. This socket can be used in the same manual as a single SPRINT socket system.

The T command allows other options to be selected. The T command sub-menu contains the following options:

N	standard single mode operation	(default)
G	Gang mode operation	
E	Eight bit set mode operation	(TOP432)
S	Sixteen bit set mode operation	(TOP432)
T	Thirty two bit set mode operation	(TOP432)
L	Switch to the right (left) socket	(DUAL)
Ln	Select socket number 'n'	(QUAD +)

N: Normal mode sets the system to the right most unit.

G: GANG mode, all sockets receive the same information, and all must be the same type. Set programming is only possible with the TOP432. In TOP432 mode - set programming can be done up to 32 bits (one TOP432). Multiple TOP432s can be used in GANG/SET mode, where multiple sets of EPROMs are programmed - each TOP432 unit receives the same information.

E: The 8 bit set mode places sequential data into each EPROM, 'stacking' the EPROMs together to hold the entire data loaded into memory.

S: The 16 bit set mode puts all odd data into the sockets 1 and 3, and even data into sockets 0 and 2. Two sets of 16 bit wide data can be programmed at a time.

T: The 32 bit set mode puts the 0th, 4th, 8th byte (etc) into socket 0, the 1st, 5th, 9th byte (etc) into socket 1, and so on.

L: As of the units can be directly and uniquely selected with this command. The change to a different TOP will cause PROM to enter the device menu, which will show the devices possible with that TOP. This mode of operation allows a DUAL or QUAD system to have different types of TOPs - one in each position.

NOTE: The SET or GANG mode must be selected before data is read into memory, because set modes and 4 megabit EPROMs requires 2 Megabytes of data - all of which is organized in the overlay data files for high-speed programming and effecient memory use.

ERRORS: Any errors detected during verify or programming will cause the LED of the socket with the error condition to turn RED. This RED LED will remain on after the programming cycle is complete. Other devices will continue to program.

2. SPRINT PLD FUNCTIONS

STARTING PLD

After entering the PLD programming utility, a menu of PLD manufacturers and vendors will appear on the screen. Next to each vendor in the menu is a single letter vendor code as shown below.

SPRINT PLD device type selection

A = Altera	B = Atmel
C = AMD	D = AMD/MMI
E = Cypress	F=Cypress/Aspen
G = Excel	H = Gould/AMI
I = ICT	J = Intel
T = Texas Inst.	U = VLSI Tech.

Enter Vendor Code

The user is then prompted for the vendor of the device that is to be programmed. The user should then type the single letter vendor code corresponding to the vendor of the device to be programmed. For example, entering a 'E' following this prompt would select Cypress , a 'D' would select MMI etc.

The user may also optionally make a print-out of all the supported devices for the version of the PAL utility that is being used. This is done by typing a '*' instead of the vendor code. The list can be directed to the disk by pressing the '+' key. Some devices may have been added since your warranty or SMP period expired. These are listed in the printouts with a '+', and on the screen as SMP. Please contact your **SPRINT** distributor for an update.

After the selection of the vendor is complete, a second menu will appear on the screen. This menu lists all the devices from the selected vendor that are currently supported by the **SPRINT** PLD programming utility. This menu also has a single key code for each device supported. The device to be programmed should be selected in the same way as the vendor was selected in the previous menu. For example, if Cypress was the selected Vendor and the device to be programmed was a PALC22V10, the user would have typed 'C' for Cypress in the first menu and 'E' for PALC22V10 in response to the second menu. Following is an example of a reduced device table as it would appear on the screen:

```
PLD device table for vendor = Cypress

    A = PALC16L8          B = PALC16R4
    C = PALC16R6          D = PALC16R8
    E = PALC22V10         F = PALC20G10

    Enter Selection
```

For some vendors (e.g. MMI) there are too many devices to display on a single menu. For these vendors a separate menu will appear which requests the user to choose a category of devices into which the device to be programmed falls (e.g. A-PALs, B-PALs etc.)

Incorrect selections are ignored. The size of the memory buffer is automatically set to the device size selected. Changing between compatible devices will not erases the data stored in memory. If a device type is changed to another device having an internal organization different from the initial device, the memory is reset to the blank condition in order to avoid operator errors.

PLDs have different internal sizes. Only valid fuse locations are enabled when a PLD type is selected. The internal size is compatible with the JEDEC data files generated by all PLD development software.

The user may at any stage during the device and vendor selection phase exit the PLD programming utility by hitting the Escape (ESC) key. This will return operation to the previous menu.

MAIN MENU

Following the device selection, the main PLD programming utility menu is displayed on the screen. This menu is composed of a list of commands. Command selection is performed with a single key stroke (the single code letter corresponding to the command in the menu). Commands may be selected with either upper case or lower case letters, **SPRINT** will understand both. The function of each command is described in the following paragraphs of this Chapter. In the top right-hand corner of the screen the vendor and device that has been chosen are displayed. If this is does not correspond to the vendor or the device that is to be programmed, then the user may change the device type and the vendor by executing command 'C' (Change device type), see below.

MORE ? ERROR DISPLAY

If an error occurs during Blank Check, Verify, or Test, the user will be prompted with the message 'ERROR - display detailed report? [N]'. Entering any key except 'Y' will cause the main menu to be displayed with the 'FAIL' box blinking in inverse video. If the error display continues over 20 lines, the message 'MORE ? [N]' will be displayed, enter 'y' to continue, any other key to exit.

For blank check and verify errors, the detailed report will be an X-plot of the device with 'X' and '-' indicating the locations where incorrect data was read. Only erroneous locations will be displayed. The displayed 'X' or '-' is the data **SPRINT** expected to read from the device - the actual device therefore has the opposite data.

For test errors, either during a programming cycle or from the main menu 'T' command, the user will have the error displayed on the screen, and will be prompted to either display more errors, or exit. If SMP is installed, it is possible to enter the Test Vector Editor directly from the error screen. In the SMP test vector editor, the user need only type 'space' to have the failing test vector location graphically displayed on the screen.

PLD PROGRAMMING COMMANDS

A ASSEMBLE

The **SPRINT** PLD programming utility has a PLD macro-assembler included. This assembler is designed to convert PLDASM source files (ASCII text) into JEDEC compatible files that are needed to program the PLD itself. Details on the format of the PLDASM source files and the functions and operation of the assembler can be found in Chapter 4 of this manual.

The Assemble command loads the **SPRINT** PLD macro-assembler. The user is then presented with a list of all the files in the default directory with the default file extension (*.PLD unless otherwise specified). The source file is then selected. The source file may be selected from the list presented on the screen or any other filename may be entered, specifying filename, directory and file extension. The assembler will generate a JEDEC standard compatible file with the same name as the source file but with a .JED file extension in the same directory as the source file.

After assembly is complete, the system will return to the **SPRINT** main menu, the user should check that the desired device type is selected, and read in the file from the disk using the 'I' input command.

B BLANK CHECK

This command verifies that the non-erasable device in the socket is empty. The device in the socket is read and the contents compared to the expected 'erased' condition. The device type selected must match the physical device inserted in the socket.

Electrically erasable devices do not require a blank check operation. These devices are automatically erased as part of the programming cycle.

The user has the option to display an XPLOT of those fuses that are not blank - as described int the Section on MORE ?.

C CHANGE DEVICE TYPE

The currently active device type can be changed using this command. The user is prompted by a menu with the list of currently supported vendors and devices as is described in the introduction to this Chapter. After re-selection, **SPRINT** returns to the main menu.

The selection sequence is as follows. The vendor name is first selected using the single key code for each vendor, as shown on the menu screen.

Then the family code is selected. The family code is typically shown on the physical device before and after the device type. For example, the family type for an AMD AmPAL16L8A is AmPAL--A. For a Texas Instruments TIBPAL16L8-10, the family code is TIBPAL -- -10.

Finally the actual device type (architecture) is selected. A little practice with selecting device types will show that this is the quickest and easiest method for selecting a device. It may be useful to print out a directory of device types with the '*' or '+' options until you are familiar with the approach.

SPRINT does not list package options, since package options do not affect the programming algorithm of the device. When using SMD devices, simply select the same device name as for the DIP device.

To allow copying from one device type to another, memory contents are not erased when changing between devices with the same architecture. The device type selected is shown on the main menu screen. If, when changing device types the internal size of the fuse arrays changes, the memory will be reset to the blank condition (checksum = 0) to avoid incorrect conversions.

SPRINT^{plus} users must check the filter switch settings before reading or programming any device.

NOTE:	**Reading and programming PLDs involves the use of voltages higher than 5V. If the device type displayed on the main menu is not the same as the device in the socket, damage to the device may result.**

CROSS PROGRAMMING

Cross programming is a feature provided to support many 16V8, 20V8 and related devices. The idea of cross programming is that the programmer accepts a first generation device type (for example 12H6) and programs a JEDEC file for that device directly into a replacement device (the 16V8). The JEDEC map of the 12H6 is not compatible to the 16V8, therefore the programmer must do the translation at the moment when the 16V8 is to be programmed. **SPRINT** supports 42 different first generation devices, and presently 2 different replacement devices (16V8 and 20V8). To use cross programming, simply select the replacement device with the 'AS' option. For for example '16V8 as'. Then select the first generation device from the '16V8 as' menu (12H6). You can now read in the 12H6 JEDEC file and program the 16V8. All architecture bits and signature fields will be automatically configured. No translation is required.

In order to support devices that are too complex for direct cross programming, the **SPRINT** PLD programming utility supports a conversion via the 'X' command. The 'X' command is explained below. Most popular PAL and EPLD types can be converted with this tool. (SMP is required).

D DOS COMMAND

This allows MSDOS commands to be executed without leaving the **SPRINT** environment. If the file 'COMMAND.COM' is not located in the current directory, or cannot be found via the PATH, then the function will not work. Any normal DOS command can be entered, in the same manner as in DOS itself. For example:

```
DIR *.JED
```

will list all the files with the extension or file type ".JED" (often used for JEDEC format files). You can return to PAL operation by typing 'EXIT'.

E EDIT MEMORY CONTENTS

SPRINT includes a very powerful editor to help you to understand the internal configuration of a PAL or EPLD. The display includes identification of the pin numbers associated with the various product terms, and displays the internal features with the node numbers corresponding to popular PLD compilers. The editor can be used to debug PAL designs, or to verify the output of assemblers or compilers. Any fuse in the device can be changed. It is recommended that 'snow' be exercised to set the fast mode (see 'snow' description). The faster screen updating is useful both for EDIT and for the test vector editor. We refer to the programmable elements of the device as 'fuses' even if they may be erasable CMOS cells.

> NOTE: **The modification of PLDASM source files and reassembling of that source is a more practical method for making modifications to a PAL's equations and is less likely to result in errors.**

After entering 'E' for Edit, the user sees a display of the XPLOT on the screen. By using the cursor keys, any fuse can be pointed to and changed if desired. For popular PAL and EPLD devices, the pin numbers corresponding to the product terms shown on the screen are displayed. Internal Set and Reset terms are displayed as SP and AR respectively. Device Architecture bits are displayed as 'A'. Other nodes internally are displayed as N1, N2 etc.

EDIT (continued)

The program then waits for user commands as follows:

home	goes to fuse 0
end	goes to first fuse, last product term or architecture row.
page up	24 lines up
page down	24 line down
arrows	moves cursor one location
^left	control left moves to left margin
^right	control right moves to right margin
<cr>	display next line
Backspace	moves cursor one location left
<esc>	exit to main program
X	stores a 0, meaning 'connect' at the current fuse
-	stores a 1, meaning 'disconnect' at the current fuse
1	set entire product term to no connect (on)
0	set entire product term to all connected (off)
*	print the XPLOT on the printer
F1	display help screen
F2	do not display product terms that are off, (all '0')
F3	toggle screen mode - fast/slow
F4	display JEDEC fuse numbers/display product term numbers

After exiting, the new checksum of memory is calculated and displayed on the main menu screen. Any change in the checksum will cause the filename display to be reset.

'Snow' is an affliction of many older CGA and MDA screens. Depending on the type of the display controller, white specks could be seen on the screen whenever the computer wrote data to the screen display. Setting 'snow' to slow mode turns off the white specks, at the expense of speed. Modern controllers do not have this problem. The default condition is fast.

G EXERCISE DEVICE (EDIT TEST VECTORS)

This function is available as an additional SMP option. Contact your **SPRINT** distributor for information about this feature if your present system does not display the 'G' command in the main menu.

Please read the Section on the 'T' function for more information about test vectors and related problems.

This function is invoked in one of two ways. The 'G' command from the main menu will start the exercise directly. This is the method an engineer would use to develop and test a suite of test vectors. The 'G' command offers a graphic display of the device in the socket, and allows interactive development of PAL/EPLD vector test functions. Once a suite of vectors has been generated, they can be stored on the disk with the normal 'W' command - they will be attached to the JEDEC fuse description file. The UNASM program can also be used to create a transportable source file of the test vectors.

The second method is to optionally enter the exercise mode if the device in the socket fails the test vectors loaded into memory as part of the standard JEDEC file. If an error occurs, either as part of the normal programming cycle or from the main menu 'T' command, the user will be prompted with the option to 'enter test/edit mode'. In the production programming environment, the operator would enter 'N', and the menu will report 'FAIL'. In the laboratory, entering 'Y' to enter EXERCISE followed by 'SPACE' to run the vectors, will cause the failing location to be displayed on the screen, the failing bits are shown in red (or inverse video in monochrome systems).

One purpose of this function is to allow a device failing a vector test to be examined. The exerciser will display in red those pins which do not match the expected result. After starting the function, entering 'SPACE' will step through the vectors in memory until an error (or the end) is reached. The 16 vectors prior to the stopping point, the current vector and the next 3 vectors will be displayed on the screen. The engineer can examine the error, change input conditions, or expected output levels, single step through the cycles and determine whether the error is due to the test vector or an actual failing device.

Another purpose is to develop test vectors for a device. This could be done as part of the design cycle. Using the exerciser, a programmed PAL in the socket can be stimulated with all input conditions and the result can be seen directly on the screen. The test vectors created in this manner can be then stored on the disk for use in testing production devices. If the device does not behave as expected, then the source file for the device can be changed until the device functions as expected.

Exercise (continued)..

The exerciser could also be used to apply signals to any device in the socket to observe the results, standard 74 series TTL, HC cmos, PROMs, or any other product can be examined in this manner. To allow any type of devices to be tested, the select menu includes the device type 'Test a chip with xx pins'. Note that Vcc drivers are available on certain pins of the socket, refer to Chapter 3 for the VCC driver list for the POD you are using.

STARTING EXERCISE

When this command is started, an image of the device in the socket is displayed on the screen. All pins are set to the level '?'. The level/function menu is displayed, and, if test vectors are in memory, the first 4 will be will be displayed. To execute the vectors in memory, the 'E', 'G', 'R', or 'SPACE' functions may be entered.

If there are no vectors, or you wish to apply your own signals, then you should set the device levels. Any pin of the device may be set to any level by the level command. Enter the level first (for example 'l') then enter the pin number, then 'RETURN'. This process can be repeated for each pin in the device. More than one pin at a time can be set by typing two numbers with a comma in between. A simple <RETURN> (single cycle) will cause the signals to be applied to the device, and the results will be displayed on the screen.

This result can be stored as a vector for future use by entering either 'U' to update the current vector, or with 'M' to make a new vector. If a vector is made, it will be inserted into the vector sequence after the current vector location, which could be either the middle or the end of the vectors in memory. After updating or making a vector, the exerciser will run from the START vector (default is vector 1) up to the new vector. The reason for assigning a START vector, is that any registered PALs' present status is dependent on all previous states. An alternative to using vector 1 as START, is to set the start vector to one which preloads, resets, or otherwise assigns all registers inside the PAL to a defined state.

The execution of each test vector consists of 3 phases. First all inputs are set to 1 or 0 according to the vector. Then a clock pulse is applied to those pins with C or K levels. Finally, the actual device output pins are read and compared to the vector levels H, L, Z, . Any differences are displayed in red (inverse background on monochrome systems). If an error is detected during execution of the vectors, execution stops.

NOTE: The 'X' level should be don't-care, however many compilers expect 'X' to be logic 0, so **SPRINT** applies a logic 0 to all pins that are not used as outputs by the vectors in memory, the possible output pins are allowed to float to a logic 1 level, allowing the output to pull the pin to 1 or 0 without causing a short circuit.

To save a set of test vectors on the disk, the user can quit the exerciser ('Q' or 'ESC') and store them on the disk with the main menu 'W' command. These vectors will be attached to the device fuse information and stored at the end of the standard JEDEC file.

The UN-assembler command will also save the test vectors by storing them in a source file so they can be assembled into any destination device.

JEDEC LEVELS

Each pin can be set to any of the levels listed below. To set a level on one pin enter the level followed by the pin number. To set a range of two or more pin numbers, enter the level followed by the lower pin number, comma, then the higher pin number. With the range method, pins set as N (not used) or V (Vcc) will be skipped.

LEVEL	TYPE	FUNCTION
C	input	fast risetime clock from 0 to 1 to 0
F		float pin (same as '1')
K	input	fast risetime clock pin from 1 to 0 to 1
H	output	test pin for 1 output level
L	output	test pin for 0 output level
N		not used for I/O will float pin to 1 unless used as power or ground
P*		preload device outputs per device specs
B*		test buried registers
V		Set pin to 5 volt Vcc supply, see VCC section
X		don't care (will float to 1 or 0)
Z	output	test pin for floating condition
?	output	read any output level
1	input	set pin to logic 1 via 4.7K ohms (slow risetime)
0	input	set pin to logic 0, minimum 16 ma sink.

MANUALLY SETTING LEVELS

After entering a LEVEL,PIN combination, the levels are first set in memory and displayed on the screen. They will be applied to the device only after the next single cycle command (simple <RETURN>). In this manner, multiple new levels can be set as inputs before any clocks are applied to generate the next state in registered devices. As a reminder that the inputs on the screen are not yet applied to the device, the color of the vectors change to blue from green, then back to green (or red) after the single cycle. As long as levels are being changed, single step does not advance the vector pointer. If no levels are changed, single step executes the next vector unless none are loaded or the pointer is at the end.

NOTE: The use of preload (B or P) implies a detailed knowledge of the device. This function is not supported by all devices, and it uses voltages above +5 volts according to each manufacturers specifications. The device in the socket must match the device chosen or damage may result. **SPRINT** knows the preload procedure for all supported PALs. If preload is used, then the outputs become inputs (set as 1 or 0) and the data on the output pins is loaded into the /Q (Q bar) of the flip flop on that pin. P and B levels must be on pin one. Be sure that asynchronous resets are not activated by the transition in and out of preload, or the preload cycle will appear to fail.

The sequence of the application of the clock is not defined by JEDEC. To generate portable test vectors, if a device has more than one clock input, then separate vectors must be used to assure the correct clock sequence (if critical). If the C or K level is set on two device pins, either one may be toggled first depending on programming system manufacturer. **SPRINT** toggles pin 1 last.

Note that high speed PALs will create noise in many systems. Thus the C and K levels employ active pull-ups to 5 volts, while the 1 level is a slower resister pull-up. For this reason you may witness mis-counting if 0 to 1 levels are used as a clock instead of C or K. Also, the K is preferred over C since the noise margin between cycles on K is better than C. (3 volts vs .6 volts in TTL compatible devices).

VCC DRIVERS

Not all pins have VCC drivers available. You should refer to the section of Chapter 3 for the POD type you are currently using. While jumpers could be used to provide power to pins that do not have a built-in VCC driver, caution is required to avoid a short between a pin at '0' and the VCC level.

The 'V' level is not a JEDEC level, and is used only to indicate which pin is VCC. The JEDEC standard indicates this pin as 'N' - not used. Based on the device selected, **SPRINT** will automatically provide VCC power and ground to all necessary pins of the device.

FUNCTIONS

These are debug like commands for generating, single stepping, and editing test vectors. Initially the starting address (START) for vector execution is set to vector 1. This START address is always displayed in the screen. The functions are listed here. If a function has optional parameters, the user will be prompted to enter these options after the function code. If no option is entered, then the default is the value '1'. The escape key can be used at any time to abort the command. These function are listed on the following 2 pages.

FUNCTION (# options)	EFFECT
A 1-16	**Assign state pin sequence.** It is often useful to display the state of the device in a HEX number. With this command the pins of the device can be entered to be displayed in HEX on the screen after each step. Enter the pin corresponding to the LSB first
E 1	**Execute.** The option parameter becomes the START vector number, execution continues from start to end or until an error is detected. This new START vector will be used for future operations. NOTE: the internal status at the new starting vector must not depend on previous vectors or errors will result.
G 1	**Execute.** Runs from START, stops at vector number entered as the option.
I 3	**edIt.** Points to currently active vector on the right side of the screen. With the cursor keys, the contents of any vector can be changed. Enter return when finished. The vectors in memory will be executed starting from the 'start' vector.
M	**Make a new vector and Execute.** The levels and actual device outputs will be written to a new vector inserted after the current location. Execute from START.
Q	**Quit**
R 1	**Run.** Execute the number of steps in option.
S 1	**Mark end** Sets the vector number chosen as the last vector.

U

Update and Execute

Replace the present vector with the actual device levels. Pins with levels H, L and ? will have the actual device outputs set into the vector. Execution then runs from START until the end or until an error occurs.

<return>

Single step cycle.

If no levels have changed, then step one cycle from the vectors. If any levels have changed, display the results of these manual level settings.

<space>

Run

Execute from the current vector until the end, or until an error is detected.

EXECUTION SPEED CONSIDERATIONS

Execution speed of the vectors is primarily dependent on the writing speed to the screen. For this reason, the user can specify to the **SPRINT** PAL software, the type of screen attached. IBM MDA and Hercules cards will automatically operate at full speed. CGA and EGA cards may run at a slower speed. The CGA and EGA screens can be switched into fast display mode by setting the snow control flag OFF. The snow control flag may be toggled by the F3 key in EDIT. Once this snow control is set, it is stored in the JED$$$$.CNF file. With some CGA boards, the fast mode option will cause bits of 'snow' to be seen on the screen. If this is not desired, the slow mode can be selected, also with the F3 key in EDIT. The fast mode also improves all screen menu displays.

H HANDLER INTERFACE

The optional handler interface is controlled by this command. The number of devices to be programmed with the pattern in memory is entered after the prompt. Then the 'P' key is entered. **SPRINT** will cycle through the number of devices selected, counting only the good units. The **SPRINT** handler interface kit is required. **SPRINT EXPERT** is strongly recommended due to the device reversal and pin continuity features.

I INPUT DISK DATA INTO MEMORY

The I command will load the data from the file selected into the buffer memory. The computer will display the standard **SPRINT** file input mode screen, as described in the File Input Mode Section of the introduction. The default file extension is '*.JED' unless otherwise specified in the original command line or in the configuration file.

SPRINT expects standard JEDEC format information. The JEDEC file is generated by any PLD development system, for example PALASM, ABEL, CUPL, LOG/iC, PLDASM, AMAZE, PLAN, PLDesigner, etc . The file contains all the data necessary to program the device for the desired function. If the data file contains test vectors, then these will be stored in a separate buffer for use by the 'G', 'T' and 'P' commands. Typically 2300 vectors (28 pin devices) can be stored in memory, depending on system memory capacity.

SPRINT checks and displays errors in the JEDEC file, device errors are a) the number of fuses, b) pin count and c) data checksum. Both the checksum and the filename will be displayed on the main menu screen.

If the JEDEC file contains the command G1, then the security fuse will automatically be set after programming, the word (auto) in the main menu indicates that security is enabled.

If the file extension is '*.PLD', **SPRINT** assumes that the file is a source file in PLDASM format and will call the PLDASM macro-assembler before loading the file. PLDASM will generate a temporary JEDEC file for the data. This method can be used to convert a source file to various devices for programming. PLDASM will attempt to assemle the file using the device type currently selected, the device type in the *.PLD file will be ignored.

L LIST MEMORY CONTENTS

After entering 'L' for List, the user is prompted to enter the start and ending product term for the display, separated by a comma. The user is also given the option of redirecting the output of the list to the printer, this is done by putting a ',*' at the end of the command line. The listing will show 'X' for a connection and '-' for an open 'fuse'. Lines containing all 'X' are not normally displayed, since such product terms have no logical function. If a ',X' is appended to the command line, then all product terms will be displayed. Many PLDs have unused portions of the fuse map, these unused portions are disabled in memory and are not displayed. The 'X' in the display corresponds to a '0' in the JEDEC file. The '-' corresponds to a '1'. The number of the product terms, counting from the lowest JEDEC fuse number, is displayed on the left hand side of the screen.

Some devices have additional architecture features stored after the main product term array. In these devices, the fuse number of the first architecture bit is displayed in the left column before the data.

M MANUAL PRODUCTION PROGRAMMING

IN **SPRINT**$^{\textbf{plus}}$, and **SPRINT EXPERT** installations with a single pod installed, this command sets up a counter for programming a fixed number of devices. After entering the number of devices to program, the operator need only insert the device, and type any key to program. Programming stops after the unit count is reached.

In **SPRINT EXPERT DUAL** systems, the counter is first initialized, afterwards the active socket switches from left to right. The currently active socket is shown by the Green LED. In this mode, one device can be programmed while the other socket is being loaded/unloaded. The operator need only type <space> to continue programming until the unit count is reached. This mode can double programming throughput.

The <ESC> key will turn off manual production mode. Failing devices are not counted.

O DISPLAY OPTIONAL INFORMATION

This command will cause the pod type number currently installed to be displayed. For selected devices, additional information will be displayed.

Lattice GAL (and compatible) devices support a User Signature field that can be accessed via the 'O' command. Entering 'O' will cause the signature information stored in the device in the socket to be read-out. The current User Signature data can be edited in ASCII by entering 'y' then the new signature data (up to 8 bytes). If new ASCII data is entered, it will be used in all further programming operations until a new JEDEC file is read-in. The 'O' command also displays the number of programming cycles which the GAL has had so far.

Certain other devices also have silicon signature (at this writing the AMD PALCE series, and the Cypress 22V10 and 20G10 devices). The 'O' command displays the Vpp code that was read-out during the previous read cycle. It is not valid until a read, blank check, verify or programming cycle has been executed.

These signature bytes (UES in the Lattice data sheet) can also be edited with this command. Follow the on-screen instructions. Executing this command checks the device currently installed in the POD socket, if a device with signature information is selected. Executing this command will replace any signature information previously stored in memory to be erased.

P PROGRAM DEVICE

When 'P' for Program is entered, the device in the socket will be programmed using the data stored in the memory buffer. The device type selected by the user must match the physical device in the socket. Before programming, the user is first prompted to confirm the programming of the device by typing a 'Y' or 'M', any other key will abort the programming command.

SPRINT^{plus} users should check the filter switch settings prior to programming.

The 'M' option is only supported by GAL type devices. If 'M' is entered the device will be marked as a 'master'. Any attempt to program a 'master' device will cause a special prompt to confirm re-programming.

The device will first be checked for erasure (Blank check). If the device is EEPROM based, then an erase cycle will be automatically executed.

If the device is not erased, and does not have an EEPROM cell, **SPRINT** will test if the pattern in memory can be over-programmed on top of the existing pattern in the device. If this is possible, the user will be prompted to 'skip blank check'. Entering 'Y' will over-program the device.

Then the device will be programmed using the algorithm defined by the manufacturer. Intelligent programming is used to keep the programming time to a minimum. The intelligent programming algorithm used is automatically determined when the vendor/device type is chosen.

After programming, the device will be verified to confirm proper programming. If there are any errors in verification, the user can have a detailed report on the screen. If the chip vendor specifies it, verify will be done at the Vcc high and Vcc low levels according to the data sheet. After verification, if the security fuse command was included in the JEDEC file, the security fuse of the device will be set. If test vectors were included in the JEDEC file, they will then be used to exercise the device.

The user is informed of the sequence of steps in the programming procedure by messages on the screen. For production programming, any detailed error report prompts should be answered with a 'N'. Operation then returns to the main menu, with either a PASS or FAIL shown in the status box.

Finally, a cycle counter on the screen, left of the PASS/FAIL box, displays the number of devices programmed with the same content. (SMP only).

Q QUIT

Returns the user to the **SPRINT** main menu, or to the operating system. All power to the **SPRINT EXPERT** pod is switched off when **SPRINT EXPERT** is not in use. When using **SPRINT^{plus}**, devices should not be left in the socket when the programmer is not in use.

R READ DEVICE TO MEMORY

This utility reads the device in the socket pod into the memory buffer when 'R' for Read is entered. Memory is first cleared, then the data is read-in. Note that after data has been read into memory, the device type may be changed to a compatible device without altering any data stored. Note that the device in the socket must match the device type selected or damage may result to the device due to the high voltages used to access the internal fuse structure.

SPRINT^{plus} users should verify filter switch settings prior to reading a device.

If the device appears to be empty, **SPRINT** will report a warning on the screen. A device may appear blank if its security fuse has been set.

S SET SECURITY PROTECTION FUSE

Many PLDs have an option to program some final fuses after all other fuses have been programmed. These final fuses prohibit reading (and verification) of the device. This means that the pattern set into the fuses of the device are no longer readable by those who should not have access to the pattern. These final fuses are called security fuses.

In manual mode, the user would enter 'S' followed by 'Y' to confirm the action. This must be the last step in any programming sequence since the 'P' command expects to be able to verify (read) a device after programming.

In automatic mode the security fuse will be set during each programming cycle, after verifying and before vector testing. If the JEDEC file contains the security fuse command (G1*), then automatic mode is enabled and main menu will display <Security AUTO>.

The automatic mode can be turned on from the keyboard by entering 'S' followed by the 'A' response, or turned off with the 'N' response.

SPRINT first programs the security fuse, then reads the array to verify that the data no longer matches the array. The PASS message is displayed only if the array data is no longer readable.

Some devices disable preload if the security fuse is set. For these devices **SPRINT** will run the test vector program before setting the security fuse in the 'P' programming cycle.

Some devices have a mode to read-out the security fuse bit. **SPRINT EXPERT** can read this bit and will indicate that the security fuse bit is set, however it will then perform a double check of the array contents to assure protection.

No programmer is able to read any device that has the security fuse set.

T TEST PLD USING TEST VECTORS

If a JEDEC file with test vectors has been read into memory, then the device in the socket can be tested for conforming to these test sequences with this command. Entering the 'T' for Test command will begin this test. This test will also be invoked after the verify operation in a programming cycle, if test vectors are in memory.

The Test operation has 3 passes for each vector. Pass 1 sets up all input pins. Any 0, 1, or X levels are presented to the device. All device outputs are set to read mode. **SPRINT** has a special method for handling the 'X' (don't care) level, which is incompletely defined in the JEDEC specification and often is a cause of vector test failures in other systems.

Pass 2 applies any clocks to the device. The vector is scanned from highest pin to lowest pin, and the clocks are output in that order. If a multiple clock device requires a controlled sequence of clocks, then there should be a separate vector for each clock. It is impossible for a universal programmer like **SPRINT** to generate simultaneous clocks on 2 pins. There will always be a time spacing between multiple clocks on in a test vector. Note that the 'C' clock is normally low, and the 'K' clock is normally high.

Pass 2 also executes the preload function if a P or B vector is found. A P or B in pin1 of the vector indicates preload. Note that entering and leaving preload mode can generate transient logic levels that can have other effects in a PAL design.

Pass 3 verifies the outputs of the device against the test vector. If there are no errors, the next vector is processed. If there are any errors, the good and bad data will be displayed on the screen, and testing will stop unless continued by the operator.

Pins defined as 'X' in the vectors are set as inputs to '0' unless the pin is determined to be an unused output, in which case it is floated. High impedance outputs are tested for incorrect '0' conditions, but not for low impedance '1' conditions. Test simulation can also be used to verify a device after the security fuse has been programmed, (note that AMD PALCE family devices disable preload when security is set).

The distribution diskette provided with **SPRINT** includes a file called TEST16R4.JED which demonstrates both the test vector and preload functions using the SHIFTER.PLD example. Many other examples of JEDEC files with test vectors are also provided on the distribution diskette.

The number of test vectors depends on available memory, a standard system with 640kb of memory can support 2300, 28 pin vectors. Larger vector files are possible, contact your **SPRINT** distributor for more information.

SOURCES OF TEST VECTOR ERRORS

1) PRELOAD

Preload is silicon device dependant. The same device type from different vendors use different preload sequences. Some of these sequences can interfere with normal device operation.

For example, if a Cypress 22V10 application uses pin 8 as an asynchronous reset input, then each time preload is done, the registers will be reset and the preload will fail. Other similar conditions exist will almost all devices. Use of the **SPRINT** vector editor will allow these problems to be quickly identified.

If preload of the output registers is attempted in a device that does not support preload, no high voltages will be applied, and an error message will be reported.

NOTE:	**As preload involves voltages above 5V, the device type selected must match the device in the socket. If an incorrect device is inserted in the socket, damage to that device may occur.**

2) Power-on Reset

If the first line of a sequence of test vectors fails, check that the a power-on condition is defined in the equations. It may be required to add an initialization test vector to reset/preset the device registers as the first vector.

3) Synchronous Clocks

Under certain conditions, it is possible that a device could double clock. Double clocking means that a counter or state machine advances 2 steps instead of one step. This can occur if the device Tco is faster than the clock rise time, and multiple outputs switch at the same time, causing internal ground bounce. First, be sure that the fast clock drivers are being used (with the JEDEC 'C' or 'K' commands - not 010 sequences). **SPRINT^{plus}** clock drivers can handle devices as fast as 15 ns without double clocking. **SPRINT EXPERT** clock drivers are designed to handle devices as fast as 5 ns without double clocking. Refer to section 3 for a list of available clock drivers. The solution to this problem is to disable the outputs in the vector with the clock, then re-enable the outputs and complete the test in the next vector.

4) Asynchronous or Multiple Clocks

Refer to Chapter 3 for the pins of your POD which support fast risetimes. Other pins have slower risetimes. Clocking of devices with internal asynchronous clocks enabled (EP600, ATV750, ATV2500, 20RA10 etc) from pins with slow risetimes may result in double clocking. This is application dependant, and exists only in extreme cases. The easiest solution is to disable the outputs before clocking the device.

If a device has multiple clocks, it is not possible to drive these at the same time, there will always be some hundreds of nanoseconds of delay between clocks. Use 1 vector for each clock edge in multiple clock devices.

5) Signal Sequence

Pass 1 applies levels from pin 1 to the last pin of the device. Pass 2 applies clocks from the last pin backwards to pin 1. Various equations are possible that may be affected by this sequence, for example a cross-coupled SET/RESET flip flop is possible in a 16L8 device, this could mis-function under certain vector sequences. Make sure that all test vectors are sequence independant.

6) ATVG (Automatic Test Vector Generation software)

This software works from the JEDEC file, and appends to the JEDEC file test vectors. After each use of the ATVG software, the test vectors must be verified on physical devices, to eliminate the 5 effects listed above. In particular, some ATVG packages do not correctly handle multiple clocks. No ATVG package will manage the Preload interference problem.

U UN-ASSEMBLE DEVICE

Entering this command will cause the data in memory to be converted into a PLDASM source file. This can be used to documented poorly documented PALs, to understand the operation of the PAL, or for conversion from one PAL manufacturer/type to another.

If no filename is selected, the user will be prompted with a name to be given to the data in memory. The data will be written to this file, then the Unassembler will be started. The user may enter symbolic names of up to 10 characters for each pin of the device, this mode may be skipped. Then the data will be converted to the PLDASM universal source format. See the UNASM Section of this manual for more information on the UNASM.

This is an optional feature. Contact your **SPRINT** distributor if the feature does not show on your menu, and you would like to add this to your **SPRINT** system.

V VERIFY DEVICE

Compares the contents of the device in the socket to the pattern in memory. If there are no differences, the PASS message will appear. If there are any differences, the user will have the option to display those fuses that are incorrect, as described in the MORE ? Section above. Those devices that are specified by the manufacturer for both low and high VCC verification will automatically have this performed.

SPRINT^{plus} users should verify the filter switch settings before using verify.

> **NOTE: The verify command will not function correctly if the security fuse of the device has been programmed.**

W WRITE BUFFER DATA TO DISK

Writes the pattern and test vectors in memory onto the disk using the filename supplied. After entering 'W' for Write, the user will be prompted to enter a DOS style path. A drive label, or path may optionally be entered. The default path is accepted if return is entered without any other character. Then the filename may be entered. If another file with the same name already exists, the user will be prompted to confirm erasure of the other file.

X CROSS PROGRAMMING UTILITY (PAL EMULATION)

Note that for 16V8 and 20V8 devices, no translation is required. These devices can be 'cross programmed' to emulate directly older PAL devices. See the '16V8 as' selections of the device menus. Also, devices from different vendors with the same internal structure require no translation. For example the 22V10 from AMD, Cypress, ICT, Lattice, Samsung, TI, and others can all use the same JEDEC file without conversion.

Note that universal translation is an extra option. Please contact your **SPRINT** distributor if your system does not support this UNASM/ASM method of conversion, and you would like to add this feature.

To use the cross programming utility, a device is selected from the device lists, then is loaded into memory - either from a JEDEC file or from a device in the socket. This device may now be converted to another pin-compatible device with the X command. After typing X, PAL will store the file in a temporary JEDEC file, then the UNASM will generate a temporary source file. At this point, the user will be asked to select the new device type using the standard change vendor menu. Finally the PLDASM will convert this source file into a new JEDEC file and load it into memory. Any test vectors in the original file will also be converted. It is suggested that the user store the converted file into the filename of his choice with the 'W' command.

Any errors will be reported on the screen, a common error is that the destination device is unable to emulate the source device - for example if a non-inverted output is used and the destination device only has inverted outputs. The destination device must also have the same number of pins as the source device. All errors will be clearly shown on the screen.

NOTE: If the destination device is a second source of the present device - like the AMD 22V10 and Cypress 22V10, then no conversion or translation is required. The X command will work correctly, however using the 'C' command to change device vendors is a faster and simpler method of converting between architecturally identical devices offered in different technologies If the devices are not interchangeable with the 'C' command, the memory and the checksum will be reset to zero

PLD ERROR MESSAGES

This is a summary of common error messages and how they should be managed.

1) Error: display detail report? Y/N

This message is output by blank check or verify. It indicates that the data read from the part did not match the expected data. This means that the device was not empty (blank check) or is not programmed correctly (verify).

2) Skip blank check? [N]

If a device was not blank during a programming cycle, it is possible to continue programming by entering 'Y'. However, this is not recommended as it requires detailed knowledge of the internals of the device. The normal response should be: 'N'. Either UV erase or discard the device.

3) Programming device - operation failed

The device could not be programmed. Present silicon technology allows a 99% or better average programming yield. Depending on the manufacturing week, the yield you experience may vary.

Another common cause of programming failure is with erasable devices, which have been damaged in the application. If a device is removed from a PC board, erased, and then programmed, there should be no problem. However, if the device was damaged in the PCB, programming will fail.

4) Manufactures Electronic Signature incorrect

Silicon Signature allows **SPRINT** to adapt the programing voltages and pulse widths automatically to a code provided by the vendor. The vendor provides specifications to **SPRINT** before the silicon is sold to customers, we then include these codes into the software. There are 2 possible causes for this message:

a) The device has been damaged and is no longer programmable. Some CMOS devices are sensitive to latch-up and static discharge, this is a major cause of failures in CMOS products.

b) The device is a new revision not supported by your version of the **SPRINT** software. Contact your **SPRINT** distributor for an update.

5) Preload not supported

The device type selected does not support preload of test vectors.

6) Device is a master, Program anyway?

This message occurs when programming a Lattice GAL (or compatible) device which has it's 'master' bit set.

7) Test Vectors not supported

This message occurs when using a DIP adapter or 40/68 pin devices adapter with **SPRINT**$^{\textbf{plus}}$.

JEDEC FILE STANDARDS

All **SPRINT** PLD programs (PAL, PLDASM and UNASM) create or use a standard JEDEC file usable with any PLD development software or programmer. The JEDEC file is a manufacturer independent data base that contains the following fields:

Fn* sets the reset state of the fuses to n (either 1 or 0)

QFn* defines the total number of fuses to be n

Gn* enables the security fuse to be programmed if n = 1

Lny* defines the fuses starting at fuse number n to the condition y (1 or 0). Sequential fuse locations are defined by a string of 'Y's until the terminating '*' is found.

Cn* the checksum of the fuse data is n

Vny* defines the vector test number n to be the sequence y. A test vector contains the following letters (the pin sequence starts at device pin 1)

C = positive edge clock
K = negative edge clock
1 = set pin to > 2.4 v
0 = set pin to < 0.8 v
Z = test pin for high impedance
H = test pin for > 2.4 v
L = test pin for < 0.8 v
N = pin not used
P = test vector is preload condition
B = preload/test buried registers

An example of a program for a device with test vectors and preload is provided on the distribution disk in the file "TST16R4.JED"

3. SOCKET PODS AND ADAPTERS

1. SPRINT EXPERT 'TOP' PODS

SPRINT EXPERT uses a split socket/pin driver approach for providing the widest range of device support in terms of architecture and packaging. The **SPRINT EXPERT** base unit contains the standard drivers and bus interface to the host computer. **SPRINT EXPERT** 'TOP' units contain one or more device ZIF sockets, and can easily be removed and exchanged by the user/operator.

This Section of the Manual lists various 'TOP' units and their functions. Additional information may be delivered with the actual TOP itself. These TOPs can be used with all **SPRINT EXPERT** systems. Each one contains an internal code that is read by software to avoid operator errors. **SPRINT** GANG and DUAL systems can use these TOPs for higher performance and simultaneous programming.

Each TOP is supported by the **SPRINT EXPERT** test utility. Test 7 identifies the installed TOP, and calls the correct test sequence. Any errors will be displayed on the screen.

In addition to the PLCC TOPs, any **SPRINT** PLCC, LCC, or PGA adapter may be used together with the standard TOP 40 DIP for direct programming support.

EXCHANGING THE TOP UNIT

It is easy to exchange the **SPRINT EXPERT** 'TOP'. First, exit from PAL or PROM to DOS or the SPRINT main menu in order to turn power off to the hardware. Then turn the entire unit over and loosen the brass screws which hold down the 'TOP'. With these screws loosened, simply insert two fingers into the opening in the back of the TOP, and press upwards. The TOP will come away from the base.

A different TOP is installed by positioning the tab at the front end of the TOP into the slot at the front of the base, then rotating the TOP backwards as far as it can go. A small amount of pressure can be applied to the front of the TOP to assure good connections.

The connectors are rated for hundreds of connection cycles, and will serve for many years. The brass screws may be tightened again - if preferred by the user.

TOP40 DIP STANDARD UNIVERSAL SOCKET

This is the standard TOP. Twenty Vcc and Ground relays in the combined TOP/base allows this 40 pin DIP socket to support virtually all devices in packages from 8 to 40 pins with a few exceptions. Both JEDEC and many non-JEDEC power/ground pinouts are supported. Programming voltages and pulses are controlled by the **SPRINT** ASIC, and can be applied to any pin. As new devices are announced by the silicon vendors, they will be supported by this TOP. Should a silicon vendor choose a non-JEDEC power/ground pinout which is not supported by the TOP 40, a small DIP/DIP adapter is the simplest way to add device support.

As of October 1989, the DIP devices requiring an adapter for use with the TOP40 are: 874x Microcontrollers, Texas Instruments TICPAL16xx, ECL PALs, certain application specific single chip microcontrollers, and AIM technology based products. See the table on page 18 'Special adapters' for a more detailed list.

If an adapter is required, the user will see a message on the main menu screen. If a different TOP is required, the Software will block program until the TOP is exchanged.

A combined Red/Green/Yellow LED functions as a status display for each socket. Red indicates error or programming failure, Green indicates 'PASS' or 'Ready', and Yellow indicates that programming is in process.

The TOP40 DIP can provides low impedance, relay power and ground connections to the pins listed below. The clock lines have sub 5 ns rise times for high speed synchronous PALs and EPLDs. The oscillator is used for EPROM based microcontrollers.

TOP 40 DIP RELAY AND CLOCK PINS

Signal	pin
VCC	10
	13
	14
	24
	27
	28
	29
	30
	32
	34
	36
	40
Ground	11
	14
	20
	26
	27
	30
Clock	1
	7
	9
	10
	11
	18
	21
5 Mhz Oscillator	
	18
	19

TOP432 DIP 4 X 32 PIN PROM SUPPORT

This TOP contains four 32 pin sockets for parallel programming of virtually any 8 bit CMOS or NMOS PROM, EPROM, Flash EEPROM or similar device. Except for the 8 data pins, all pins are freely configurable by SPRINT software. Devices with 24, 28 and 32 pins can be read and programmed. Each socket has its own 8 bit data driver allowing the same, or different patterns to be written to the four devices at the same time.

Device reversal and pin continuity testing is supported, preventing errors in programming. PLDs are not supported by the TOP432, for size, cost and performance reasons.

Operation is possible with only 1, 2, or 3 of the sockets filled with chips, although the SW will report a pin-continuity error if a chip is not inserted. This error message can be bypassed.

One benefit of the extra logic on the 'TOP432' is that programming and verify cycle times are even faster than the standard TOP40. This is especially useful for PC or XT type computers with slower clock frequencies since reading and verifying (but not programming) are CPU speed dependant for calculations.

The concept of the TOP432 unit allows both JEDEC EPROM pinouts and other pinouts to be supported. For example, CMOS high speed PROMs have address bits and /CE signals on different pins than the usual EPROM. High density EPROMs with ROM pinouts are also not JEDEC EPROM pinout compatible. All of this is no problem to **SPRINT EXPERT** or the TOP432.

To use the TOP432, start the program PROM432 which is supplied on floppy disk. All operations are the same as the standard PROM software with a the addition of the 'T' command. SPRINT SW will report an error if the incorrect TOP is installed.

The default condition is the GANG mode of operation. The 'T' command in the main menu switches between GANG and SET modes of operation. Eight, 16 and 32 bit SET modes are possible.

In case of EPROM error during programming or verify, the LED next to the socket will turn RED and remain RED until the next operation. If the error occurs during programming, the remain EPROMs will continue to be programmed, and the defective EPROM will not slow down the programming process.

In batch mode, the error code from a TOP432 error is 0x20 plus the bit map of the errored sockets. See the Introduction portion of the manual on BATCH mode operations.

MULTIPLE TOP432 INSTALLATIONS

One possible use of the TOP432 is with the DUAL, QUAD or OCTAL base systems. In these configurations, 8, 16 or 32 memory chips can be programmed at the same time. Each of the TOP432 units can be addressed individually, all can be accessed in a 'broadcast' mode. At most 4 different patterns (the 8 or 32 bit SET modes) are possible. That means that an OCTAL system could have 32 EPROMs with the same pattern, or 4 sets of 8 EPROMs, with 4 different patterns.

Error during programming are correctly managed, with the RED LED indicating the bad device, and programming operation continuing on the other devices.

To operate SPRINT EXPERT in multiple TOP432 mode see the 'T' command in the PROM section..

TOP 432 GANG PROGRAMMING

Gang programming refers to programming all devices with the same contents. In this simple mode, one to four PROMs can be inserted into the sockets and the P command is executed. **SPRINT EXPERT** will check how many devices are installed and program those parts. Data is read from the disk into the **SPRINT** buffer memory and programmed into the devices. The 'Read' command only addresses socket 1. The 'Verify' and 'Blank' commands check all four sockets.

TOP 432 SET PROGRAMMING METHODS

There are 3 different set programming modes as shown below.

In all set modes, the **SPRINT** internal memory buffer is first prepared for the data to be programmed, then the programming operation begins. All set mode programming uses the same programming sequence. The **SPRINT** memory buffer is initialized by reading in a file. When a TOP432 is installed, an additional item in the main menu is displayed to allow the memory buffer to be correctly managed for the set programming operation.

All four PROMs to be programmed must have compatible programming voltages and timing in order to work. We strongly recommend the use of four identical devices.

8 BIT SET

In this mode, two, three or four PROMs are programmed with sequential information in a single cycle. For example, a 1 megabyte program is to be stored in four 256k byte 27020 devices. **SPRINT** will split the original program into four sections, and will generate an intermediate disk file containing data for the combined 4 PROMs. When programming is started, the four PROMs will be written with different data sets.

Upon completion, socket 1 is the lowest address memory, socket 4 is the highest. If less than 4 EPROMs are inserted, there will be a pin continuity error - this is easily skipped.

16 BIT SET

In this mode, data for a 16 bit microprocessor (or similar) is programmed into two PROMs at once. Socket 1 is the EVEN byte and socket 2 is the ODD byte. No special organization is required for the data in the **SPRINT** internal memory buffer.

If a second set of two PROMs are inserted into the sockets 3 and 4, they will be programmed with the same information as 1 and 2 respectively.

32 BIT SET

In this mode, data for a 32 bit microprocessor is programmed into four PROMs. Socket 1 is byte 00, socket 2 is byte 01, socket 3 is byte 10 and socket 4 is byte 11. No special organization for the data read into memory is required, SPRINT will automatically program the 4 bytes of the 32 bit word into the 4 memories in the sockets.

SKIPPING AN EMPTY SOCKET

If only 1, 2, or 3 memories are to be programmed, the error message 'pin continuity' can be skipped with the <ESC> key, programming will continue normally on those memories installed.

TOP 3 PLCC AND LCC FOR 20, (24) 28 AND 32 PIN DEVICES

This triple socket unit is available in PLCC or LCC versions. Operation is identical with standard PROM or PAL programming. Since all 24 pin DIP chips are put into 28 pin PLCC packages, there are certain 'NC' no connect pins to be managed by the SPRINT SW. This is done, allowing one TOP to handle virtually all devices. Note that the test vectors for 24 pin PLDs in 28 pin SMD sockets will be automatically converted, with the no connect pins correctly skipped.

Relays are used for high-quality power and ground connections for JEDEC and non-JEDEC PLD and PROM/EPROM type devices. High speed clock drivers allow vector testing of synchronous EPLDs with access times as low as 5 ns, the 'C' or 'K' command in the JEDEC format must be used for the high speed clock drivers to be active. Note that using pins without high speed clock drivers as clocks for asynchronous EPLD designs may result in double clocking if many outputs switch at the same time.

Signal	pin	
VCC	1	20 pin package
	20	20 pin package
	1	28 pin package
	7	28 pin package
	21-24	28 pin package
	28	28 pin package
	32	32 pin package
Ground	10	20 pin package
	11	20 pin package
	7	28 pin package
	8	28 pin package
	14	28 pin package
	15	28 pin package
	21-23	28 pin package
	16	32 pin package
Clock	All	20 pin package
	1-19	28 pin package

Note: Not all Vcc, Ground and clock options are shown for the 32 pin socket.

TOP 44 PLCC, LCC, PGA

The TOP44 has 44 SPRINT pin drivers, one for each pin, to allow complete SW configurable operation. The standard version is PLCC, LCC devices are supported by the same unit with the use of 'inserts' (supplied) to convert the PLCC socket to an LCC socket. A PGA version is also available. Operation is identical with standard PROM or PAL programming. SPRINT recognizes the TOP44, when it is installed, therefore the menu will list only those devices supported. Since most 44 pin PLCC devices are also 40 pin devices, SPRINT will automatically handle the different pinouts and no connect pins. Test vectors, generated for 40 pin devices, will also work in the 44 pin equivalent package.

Relays are used for high-quality power and ground connections. Built-in oscillator emulation allows popular 87xx family single chip microcontrollers to be programmed. High speed clock drivers allow vector testing of synchronous EPLDs with access times as low as 8 ns, the 'C' or 'K' command in the JEDEC format must be used for the high speed clock drivers to be active. Note that using pins without high speed clock drivers as clocks for asynchronous EPLD designs may result in double clocking if many outputs switch at the same time.

High speed clock drivers allow testing of synchronous EPLDs with access times as low as 8 ns. Devices supported include JEDEC EPLDs, JEDEC 16 bit EPROMs, microcontrollers, EP900, MAX, ATV2500, and many more.

TOP44 POWER, GROUND AND CONTROL PIN DRIVERS

Signal	pin
VCC	1
	3
	11
	14
	25
	27-30
	32
	34
	35
	36
	40
	44
Ground	10
	12
	13
	20
	22
	33
	34
	43
Clock 1	
	2
	7
	9
	11
	24
	34
	44
5 Mhz Oscillator	
	3
	4
	20
	21

TOP 68 PLCC, LCC, PGA

The TOP68 has 68 SPRINT pin drivers, one for each pin, to allow complete SW configurable operation. The standard version is PLCC, LCC devices are supported by the same unit with the use of an 'insert' (supplied) to covert the PLCC socket to an LCC socket. A PGA version is also available.

Relays are used for high-quality power and ground connections. High speed clock drivers allow vector testing of synchronous EPLDs with access times as low as 8 ns, the 'C' or 'K' command in the JEDEC format must be used for the high speed clock drivers to be active. Note that using pins without high speed clock drivers as clocks for asynchronous EPLD designs may result in double clocking if many outputs switch at the same time.

Devices supported include both JEDEC and popular non-JEDEC 68 pin EPLDs, the EP1800, Altera/Cypress MAX and other popular pinouts.

Operation is identical with standard PROM or PAL programming. SPRINT recognizes the TOP68 when it is installed, therefore the menu will list only those devices supported. One advantage of the 68 pin drivers provided by the TOP68, is that vector testing of the programmed device is now possible.

TOP68 POWER, GROUND AND CONTROL PIN DRIVERS

Signal	pin
VCC	3
	4
	18
	20
	21
	27-30
	32
	34
	36-38
	52
	54-55
Ground	1
	15-16
	32-33
	35
	49-50
	66-67
Clock	1
	7
	9
	11
	17
	36
	51

2. SPRINT^{plus} PODs

The **SPRINT^{plus}** PC/XT/AT plug-in board has 28, universal pin drivers. The standard 28 pin POD has one pin driver for each socket pin and therefore can program thousands of different devices in DIP packages, using the same socket, without adapters. The 28 pin POD may be exchanged for 32 and 40 pin MegaPODs, or for a 40 pin 8751/52 POD. These POD options contain extra logic to allow the **SPRINT^{plus}** system to expand the number of address pins available for higher density EPROMs.

In addition, many plug-on adapters are available to support SMD components, newer devices using the center pinning JEDEC format, and other special devices. A complete list of currently available adapters is available from your **SPRINT** distributor.

The type of POD installed can be displayed using the Option command of either PAL or PROM software. The Test utility also displays the POD type installed, and performs the correct test sequence for each different POD type.

EXCHANGING THE POD

The user can exchange the PODs connected to the PC/XT/AT plug-in card by unplugging the 40 pin ribbon cable from the back of the computer, and plugging-in the cable from the different POD.

Warning:

While this operation can be done with power applied, a misalignment of the connector could cause a short circuit to the SPRINT^{plus} card. We recommend that power be turned off when exchanging pod types.

FILTER SWITCHES

SPRINT^{plus} has five filter switches which allows the 28 pin socket to support a very wide range of components. Switches 1 and 2 engage Vcc bypass capacitors for 24 and 20 pin devices. Switches 3 and 4 engage Vpp bypass capacitors for different types of EPROMs. Switch 5 is especially provided to improve the programming yield of the Cypress 7C29x type of high speed PROM when set to the off condition.

When a device is selected via the 'Change Device' menu, or when starting **SPRINT**, there is normally no device in the socket. **SPRINT^{plus}** will read-back the switches to verify their settings and will inform the user if the switches read back incorrectly.

Note that using Auto ID mode, with EPROMs, may cause the setting of switch 3 to be incorrectly read back due to the presence of the device in the socket. The condition can be ignored, the user must check that the switches are correctly set.

The design with these filters switches allowed the POD size of **SPRINT^{plus}** to be kept very small, and has given **SPRINT** the lowest Vcc and Vpp noise during programming and vector testing. The result has been many certifications from silicon vendors.

Additional noise elimination circuitry in the PODs allow even the highest speed memory and logic components to be reliably programmed.

Warning:

Do not attempt to lengthen the ribbon cable. Extensive testing has shown that the cable length provided with the POD is the maximum. SPRINT EXPERT allows longer cable lengths.

28 PIN UNIVERSAL DIP PODS

There are two standard 28 pin PODs. **SPRINT^{plus}** users with serial numbers greater than 799 should have POD type 5B. Users with serial numbers lower than 800, (as well as those users who have upgraded their QuickPro systems), should have POD type 7.

With the 28 pin POD, the standard ground is pin 14, and Vcc can be driven to pins 24, 26 and 28 for 20, 24 and 28 pin corner pinning devices.

These PODs include relays to also handle the Cypress version of the 28 pin center pinning package for the 7C330 family and the 28 pin MAX family devices. This relay allows Vcc to be applied to pin 22, and grounds to pins 14 and 21.

The Type 7 POD has a serial number PAL located inside of the POD. During startup or power-off of any of the **SPRINT** software utilities, there should be no device inserted the socket.

The POD type is marked on the bottom of the POD plastic box, and type 5 or 7 can be recognized by software.

The type 5B POD contains additional noise reduction circuitry. Any type 5 or 5A pod can be customer-modified to a 5B POD, if you do not currently have a 5B POD, contact your local distributor for update information.

For type 5B PODs, or modified 5 and 5A PODS, the capacitors inside the POD on pins 3 and 15 must be 0.1 uF, not 0.022uF. These capacitors are connected to a 2.2K resistor and a diode. These modifications are required for new generation high-speed EPLDs and can be customer-installed without affecting the warranty.

32 PIN MEGABIT EPROM POD

SPRINT^{plus} offers two MegaPOD options. The MegaPOD32 supports 28 and 32 pin devices in the JEDEC standard EPROM pinout. The kinds of EPROMs supported ranges from the 2764 (8k x 8) to 27020 (256k x 8). Paged EPROMs (for example the 27011) are also supported. New Flash EPROMs with various densities in 32 pin packages are programmed with the MegaPOD32.

The MegaPOD32 also executes the complete programming cycle for Quickpulse type EPROMs in significantly less time than is required by the 28 pin universal POD.

The **SPRINT**^{plus} PROM utility software automatically identifies the 32 pin MegaPOD and adjusts the menus to display only those devices that are supported in this POD.

All other PROM operations are unchanged.

When using Auto ID for EPROM type selection, set switch 1 'on' for 28 pin devices, and 'off' for 32 pin devices.

40 PIN MEGABIT EPROM POD

The MegaPOD40 supports 16 bit wide devices in the industry standard 40 pin package. The kinds of EPROMs supported ranges from the 27257 (16k x16), up to the 274096 (256k x16).

The **SPRINT**^{plus} PROM utility software automatically identifies the 40 pin MegaPOD and adjusts the menus to display only those devices that are supported in this POD.

All other PROM operations are unchanged.

No filter switches are used with this POD.

8751 MICROCONTROLLERS: POD51 AND ADAPTER S422

There are two ways to program the standard 8751 microcontroller with **SPRINT^{plus}**. Either the adapter can be used, or the special POD can be used.

The Intel 87C51FB device requires the POD51. Smaller devices can use either the POD or the adapter.

Operation of the PROM utility is the same as described in Chapter 1 of this Manual.

The security fuse setting is controlled by the last byte in the memory buffer. Bit 0 is fuse 0, bit 1 is fuse 1 (if provided in the silicon).

Note that the POD51 cannot be automatically detected by the **SPRINT^{plus}** software.

SPRINT^{plus} DIP ADAPTERS

As of February 1990, the following devices require special adapters in order to be programmed by **SPRINT^{plus}**:

Number	Device
S413	EP900, EP910, 5C090, 5C091
S414	EP1210, 5C121
S417	PLS104, PLS105, PLS105A, PLUS105
S422	8751, 87C51, 87C52, 87C51FA and similar devices
S434	General Inst. PIC 16C52/54
S435	General Inst. PIC 16C53/55
S436	28 pin JEDEC center Vcc/Ground: 26V12, 26CV12
S445	40 pin 87C75

SPRINT EXPERT AND SPRINT^{plus} ADAPTERS

Devices with non-standard pinouts, special packaging, SMD components and devices with unusual programming requirements need some kind of adapter for use with any universal programmer. **SPRINT** offers a complete line of adapters for those devices that do not fit, for various reasons, into the standard **SPRINT** PODs.

There are three categories of devices requiring adapters:

A) SMD components
B) Unusual DIP Pinouts or SMD packages
C) Special timing or clocking interfaces

As far as possible, SPRINT standard PODs and TOPs will support all devices, when this is not possible, an adapter is required.

For SMD components, either the PLCC, LCC or PGA 'TOPs' or the adapters can be used with SPRINT EXPERT.

GENERAL PURPOSE SMD ADAPTERS

As of February 1990, the following list of SMD adapters are available. Contact your local **SPRINT** distributor for an updated list of adapters.

SPRINTplus requires these adapters for all devices.

SPRINT EXPERT can use these adapters together with the standard TOP 40 DIP unit. Use of an SMD adapter instead of the exchangeable TOP unit can be done at any time. Notice that the NC (no connect pins) of many 24 pin devices in 28 pin packages are not always in the same location, please check that the pinout and the adapter match before programming.

When using one of these adapters, the device is programmed as if it were a DIP device. All references by the test vectors to pins refer to the DIP pinout. Always insert the adapter with the pins in the lower half of the socket.

Number	type	Function
S401	LCC	20 pin universal
S402	PLCC	20 pin universal
S403	SOIC	24 to 28 pin universal
S404	SOJ	20 pin universal
S405	LCC	32 pin universal (for Mega32 or EXPERT)
S406	LCC	24 pin device in 28 pin package. Example 22V10
S407	LCC	28 pin EPROM in 32 pin package
S408	PLCC	24 pin device in 28 pin package. Example 20L8
S409	PLCC	24 pin device in 28 pin package. Example 22V10
S410	JLCC	EP600, EP610, 5C060
S411	JLCC	EP900, EP910, 5C090
S412	JLCC	EP1210, 5C121
S415	LCC	EP1800, 5C180
S416	PLCC	EP1800, 5C180
S419	LCC	8751 family (**SPRINT**plus use only)
S420	PLCC	8751 family (**SPRINT**plus use only)
S424	JLCC	44 pin EPROM, like 27210
S425	PLCC	32 pin universal (for Mega32 or EXPERT)
S430	PGA	EP1800, 5C180
S438	PLCC	28 pin EPROM in 32 pin package
S441	PLCC	ATV2500

SPECIAL PURPOSE DIP ADAPTERS

As of February 1990, the following list of special adapters are available. Contact your local **SPRINT** distributor for an updated list of adapters.

SPRINT EXPERT can use these adapters together with the standard TOP 40 DIP unit. When using one of these adapters, the device is programmed as if it were a DIP device.

Number	**Function**
S418	ECL1 adapter for N.S., Aspen, Signetics
S421	8748, 8749
S423	8741AH, 8742AH (note AH type)
S426	75P1xxC devices in shrink DIP packages
S427	Signetics 87C751 DIP
S428	Signetics 87C751 PLCC
S429	75P3xx devices in QFP packages
S431	8741A, 8742A (note A type)
S439	TIC16xx devices
S442	NEC uPD 77P56 speech processor, DIP
S442	NEC uPD 77P56 speech processor, SOP
S444	Signetics 87C552 PLCC

4. SPRINT EXPERT MULTISYTE SYSTEMS

SPRINT EXPERT can be configured for multiple programming sites. Up to 16 base units (sites) may be connected at one time. The SPRINT EXPERT concept for GANG programming (where 2 or more devices receive the same information) is based on 2 key issues:

A) Each socket must be fully functionally isolated from other programming sites.

B) There must be no loss of programming speed in GANG mode.

C) Any SPRINT EXPERT TOP may be used.

D) Device reversal and pin continuity testing are possible on each device.

E) Errors in one device during programming do not stop or slow down the programming of the remaining devices.

Issue A requires that each socket have it's own pin drivers - up to 104 pins each - in order to program any mixture of EPLDs, EPROMs or any Logic or Memory device. Issue B requires that a method be used to allow all sockets to be verified in parallel during the programming and post-programming verify cycles. SPRINT EXPERT, with the SPRINT BUS, does this. Issue C means that existing and future TOPS can be used (in GANG operation, all TOPS must be the same type). Issues D and E require that each TOP be individually selectable at high speeds.

SPRINT EXPERT Systems configured with more than one base unit are listed below:

SPRINT EXPERT DUAL:	2 base units	
SPRINT EXPERT QUAD:	4 base units	
SPRINT EXPERT OCTAL:	8 base units	
SPRINT EXPERT HEX:	16 base units	(future)

Because of the additional power requirements, MultiSyte systems have their own built-in AC power source. This is switchable between 110 and 220 volts, with a UL/VDE approved power supply and mains switch on the back of the unit.

Operation of a MultiSyte configuration is identical to the standard SPRINT system with 2 exceptions:

A) The device list for GANG programming may differ from the single site programming of SPRINT EXPERT. This situation is temporary as the devices are re-qualified for GANG mode.

B) The DUAL system contains a special additional feature - the SWAP mode.

See the PROM and PAL sections for more information on the operation of the 'M' and 'T' commands for selecting and SW configurations for the MultiSyte systems.

4. PLDASM

PROGRAMMABLE LOGIC DEVICES

Programmable Logic Devices (PLDs) (also known as PALs) are popular devices for implementing digital designs. These devices can be used where earlier systems used TTL or CMOS logic ICs. The PLDASM is a tool that allows Boolean equations to be programmed into a PLD in order to perform a user-defined logic function. Boolean equations make it possible to describe a function in an efficient manner, and this assures that the designer achieves the most compact solution with the fastest propagation delays. Furthermore, with Boolean equations the PLD can function as an address decoder, state machine or counter, and perform any number of other tasks ranging from the simple to the complex. While initially PLDs provided a savings in the amount of space used on a PC board, recent high speed PLDs are often significantly faster than the equivalent circuit implemented in TTL logic. Another recent development in PLDs is the complexity of the macrocells used for I/O. PLDASM automatically configures these macrocells, according to a set of simple rules which apply to all the PLDs supported by PLDASM. This allows substitution of one device for another, and reduces the amount of time required to 'learn' a new PLD.

A PLDs internal structure is built as an AND/OR matrix. A programmable input AND array can generate any AND function of all device inputs (with or without inversion). These AND functions are called 'Product Terms'. Product terms feed a multiple input OR gate. Since the AND/OR matrix can express any Boolean transfer function, the flexibility and functionality of a PLD is limited only by the number of terms available in the AND - OR arrays. PLD devices are available in different sizes, some with over 40 inputs, and some with up to 19 Product Terms per output. The outputs range from simple tri-state drivers to complex registered macrocells with programmable inverters.

BOOLEAN FUNCTIONS

In an unprogrammed PLD, all fuses are intact. In other words, every input line is 'ANDed' with all other input lines (including any feedback terms available in the device) The output of these AND functions is fed into an OR gate and is then either fed onto more complex functions or presented directly on the output pins of the device.

For example, let us assume that we have a simple PLD with two input terms (A and B) and two output terms (X and Y). Internally, the device also makes the inverse of the input terms available (/A and /B). In the unprogrammed state, the logical function of the device can be represented by the following Boolean equations.

$$X = A*B + /A*/B + A*/B + /A*B$$

$$Y = A*B + /A*/B + A*/B + /A*B$$

In this state clearly the device has little use, X and Y are always equal to 1, regardless of the inputs A and B. However, when some of the terms in each of the AND functions are removed, the power of the device becomes obvious. For example, let us assume that the following fuses are 'blown':

from X, /A*/B, A*/B

from Y, A*B, /A*/B, /A*B

In the example given, the fuses were 'blown' so that no connection remained. The equations that remain after programming of the device are shown below.

$$X = A*B + /A*B$$

$$Y = A*/B$$

As can be seen, very quickly it becomes possible to provide complicated logic functions in a single package. The other main advantage of PLDs is that their precise function can be adapted by the individual designer to meet the application needs, even if the design specification changes after PC boards have been built, (or if bugs are found during system testing and production).

The above equations are usually entered into a disk file using an editor such as Wordstar or Sidekick. Be careful to avoid printer control codes which are created by programs such as Word or Wordperfect. The disk file is passed through the PLDASM to create a JEDEC file. The JEDEC file can be easily loaded into the PAL program for programming a device. If desired, the equations can be viewed using the PAL Editor function. Please note that a blown fuse is represented by a '1' in the JEDEC file, or as a '-' in the XPLOT or Editor. An intact fuse is represented by a '0' or a 'X'.

BOOLEAN TO JEDEC TRANSLATION

Obviously, the PLD itself is not able to understand Boolean equations in the form given above. It is therefore necessary to translate the information from simple Boolean equations into a form that may be used to program the PLD.

In order to program a PLD, it is necessary to address each fuse in the device individually and to either blow it (create an open circuit) or to do nothing and leave it intact. The **SPRINT** PLD programming utility uses a fuse map to determine which fuse to blow and which fuse to leave intact. The **SPRINT** PLD Macro-Assembler creates a fuse map from an ASCII text file which contains Boolean equations. The fuse map that is created is called the JEDEC file.

For each input signal, there are two input line numbers, one for the actual input signal and one for its inverse. So, for this device there will be four input line numbers (1 = A, 2 = /A, 3 = B, 4 = /B). Additionally, there will be eight product line numbers as there were eight OR combinations in the unprogrammed device (4 for each output term). Therefore, for this device, the fuse map needed by the programming utility to create the Boolean functions described is shown below.

Product Line Number	Input Line Number 1 2 3 4	
1	X - X -	
2	- X X -	
3	- - - -	X
4	- - - -	
5	X - - X	
6	- - - -	
7	- - - -	Y
8	- - - -	

The fuse map shown here is stored in a JEDEC file where each fuse location represented by an 'X' is stored as a '0' (zero) and will be unaffected by the programming utility. Each location represented by a '-' is stored as a '1' and will be blown by the programming utility.

The program that produces this fuse map is the **SPRINT** PLD Assembler.

SPRINT PLD MACRO-ASSEMBLER

The **SPRINT** PLD Macro-Assembler is designed to convert an input file, created by any editor, into a standard JEDEC file for use by the **SPRINT** EPLD/EPROM programmer. The JEDEC file can also be used with other types of programmers.

The source file consists of definitions of all the pins in the device, including power and ground pins, followed by Boolean equations for the function to be performed by each output pin in the PLD. **SPRINT** PLDASM supports a flexible Macro feature which reduces the amount of data that has to be typed-in for the PLD equations. This Macro function also makes the input file easier to read and understand, for better documentation of the application.

The input file is entered using any editor on the IBM PC. If the editor inserts control characters for right justification or other purposes, the non-document mode (as in Wordstar) should be selected. This is because the **SPRINT** PLDASM Macro-Assembler will not be able to understand these control characters.

The completed source file is assembled into a JEDEC file using PLDASM. Errors in the source file are identified and reported to the user in clear text. The line number/character number where the error was detected is displayed on the screen for use in correcting the error.

The **SPRINT** PLDASM automatically performs two passes in the assembly process. The first pass sets the macrocell output stage in high-density PLDs, the second pass fills in the equations. Device macrocells are not to be confused with the Macro function of the **SPRINT** PLDASM.

Shown below is an example of the use of PLDASM which demonstrates the operation. From the main menu of the **SPRINT**, PAL program, typing an 'A' for Assemble will load the PLDASM Macro-Assembler and display all files in the **SPRINT** default directory with the '.PLD' extension. The user can position the highlight to the filename desired and enter return, or a filename may be typed-in after the prompt as shown below:

```
Enter name of file :
```

PLDASM assumes a file type of '.PLD', so if the desired file has the file extension '.PLD', then only the name need be entered, otherwise the full filename and file extension needs to be entered:

```
Enter name of file : EXAMPLE
```

If the file EXAMPLE.PLD is on the default drive, it will be used by the **SPRINT** PLDASM as the source for the assembly, and a JEDEC file with the filename EXAMPLE.JED will be created on the same drive and in the same directory. If the source file resides on a different drive or in a different directory or with a non default file extension, then the expanded entry would be as shown here:

Enter name of file : B:\PLDASM\EXAMPLE2.PLA

The resulting file will appear on drive B:, directory PLDASM as EXAMPLE2.JED. The result file always has the file extension '.JED'.

For a list of all devices currently supported by the PLDASM, press the <Esc> key instead of entering a filename.

COMMAND LINE OPTIONS

The PLDASM can also be started with several command line options :

A > PLDASM [-i Filein] [-o Fileout] [-d Device] [-x]

The options are used as follows:

a) -i This forces the input file name to the one following the i character. No file select menu will appear. This is suitable for batch operation.

b) -o The resulting JEDEC file may have a name different than the source file. The name after the 'o' character will be used instead of the default - which is the source file name.

c) -d The device defined in the source file may be replaced with any other device - the overriding device name follows the 'd' character.

d) -x If this option is included - the resulting JEDEC file will not be written to the disk - it will be compared to the JEDEC file of the same name and all differences will be reported.

SPRINT PLDASM SYNTAX

A source file for the **SPRINT** PLDASM consists of the following parts:

Comments (Documentation)

Device selection

Pin definition

Macro definition

Equations

End

All text in the source file can be either upper or lower case. **SPRINT** PLDASM ignores the case of the text. **SPRINT** PLDASM will ignore all blank lines included in the source file.

COMMENTS

Comments may be entered on any line in the source file. Comment fields within the source file are started with the character '. All text after the ', until the end of the line are ignored. Comments can be placed anywhere in the input file. All text preceding the device selection denoted by the word 'DEVICE' is ignored and may be used as an extended comment field.

LABEL DEFINITION

A label in the PLDASM consists of any string of up to 10 ASCII alphabetic or numeric characters. The first character of a label must not be a 1 or a 0. A '/' may precede the label to indicate active low functions. Some words are reserved, see the list below.

The labels '1' and '0' are reserved words and indicate fixed logical 1 and 0 conditions respectively. See the complete list of reserved words later on in this section.

DEVICE DEFINITION

The word DEVICE (or device), in the source file, is the logical start of the data to be assembled by PLDASM. The word DEVICE must be in the text of the source file starting in column 1 of a new line to be recognized by the **SPRINT** PLDASM Macro-Assembler. The device type to be used by the assembler must follow the word DEVICE, there must be a space between DEVICE keyword and the type selected. Currently supported devices are listed on the screen by entering a <ESC> instead of the filename (see above). Note, device types entered are only generic device types, the manufacturer of the device to be programmed should not be included in the device definition line. It may be included as a comment field if desired by the user.

Example:

DEVICE 20L10

device 16c1

SECURITY FUSE CONTROL

The security fuse in a PLD may be automatically set during programming by putting the keyword 'security' in the source file, between the device definition and the keyword 'start'. The output JEDEC file will include a command to the PLD programmer to set the security bit of the device after programming and verification. The default condition is to leave the security fuse intact. Setting the security fuse prohibits anyone from reading stored data from a PLD

PIN DEFINITION

After the device definition, and before the keyword 'START' (see below), the input pins, the output pins and Macros are defined. Typically the pins are defined first, but Macros and pin definitions may be mixed.

Only used pins need to be defined at this stage. The pins for VCC and ground may also be defined. During assembly, the **SPRINT** PLDASM assembler will verify that the pin definitions given match the device that was defined in the device definition. If the pin definition is incorrect or does not match the device definition, the **SPRINT** PLDASM assembler will produce an error and a JEDEC file will not be created.

The pin definition consists of a label followed by a pin number. There may be an optional equal ('=') sign between the label and the pin number. The pin number is the decimal physical pin number. One pin definition per line is allowed.

NOTE:	**Some macrocell functions are defined at pin definition time. See the application note section for details on the device being used.**

MACRO DEFINITION

A MACRO definition consists of the word MACRO in column 1 of a new line. The MACRO keyword is followed by a label (the MACRO name) followed by a string of characters up to 180 characters long (end-of-line characters are ignored). The terminator is the character ';'. A common error when defining MACROs is to omit this terminating character. The ';' character is not included in the Macro. PLDASM allows up to 40 macros.

Later, in the equations, the Macro name can be inserted into the text where the string is to be substituted. In the equations, the Macro name is preceded by a '&'. An example is shown below:

```
....

MACRO test1  /a13*/a12*/a11* a10* a9;

....

START

....

output = &test1* a1;
```

The AMD PALASM example of a Barrel Shifter is included on the distribution floppy diskette to show the use and advantage of Macros.

Macros cannot be inverted, so a '/' may not precede the macro name in the equations.

EQUATIONS

The keyword 'START' in column 1 signifies the end of the pin and MACRO definitions. The text following the 'START' keyword, until the 'END' keyword is found in column 1 of a new line, are the equations which define the functions of each output pin in the specified device. The equations are assembled into the JEDEC file ready for programming into the PLD' device chosen. Illegal combinations, such as using a dedicated input pin for output, are identified by the **SPRINT** PLDASM assembler and will be reported with line and character number.

Each equation line consists of the following parts:

output pin	label
enable (optional)	label.ena
function (first line)	macrocell function
input equations	set of 'AND' equations
+	indicates additional OR'd Product Terms
^	indicates an exclusive OR'd Product Term
;	end of equation for this pin

FUNCTIONS

All devices have outputs, some devices allow hidden 'nodes'. PLDASM treats both as macrocells. The Boolean result of the equations entered for a macrocell can be applied directly to the output, or can be synchronized by either D or T type registers. In addition, the equation can be inverted, allowing the application of the De Morgen theory to reduce the number of product terms required. These inversion and register options are defined with the syntax shown below:

macrocell functions (outputs)

=	combinatorial equation result
/=	active low combinatorial result
:=	D type registered result
/:=	active low D type registered result
^:=	T type registered result
/^:=	active low T type registered result

The data sheet of the device selected will indicate the types of outputs allowed. Incorrect selections will be reported as errors by PLDASM. If a LABEL is assigned to a physical device pin and the output enable is active, (by default or explicitly controlled with the '.ENA' modifier), the equations stored inside of the device will define the function of the output pin. For those devices with hidden functions, the LABEL could be an internal NODE, and the equation would define the function of that NODE.

The '/' character before a label signifies inversion. **SPRINT** PLDASM enforces the concept that the label in the pin definition and the output pin label in the equations, must have the same inversion status. Use a '/=' or a '/:=' in the FUNCTION field to indicate that the result of the equation is active low. When converting from a MMI/AMD PALASM file, simply move the '/' from before the output pin to the macrocell function pin, i.e. before the = or :=.

Example:

MMI/AMD equation.	/OP1 := IP1*IP2 + IP3
SPRINT PLDASM equation	OP1 /:= IP1*IP2 + IP3

INTERNAL NODES

Every output of an AND or OR equation in a device is a node. Most nodes are buried inside a device, and do not have to be given names. For example the eight AND functions in each output stage of the 16R8 go into a fixed OR gate, there is no need to reference these points, they are hidden and are automatically managed

Sometimes other internal functions need to be defined, for example the RESET and PRESET product terms available in many PALs.

PLDASM assigns these internal function with pseudo pin numbers called 'node numbers'. The ARESET and SPRESET and other built in nodes require no definition by the user. See the reserved word list below.

Many new devices have internal registered or combinatorial feedback logic that is either always hidden inside the device, or can be hidden by the user. PLDASM assigns these internal logic elements a pseudo 'pin' number that is higher than the number of pins in the physical package. To use a NODE, simply assign a LABEL to the number shown in the table, and write the output equations in the same way as for a physical output pin. See the section on Dual Feedbacks and Buried Registers for more information on the use of nodes.

ACTUAL BOOLEAN EQUATIONS

After the output (or node) has been specified, and the macrocell function is defined, the actual Boolean equations for each output stage are listed. The equations consist of groups of AND and OR terms.

AND terms are specified by the '*' symbol. For example, A 'AND' B is specified by the equation 'A * B'. OR terms are specified by the '+' symbol. For example, A 'OR' B is specified by the equation 'A + B' and A 'XOR' B is specified by the equation 'A ^ B'. Each equation for each output pin must be terminated with a ';'.

Each AND term consists of LABELs, and MACROs separated by '*' and terminated by either a ';' or '+' or '^'. The ';' terminator denotes the end of the equations for this pin. The '+' and '^' terminators separate AND terms, it indicates that these AND terms are OR'd or XOR'd together.

The LABEL is a standard PLDASM label, with an optional '/' in front to indicate inversion. MACROs can be used to replace any combination of LABEL, '/', '*', '+' or '^'. MACROs are invoked by an '&' in front of the MACRO name defined above. Macros cannot be inverted, so no '/' may precede the MACRO name.

The example in this chapter shows the use of MACROs within the input equations.

CLOCKS

Traditional PALs and EPLDs use a single clock for all flip flops (pin 1). In these devices the PLDASM assumes that the clock comes from pin 1. Some newer devices allow a clock Product Term to be defined for the each flip flop. These devices are usually called Asynchronous PALs. PLDASM can generate this clock Product Term via the '.CLK' modifier which allows an equation to be written for each of the clock Product Terms. In the EP600, EP900 and other devices with a selectable synchronous or asynchronous clock, the PLDASM will automatically default to synchronous unless a '.CLK' modified Product Term is defined.

OUTPUT ENABLE

Some devices have the ability of producing tri-state outputs on some pins and hence the ability to enable or disable the output driver on those pins. The determination of whether to enable or disable the output from these pins may be defined in a separate equation, thereby only enabling the output to be high or low under certain logical conditions. This extra programming is not mandatory. If no enable equation is included in the equations for any output pin supporting this function, **SPRINT** PLDASM will assume that the output is continually enabled.

In order to specify that the desired output pin only be enabled under certain logical conditions, an equation needs to be defined for the conditions under which the output should be driven. The equation defining this is virtually the same as a standard output equation, except that the OUTPUT PIN definition in the first column is suffixed with '.ENA'. This is called a pin modifier. Other modifiers are also possible - please see the next section for more information. If the '.ENA' enable is used, then two equations need to appear for that output pin, one to enable the output and one for the logic state of the output. The function that is allowed in an enable equation is '=' or '/=' depending on the device. The enable equation must be terminated with a ';'. An example of the enable function is included in this manual to demonstrate its use.

Standard PALs have the output enable function hard-wired for registered outputs to pin 11 or pin 13 (20 or 24 pin devices). If an equation is provided to connect the output enable to these hard-wired connections, PLDASM will check to verify that this hard-wired connection exists in the physical PAL. The feature is provided to allow upwards compatibility to devices which are socket compatible to the older PALs, but also offer a programmable output enable Product Term, for example the 18U8 and 20G10 type of devices. It is recommended to always include the hard-wired enable equation:

```
LABEL.ENA  =  /PIN11
```

when using older 16R4 or similar type PALs to allow for easy conversion to new, CMOS EPLDs. This approach is also used by the UNASM to allow conversion from one type to another.

If the selected device has a direct connect path to an I/O pin which can be used to bypass the output enable Product Term (for example the Cypress 20G10), then the PLDASM will automatically select the higher speed direct connect path if the output enable equation includes only one pin, and that pin matches the physical direct connect path.

CLOCKS, XOR AND OTHER SPECIAL FEATURES

Many newer device support special features in the macrocells, for example XOR Product Terms, async clocks, resets etc. PLDASM uses the same concept as the .ENA for other function. Examples of these files are on your PAL diskette in the Examples sub-directory. A modifier is added to the pin name used in an output equation. An operator is used in the equation itself. A list of modifiers and operators is shown below:

modifier	function	typical devices supported
.if	force feedback from the I/O pin	See list
.rf	force feedback from the register	See list
.cf	force feedback from the PT OR gate	EP310
.ena	output enable	most
.clk	output register clock, asynchronous	7C331 EP600 20RA10 ATV750 and others
.rst	output register reset, one register only	7C331 20RA10 EP600 ATV750
.set	output register preset, one register only	7C331 20RA10
.xor	XOR Product Term	7C331 20X4/6/8 20XR4/6/8 etc
.iclk	input register clock Product Term	7C331
.irst	input register reset	7C331
.iset	input register set	7C331
.j	J input of JK flip flop	78C800
.k	K input of JK flip flop	78C800

Operators separate the inputs in the equations. Three kinds of operators are allowed:

operator	function	devices supported
*	AND	all
+	OR	all
^	XOR	20X4/6/8

FUSE

Some PAL or EPLD devices have fuse functions which cannot be effectively described in the PLDASM syntax. Presently, the only device that fits this limitation is the Ricoh 16LC8 family. To use these unusual functions, write the code as if the extra fuses were not set, then set each fuse with the statement 'FUSE xxxx;' where xxxx is the number of the fuse to be specially programmed. FUSE can be used in any device to force any condition. It is suggested that FUSE not be used without a detailed understanding of the device being programmed.

Example:

```
FUSE 1234
```

will set fuse number 1234 to a 1. Note, the default condition for all fuses is 0. Always put the 'FUSE' statement at the end of your files.

VECTORS

Test vectors may be inserted into the source file by using the label 'VECTOR' followed by the exact test vector sequence as used in the standard JEDEC file. The purpose of this feature is to allow the UNASM to store the test vectors found in a JEDEC file in a form that can be used to re-assemble the source into another device - without loss of the test vector information. See Chapter 2 for information on the test vector standard. We suggest that test vectors be generated using the PIN EXERCISER function of PAL.

END OF FILE

The keyword 'END' indicates the end of the file. If the **SPRINT** PLDASM program terminates without the message 'Assembly Complete', then it is likely that the keyword 'END' has been omitted from the source file. Successful completion of the **SPRINT** PLDASM assembler will result in a JEDEC file being written to the disk and a message on the screen signifying a successful completion of the operation.

RESERVED WORDS

Some words are reserved for use by the PLDASM, these must not be used as labels. The words are found in 3 groups: Assembler Directives, Device Special Features, and Device Built-In Nodes.

1) Assembler Directives are control words for the PLDASM itself. Note that the numbers '0' and '1' can be used to set a Product Term to all OFF(0) or ON (1):

DEVICE	LABEL	END	START	
MACRO	VECTOR	FUSE	0	1

2) Device Special Features control internal fuses in the device:

SECURITY	this enables automatic security fuse setting after programming
MISER	see device data sheet
TURBO	see device data sheet
ZERO	set zero power standby mode

3) Device Built-In Nodes:

label	note	device examples
ARESET	global	22V10, and others
SPRESET	global	22V10, and others
OBSERVE	global	23S8
$CLRA		78C800
$CLRB		78C800
$LEA		78C800
$LEB		78C800

Device Built-In Nodes are used in the same manner as any other LABEL in PLDASM. For example:

```
ARESET = label1 * label2;
```

SPRINT PLDASM EXAMPLE

The directory \EXAMPLES on diskette #2 contains source code examples for many devices, especially ones with unusual functions. You can use these examples to understand the use of the PLDASM. One example, the 16R4 will be explained here in detail.

The attached example shows a file using the PLDASM and MACROs. The file is from the AMD data book on PALs.

The first section is the header text: all data up to the reserved word 'DEVICE' in column one is assumed to be comment and is ignored by PLDASM. In the example, this explains who wrote the file, and some background information as documentation for the reader only.

Then the DEVICE is defined, here it is a PAL16R4. The JEDEC file created for a device is the same for all manufacturers of the same device, so no vendor need be specified, only the generic device type.

Then the pins, and MACROs are defined. Any sequence may be used, but typically the numeric order is the easiest to read.

The keyword 'START' indicates the beginning of the equations. The output pin label starts in column 1. It has the same polarity (a / in front) as the definition above.

The output pin /ZERO has been added as an example of the use of the output enable. When the inputs S0, S1, and S2 are all 0, the output will be enabled. It will be high if any of the data bits (D0 to D7) are 1, otherwise it will be low. If the inputs S0, S1, S2 are not all 0, then the output will be high impedance.

The keyword 'END' indicates the end of the file. After the detection of this keyword, the **SPRINT** PLDASM will write the translated JEDEC file with the file extension '.JED' to the disk. The data can then be read by **SPRINT** by an input command with the same filename as the original '.PLD' file.

The resulting JEDEC file is shown after the '.PLD' file.

```
PAL DESIGN SPECIFICATION
PAT001          KEVIN M. OW-WING 6-22-85
4-BIT SLICE FOR AN 8 BIT BARREL SHIFTER
ADVANCED MICRO DEVICES
Modified to show the output enable function.
device 16r4;
CK          1
D7          2
D6          3
D5          4
D4          5
D3          6
D2          7
D1          8
D0          9
GND         10              'This is a comment
/E          11
/EQU0       12
S0          13
Q0          14
Q1          15
Q2          16
Q3          17
S1          18
S2          19
VCC         20

macro       f0              /S2*/S1*/S0;
macro       f1              /S2*/S1* S0;
macro       f2              /S2* S1*/S0;
macro       f3              /S2* S1* S0;
macro       f4              S2*/S1*/S0;
macro       f5              S2*/S1* S0;
macro       f6              S2* S1*/S0;
macro       f7              S2* S1* S0;

start

Q3 /:=      &f0*/D3 +  &f1*/D2 + &f2*/D1 + &f3*/D0 +
            &f4*/D7 +  &f5*/D6 + &f6*/D5 + &f7*/D4;

Q2 /:=      &f0*/D2 +  &f1*/D1 + &f2*/D0 + &f3*/D7 +
            &f4*/D6 +  &f5*/D5 + &f6*/D4 + &f7*/D3;

Q1 /:=      &f0*/D1 +  &f1*/D0 + &f2*/D7 + &f3*/D6 +
            &f4*/D5 +  &f5*/D4 + &f6*/D3 + &f7*/D2;

Q0 /:=      &f0*/D0 +  &f1*/D7 + &f2*/D6 + &f3*/D5 +
            &f4*/D4 +  &f5*/D3 + &f6*/D2 + &f7*/D1;
' Now define the output enable for the "/ZERO" pin

/EQU0.ENA   = /S0*/S1*/S2;
/EQU0       /= /D0*/D1*/D2*/D3*/D4*/D5*/D6*/D7;
end
```

EXAMPLE JEDEC FILE
BARREL SHIFTER

```
*
F0*
QF2048*
L512 11101110111111111101111111101111*
L544 11101110111111111111101111011111*
L576 11101101111111111111111110101111*
L608 11101101111111111111111111011011*
L640 10011110111111111111111111101111*
L672 11011010111111111111111111011111*
L704 11011101101111111111111111101111*
L736 11011101111110111111111111011111*
L768 11101110111111111111101111101111*
L800 11101110111111111111111110011111*
L832 11101101111111111111111111101011*
L864 10101101111111111111111111011111*
L896 11011010111111111111111111101111*
L928 11011110101111111111111111011111*
L960 11011101111110111111111111101111*
L992 11011101111111111011111111011111*
L1024 11101110111111111111111110101111*
L1056 11101110111111111111111111011011*
L1088 10101101111111111111111111101111*
L1120 11101001111111111111111111011111*
L1152 11011110101111111111111111101111*
L1184 11011110111110111111111111011111*
L1216 11011101111111111011111111101111*
L1248 11011101111111111111101111011111*
L1280 11101110111111111111111111101011*
L1312 10101110111111111111111111011111*
L1344 11101001111111111111111111101111*
L1376 11101101101111111111111111011111*
L1408 11011110111110111111111111101111*
L1440 11011110111111111011111111011111*
L1472 11011101111111111111101111101111*
L1504 11011101111111111111111110011111*
L1792 11101110111111111111111111101111*
L1824 10111011101110111011101110111011*
C76c0*
0000
```

SPRINT PLDASM OPERATION

After starting PLDASM, the filenames with the type '.PLD' on the current default directory will be displayed. Use the cursor keys to highlight the desired file and enter 'RETURN', or enter the filename (the file type default to '.PLD') then 'RETURN'. If the message 'File Not Found' is displayed, please check the filename and path and try again.

Entering <ESC> instead of a filename will cause a list of all supported devices in the current release to be displayed on the screen.

After the file is selected, then the processing of PASS1, PASS2 and the writing of the JEDEC file to the disk will be reported on the screen.

If any errors are detected, they will be reported with the line number and character number of the position in the file where the error was detected. The type of error will be displayed with one of the messages described overleaf. The line containing the error, and the previous 4 lines will be displayed on the screen for easy identification. An arrow points to the location where PLDASM detected the error. In some cases this may be a few characters after the actual error location. Note that the character position counts tabs as one character. The error messages are designed to be self-explanatory, additional comments to the types are listed below:

SPRINT PLDASM ERROR MESSAGES

Error Message	Explanation
Device not supported	See list of currently supported devices.
Macro definition error	Macro too long, or too many Macros.
Reserved word	Some words are reserved for use by the PLDASM, see the reserved word list. In any case a reserved word was found in an unexpected location.
Too many/few pins	Attempt was made to define pins not on the physical device.
Missing terminator -> ;	A';' is required at end of the line.
Not enough product terms	The user attempted to use more OR terms than in the physical device
Label not defined	Check spelling, labels need to be listed in the definition portion of the file.
Improper use of pin	Attempt was made to use a pin in a manner not supported by the physical device. For example an active high output on a PAL16L8.
Use /= or /:= for output inversion	See section on OUTPUT PIN
Output enable error	Attempt was made to use an output enable on a pin of a device which does not have it.
Name already defined, or missing START	A LABEL has been defined twice, or the directive START is missing.

Error Message	Explanation (continued)
Pin cannot be used as input	The physical device does not allow this pin to be used as an input
XOR not allowed on this pin	XOR is not supported.
Feedback from this pin is not possible	The pin is output only. Occurs in GAL direct mode for example.
This node does not allow inversion	Node is active high only.
Missing keyword or ; character	Usually occurs when End Of File is reached before the directive END
Missing operator - either ^ or + needed	Occurs when a two labels are shown without an operand between them.

APPLICATION NOTE:

This section provides some special notes on the use of PLD devices with special macrocell features. The traditional MMI PAL type of structure as used in the 12L6 or 16R4 type of devices forms the basis for all devices in the PLDASM. Devices with higher pin counts and more Product Terms do not change the basic concept. Even those parts with macrocells are still compatible with the original concept, the macrocells only give the user the important freedom to re-configure certain pins to match the task better. All devices supported by PLDASM, except those listed below, conform fully to the original MMI concept. The parts listed here have some extra features that require device specific explanations.

PLDASM supports automatically all device features, the user need only focus on the application. Detailed error messages from PLDASM will inform the user when a device does not support the functions attempted.

PROGRAMMABLE FEEDBACK OPTIONS

Some devices (marked * in the following list) support independently programmable feedback options. Normally a PAL or EPLD has one feedback per macrocell additional feedback options compared to all other PAL type devices. The PLDASM will default to treat these macrocells in the same manner as the standard MMI type PAL devices unless overridden by feedback options in the label definition. The feedback override option types are:

.CF Combinational - output of or gate in macrocell

.RF Registered - output of register in macrocell

.IF I/O pin - feedback is from the device I/O pin

The default feedback conditions are:

.RF for registered outputs (:= macrocells)

.IF for combinational outputs (= macrocells)

The designer can force .IF, .CF or .RF by adding this name to the label in the label definition. For example:

```
DTACK.IF  15;

COUNT.RF  16;
```

This will cause the PLDASM to use the device pin 15 as feedback. The output pin can be registered or not, without affecting the selection of .IF . Likewise, the register in the macrocell of pin 16 will be used for feedback into the device, even if pin 16 is set for combinational mode.

HIDDEN (BURIED) REGISTERS

Some devices have the ability to have flip flops that are used inside the device without any direct access to the I/O pins. These are called buried or hidden registers. PLDASM assigns these flip flops pseudo pin numbers called 'internal nodes'. A list of devices with internal nodes is shown below. The node numbers are like pin numbers, except that pseudo pin number is higher than the actual number of pins in the device package.

DUAL FEEDBACKS

Many new devices now support dual feedback input terms. For many years this concept was known to offer the possibility to double the use of the I/O pins in the package, by 'hiding' the registers in the I/O macrocells. However the price for dual feedbacks was device speed, since the extra feedback lines increased the number of input terms by 50% - making a larger array - which is slower. Now some parts have even dual and also hidden registers. PLDASM supports dual feedbacks by providing pseudo pin numbers called internal nodes for the extra feedback. This allows the register in a I/O macrocell to have a different name than the I/O pin which is connected to that macrocell.

PLDASM uses the default concept for all dual feedback macrocells. If an output is registered, then the default feedback will be from the register. If an output is combinatorial, than the default will be from the I/O pin. If the second feedback is required, then the internal nodes (pseudo pins) must be used. This is best explained in the four possible cases:

CASE 1: registered output, pin feedback not used. This is coded like any PAL16R8.

CASE 2: combinatorial output, register not used. This is coded like any PAL16L8.

CASE 3: registered output, both register and pin feedback is used. The register section is coded like any 16R8, the pin feedback is the internal 'node' that is assigned to that pin. If the output enable is disabled, then the register is hidden.

CASE 4: combinatorial output, register is used. This example implies that the equation going to the 'D' of the register is available on the I/O pin. The combinatorial section is coded like any 16L8, the feedback from the register output can be referenced via the internal 'node' that is assigned to that pin.

In the next example, we see that pins 26, 27 and 28 do not exist in the physical device, but do exist as internal nodes. In bit0 of our presettable counter, the register is not hidden, and it can be read-out with the /OE control, the default feedback is from the register, but in order to do preload from a bi-directional bus, we need to access the I/O pin with a different name. In bit1, the register cannot be read-out, the pin is used as an input, and this input has a function not directly related to the register, the I/O is accessed via the internal node. In bit2, the pin is combinatorial, therefore the feedback has the same name as the output signal. If we wish to access the register's Q output, then we use the internal node for this pin. As we see, the internal node refers to the feedback path not selected as the default condition.

DUAL FEEDBACK EXAMPLE:

```
device EPL204

' pins with registered outputs

reg0            19      'pin 19 registered output
dbus0           28      'pin 19 I/O pin

hidreg1         18      'pin 18 register
preload         27      'pin 18 I/O pin

' pins with combinatorial outputs

hidreg2d        17      'D input of pin 17 register
                        ' also I/O pin name
hidreg2q        26      'Q output of pin 17 reg

' regular input pins

OE              2
dbus1           3

start

reg0.ena        = /OE;
reg0            := reg0  * /preload             ' case 3
                 + dbus0 *  preload;

hidreg1.ena      = 0;                           ' hidden reg.
hidreg1         := hidreg1 * /preload           ' case 3
                 + dbus1   *  preload;

hidreg2d         = hidreg2q * /preload          ' case 4
                 + hidreg2d *  preload;


end
```

LATTICE GAL16V8/20V8, AMD PALCE16V8/20V8.

These devices have three basic configuration modes. While the data sheet shows 5 different macrocell combinations, the 16V8 and 20V8 silicon does not allow all of these to be selected at the same time.

1) The default mode is the 'dedicated' mode. In this mode all outputs are combinational, no output enable is permitted, but there are up to eight Product Terms per output stage. The outputs cannot be used as inputs (no feedback).

2) The combinational output mode is selected by PLDASM if any of the outputs in the equations have an output enable term (note that a 'LABEL.ENA = 1;' may be used for the always enabled condition). In the GAL16V8/20V8 combinational mode, the two outermost macrocells are output only (no input or feedback), the remaining six may be used as output or input or bi-directional. The equations may have up to seven Product Terms per output, plus one output enable.

3) The third mode is the registered mode, this mode is selected if any of the outputs have been set as a registered output. In this mode, registered outputs have eight Product Terms and a pin 11 (13) direct coupled output enable, while combinational outputs have seven Product Terms and a programmable output enable.

In these devices, an additional set of fuses is provided to turn-off unused Product Terms in order to reduce power consumption. These fuses are automatically set by PLDASM for the lowest possible power consumption.

The signature bytes described in the data sheet can be edited in ASCII with the 'O' command in 'PAL'. Any data entered with this editor can be stored in a JEDEC file with the 'W' command.

PLX TECHNOLOGY PLX448/464

The dual feedback node numbers for pins 13,15-23 are 32-39. These devices also have an open collector fuse option for pins 13,15,22,23. In order to select open collector outputs, use the FUSE statement to set the correct bits as shown below:

Fuse 5098	pin 23 open collector
Fuse 5101	pin 22 open collector
Fuse 5112	pin 15 open collector
Fuse 5115	pin 13 open collector

CYPRESS 20G10

This device supports a direct output enable path from pin 13. While the normal user will generate an output enable (if required) with a Boolean equation, the speed of the pin 13 direct path is about 5 ns faster. This direct path will automatically be selected by PLDASM if the equation for the output enable of any pin contains pin 13 only.

FPLAS

The FPLAs supported by PLDASM are of the folded array or expander concept. An example is the EXEL 78C800. The PLDASM will automatically solve the equations to match the device resources. AND/OR combinations will be assigned to internal Product Terms. A simple minimizer assures that expanders are only used when a NOR function is required. Note that the outputs of an expander can only be OR'd, not AND'd since the expanders only have one, active high, output. The expanders may be assigned node numbers in order to build Set/Reset flip flops, otherwise it is normally not necessary to assign them node names. For the 78C800, see the special file, 'EXEL.DOC' on the PAL disk for examples.

INTERNAL NODE NUMBERING

This section lists some special devices and their internal node numbers.

Cypress CY7C331

Pin	Node	(12 input flip flops on pins 28 to 15)
28	29	
27	30	
26	31	
25	32	
24	33	
23	34	
20	35	
19	36	
18	37	
17	38	
16	39	
15	40	

AMD 23S8: Buried registers 0 to 5 are mapped to nodes 21 to 26

EXEL 78C800 see the detailed examples in 'EXEL.DOC'

Atmel ATV750: Each macrocell has 2 registers, Q0 and Q1. If no NODES are used, then the output of Q0 will be coupled to the physical I/O pin, Product Term sharing from Q1 will be automatically enabled if required. If the user wants to hide Q0, and use the I/O pin as an input, then the output of Q0 is accessed via the node number shown below. Q1 is always hidden and is only accessed via the Node number shown below. An example ATV750.PLD file is provided on the PAL disk.

Pin	Q0 Node	Q1 Node
23	32	42
22	33	43
.	.	.
.	.	.
15	40	50
14	41	51

Ricoh EPL204, all output macrocells have dual feedbacks. Pins 12-19 have dual feedback nodes 21-28.

DEVICE NAMES AND FEATURES

PLDASM uses the name shown in the first column to select a device. Note that some vendors have used the same name for incompatible devices (Lattice and Signetics 16V8 for example). Thus two names are provided. All devices use the standard syntax, some have additional features as listed below:

Name	Vendor	Specials
6l16	MMI	
8l14	MMI	
10h8		
10l8		
12h6		
12l6		
12l10		
14h4		
14l4		
14l8		
16c1		
16h2		
16l2		
16l6		
16l8		
16lc8	Sprague	
16n8	TI	
16rc4	Sprague	
16rc6	Sprague	
16rc8		
16p8		
16r4		
16r6		
16r8		
16rp4		
16rp6		
16rp8		
16v8	Lattice	Automatic Mode selection

Name	Vendor	Specials
*18cv8	ICT	
18l4		
*18n8	TI	
18p8	AMD	
18u8	AMD/MMI	ARESET, SPRESET
20c1		
*20cg10	ICT	ARESET, SPRESET
20g10	Cypress	
20l2		
20l8		
20l10		
20r4		
20r6		
20r8		
20ra10	MMI	Async clocks
20rp4		
20rp6		
20rp8		
20x4		XOR
20x8		XOR
20x10		XOR
20xrp4	AMD	XOR
20xrp6	AMD	XOR
20xrp8	AMD	XOR
20xrp10	AMD	XOR
20v8	Lattice	Automatic Mode selection
.22cv10z	ICT	ARESET, SPRESET
22p10	AMD	
22v10	CYP, AMD, TI and others	ARESET, SPRESET
22vf10	AMD	ARESET, SPRESET
*22vp10	TI	ARESET, SPRESET
22xp10	AMD	
*23s8	AMD	ARESET, SPRESET, OBSERVE, buried register nodes 21-26
.26cv12	Lattice	ARESET, SPRESET
.26V12	AMD	ARESET, SPRESET
32vx10	MMI	ARESET, SPRESET, dual feedbacks
*5c031	Intel	ARESET, SPRESET
5c032	Intel	Turbo, Miser
*5c060	Intel	Async clocks, T type Registers, turbo

Name	Vendor	Specials
7c331	Cypress	Async clocks, T type registers
78c800	Exel	FPLA structure, see EXEL.DOC
e16p8	National	ECL
e301	Cypress	ECL same as E16P8 function
*ep310	Altera	ARESET, SPRESET
ep320	Altera	Turbo, Miser
*ep600	Altera	Async clocks, T type, turbo
ep900	Altera	Async clocks, T type, turbo
epl10p8	Ricoh	Extra features accessed via FUSE
epl12p6	Ricoh	statement in these RICOH parts
epl14p4	Ricoh	
epl16p2	Ricoh	
epl16p8	Ricoh	
epl16rp4	Ricoh	
epl16rp6	Ricoh	
epl16rp8	Ricoh	
.epl204	Ricoh	dual feedbacks, nodes 21-29
plc16v8	Signetics	different JEDEC map than Lattice
.plc18v8z	Signetics	
plc20v8	Signetics	different JEDEC map than Lattice
.plc22v8z	Signetics	
.plx448	PLX Tech	dual feedbacks, open collector, ARESET, SPRESET
.plx464	PLX Tech	dual feedbacks, open collector, ARESET, SPRESET
.v750	Atmel	Async clocks, dual feedbacks

Notes: **Devices marked (*) have programmable feedback options. These default to the MMI original 16R6 style register feedback or I/O pin feedback. The default can be overridden using the .if, .rf, .cf label modifiers in the pin definition to force I/O feedback, Register feedback or Combinatorial feedback. Note that Combinatorial feedback (.cf) is not possible with all devices. Check the data book for details of the output macrocells.**

EP600, EP900 JK AND RS FLIP FLOPS, USING T FLIP FLOPS

In the data sheets for these devices, the output macrocells are shown with D, T, JK and RS flip flops. In the physical silicon, there is only a D and a T option. The user must create out of the T type, the JK and RS type of flip flops.

This note can be applied to any EPLD with T type flip flops, for example the 32VX10, 7C330, 7C331, 20Xxx etc.

The following general purpose equation will explain how to code the JK option. The RS option is similiar.

EXAMPLE:

```
Q.J     :=  A * B;        ' this is the function required
Q.K     :=  C * D;
```

EP600 solution:

```
Q       ^:= A * B * /Q
          + C * D *  Q;
```

NOTE: EP600 output inversion method:

```
/Q      ^:= A * B * /Q    ' active low ouput
          + C * D *  Q;
```

(the use of ^:= causes Q to toggle when the equations are valid
the use of /^:= causes Q to toggle when the equations are <u>not</u> valid)

20X8 solution:

```
Q       /:= A * B * /Q    ' 'J' function
          + C * D *  Q    ' 'K' function
          ^ Q;            ' make 20X8 'D' into 'T'
```

5. SPRINT UN-ASSEMBLER OPERATION

UN-ASSEMBLER UTILITY - UNASM

The **SPRINT** UNASM is an extra option. Call your **SPRINT** distributor for details on upgrading to the UNASM if the message 'SMP required' appears on the screen. The **SPRINT** Un-Assembler converts JEDEC files back into PLDASM source files. This tool allows existing PALs to be documented and changed easier than working with the X plot editor. The Un-Assembler also preserves any test vectors in the JEDEC file, and allows them to be re-assembled into a JEDEC file after changes have been made - including changing the device type selected.

PAL EMULATION - TRANSLATION

The Un-Assembler is a key part of the **SPRINT** universal PAL/EPLD translator (Emulator). This function allows one PAL or EPLD architecture to be converted into another device with a compatible pinout. **SPRINT** supports cross programming directly in the PLD programmer utility. Simple translations of PAL to 16V8/20V8 type devices are managed by this cross programming function.

More complex conversions, for example a 20RS4 to 22V10, 26CV12 to 26V12, 16V8 to 18V8 (etc, etc) are often possible, but are beyond the scope of the PLD programmer. **SPRINT** provides a universal syntax for PLDs, this means that the same source file can be assembled into various devices from different manufacturers.

This translation/emulation function is automatically executed from the PLD utility with the 'X' command.

STARTING UNASM

The **SPRINT** Un-Assembler can be started either from the DOS prompt '>' or from the PAL main menu. The user will be shown a list of '.JED' files and may select any file, or change drive and directory, in the same manner as the input command described in the PAL or PROM sections of this manual. Typing <Escape> will display a list of supported devices in the version on your disk. This list is being expanded regularly. Please contact us if the device you need is not included in your version.

After selecting a file, the user will be asked to specify the device type to use in converting the JEDEC file to source. Any errors in selecting the device type will be reported on the screen.

COMMAND LINE OPTIONS

UNASM may be started in batch mode, with the user options included into the DOS command line after 'UNASM'. The options available are shown here:

> UNASM [-Ifilename] [-Ofilename] [-Ddevice] [-N]

-I causes the filename immediately following to be used as the input file. The input file screen will not appear;

-O causes the filename immediately following to be used as the output file name. Any file on the disk with this filename will be erased;

-D causes the device type following the -d' to be used. The device type request message will not appear;

-N causes the pin name request to be skipped.

DEVICE LIST

The list of currently supported devices may be shown on the screen of the computer by pressing <ESC> instead of selecting a file.

USER PIN NAMES

Once the device type has been selected, the user is given the option to assign pin names to the device. If a 'Y' is typed, an image of the device will be shown on the screen, and the user can enter a pin number, then the desired name to any pin. This is repeated until a '0' is entered instead of the pin number. The pin name may contain a '/' in front - indicating that it is a logically inverted signal. The addition of the '/' to a name does not change the function of the device - it only serves to make the Boolean equations easier to read.

UNASM OPERATION

Once all input parameters have been specified, UNASM will read the JEDEC file, convert it to source and write it to the disk with the file type changed from '.JED' to '.PLD' (unless the -O option was used). Any errors will be displayed on the screen.

To allow the source file to be used to generate a functionally equivalent device in another technology or another vendor, the source file contains all device internal features explicitly defined. For example the 16R4 device will show all registered outputs with the output enable as connected to pin 11. A 18CV8 or EP310 will have any feedback options which are not the standard default registered or combinatorial (like the 16R4 outputs) to be marked with the .rf, .if and .cf options as explained in the PLDASM manual.

If any test vectors are included in the JEDEC file, they will be shown in the source file with the keyword 'vector'. The sequential vector number will be shown as a comment. The user can edit and insert vectors in the source, they will be converted into JEDEC format vectors by the PLDASM.

Other functions such as security (automatic setting of the security fuse after programming), 'MISER' or 'TURBO' will also be displayed in the source file with the appropriate keyword. This also preserves compatibility during translation. Note that since 'TURBO' and 'MISER' bits do not affect the logical function of the device, only the speed/power features, the PLDASM will ignore these commands in devices that do not support them.

APPENDIX A

TECHNICAL NOTES

SPRINT^{plus} TECHNICAL DESCRIPTION

Size	IBM PC standard full length card.
I/O address	Uses port addresses 300-31F hex. Base address switchable to any address between 200 and 3E0 HEX. 32 port addresses are required.
Power	5V 1.0 amp + 12V 1.0 amp (peak) .4 amp average - 12V 0.05 amp
Socket pod	This is the external socket for connection to the device to be programmed or read. It provides a 28 pin .3 and .6 inch wide socket for compatibility with a wide range of devices. Five 'Filter' switches are located on the pod for bypass capacitors according to manufacturers' published programming specifications: 1 = Vcc filter capacitor for 24 pin parts 2 = Vcc filter capacitor for 20 pin parts 3 = Vpp filter capacitor for socket pin 1 4 = Vpp filter capacitor for socket pin 22 5 = current limiter for socket pin 20
Ground	A relay can select device ground as socket pin 14, or pin 8 and 21. This relay is included on socket pods type 2, 5 and 5A.
Timer	On board real time clock. Pulse widths are programmable from 10 us to 25.5 ms in 200 ns steps.
Vpp	Two separate voltages programmable from 5 to 25.6 volts in 100 mv steps. A DC-DC converter on-board operates from the +12V supply. These voltages can be switched under software control to any device pin.

Vcc

Programmable from 0 to 9.0 volts in 50 mv steps. Drawn from the PC's +12V supply. Switched to pins 24, 26 and 28 of the socket for 20, 24 and 28 pin devices. Switched to pin 22 by the relay in pod type 5 and 5A.

Pin drivers

28 pins can each be programmed to be:

Output:

Logic 0.	Output from a 7407 driver.
Logic 1.	5 Volts via a 4.7K resistor.
Vpp1	50 ma per pin, 300 ma total.
Vpp2	50 ma per pin, 300 ma total.

(Pulse currents up to 800 ma for under 100 us)

Input:

74HCT type input, current limited with a 10k resistor.

Memory

The system must have enough free memory available to store the **SPRINT** software and device data. Memory required by **SPRINT** (not including MSDOS and resident programs) is shown below :

a) PROM,

up to 27256	190 K
27512	254 K
271024	318 K
272048	446 K
274096	446 K + hard disk overlays

b) PAL

all devices	256 K

c) Other programs

128 K

TECHNICAL NOTES

SPRINT EXPERT TECHNICAL DESCRIPTION

Size	IBM PC short length card.
I/O address	Uses port addresses 300-30F hex. Base address switchable to any address between 200 and 3E0 HEX. 16 port addresses are required.
Power	5V 1.0 amp + 12V 1.0 amp (peak) .4 amp average - 12V 0.05 amp
Socket base	This is the external base unit for holding various TOP socket units for connection to the device to be programmed or read. It provides the SPRINT bus interface, 40 pin drivers, 4 high speed clock drivers, and relays for 8 standard power pins. Two DIN 41512 connectors establish the connection between the base and the exchangeable TOPs.
Socket 'TOP'	Any one of several TOP socket units that are available. Set section 3 for details on each TOP.
SPRINT BUS	a 34 line bus capable of driving up to 16 base POD units for gang operation.
Ground	Relays in the TOP unit provide high quality, low impedance grounds suitable for support of high speed devices.
Timer	On board, crystal controlled, real time clock driven pin sequencer. Accurate pulse widths are programmable from 1 us to mulitple seconds in 200 ns steps.
Vpp	Two separate voltages programmable from 5 to 25.6 volts in 100 mv steps. A DC-DC converter on-board operates from the +12V supply. These voltages can be switched under software control to any device pin.
Vcc	Programmable from 0 to 9.0 volts in 50 mv steps. Drawn from the PC's +12V supply. Switched to pins various pins via relays.
Vpullup	Programmable pullup voltage for the active 1 signal level. Range is 0 to 12 volts in 100 mv steps.

Pin drivers

up to 104 pins (base + TOP) can each be programmed to be:

Output:

Logic 0.	4 ma drive to ground.
Logic 1.	Vpullup via 10K resistor.
Vpp1	200 ma per pin, 900 ma total.
Vpp2	200 ma per pin, 900 ma total.

(Pulse currents up to 800 ma for under 100 us are supported)

Input:

CMOS input, 1.5 volt threshold, protected up to 30 volts.

Memory

The system must have free memory available to store the **SPRINT** software and device data. Memory required by **SPRINT** (not including MSDOS and resident programs) is shown below :

a) PROM,

up to 27256	190 K
27512	254 K
271024	318 K
272048	446 K
274096	446 K + hard disk overlays

b) PAL

all devices	256 K

c) Other programs

128 K

APPENDIX B

INSTALLATION

HARDWARE INSTALLATION

SPRINT can be installed in any 8 or 16 bit slot of the IBM PC/XT/AT or compatible computer. **SPRINT** operation is completely independent of CPU clock speed, systems from 4.77MHz to over 25 MHz have been successfully used with **SPRINT**. Simply open the computer, and plug the card into a spare slot. **SPRINT** uses the port addresses 300-31F Hex. If you are aware of another card in your system with the same I/O port address, you should change the address of the **SPRINT** (or other card) now.

After the chassis is closed, the socket pod can be plugged into the connector on the back of the **SPRINT** PC card.

SPRINT^{plus} uses a 40 pin connector on the back panel. The flat ribbon cable connector to the socket adapter is keyed and can only be inserted one way, it is long enough to allow the adapter to be placed on top of the system next to the display.

SPRINT EXPERT uses a 37 pin D type connector. It cannot be inserted incorrectly. This cable is about 100 cm long, allowing tower type computers to be on the floor, while the POD is on the desktop. This long cable is also suitable for handler applications.

We have chosen the address 300 Hex for the board because very few other products have used this address for I/O cards. If you have problems during the software installation section, (usually **SPRINT** will display an error message from the TEST utility), please first temporarily remove other add-on cards from your system; if the problem goes away, that means there is an I/O port address conflict. Either **SPRINT** or the other board using address 300-31F hex needs to be changed.

I/O PORT ADDRESSING

After installation of the hardware, we suggest that the TEST program be executed in order to verify that the system is operating correctly. This system test will also identify if there is a conflict in I/O port addressing with any other card in your system. If so, the **SPRINT** board address can be changed. Jumpers are provided on the **SPRINT** board for this purpose

The I/O port address of the **SPRINT** card defaults to 300 Hex. Other addresses will be activated if a configuration file is found when the program is started.

The name of this file is:

SPRINT EXPERT	EXIOADD.CNF
SPRINT^{plus}	IOADD.CNF

The contents of this file are simply the 3 character HEX address of the physical board. Simply type:

```
DOS>    COPY  CON:  EXIOADD.CNF
        300
        <F6>
        <ENTER>
```

This file must be located in the directory containing the SPRINT driver SW.

CHANGING SPRINT^{plus} HARDWARE I/O PORTS

The jumpers are located between chip positions F0 and F1, near the metal plate at the end of the card, with jumper 1 in the lowest and jumper 4 in the highest position. Jumpers 1, 2 and 3 are shorted in the PC-board and must be cut to change according to the following table.

address	jumpers inserted	
220 *	4 1 2	
240 *	4 1 3	
260	4 1	
280	4 2 3	
2A0	4 2	
2C0 *	4 3	
300	1 2 3	(factory setting)
340	1 3	

CHANGING SPRINT EXPERT HW I/O PORTS

There are four jumpers on the board for setting the I/O address. The standard address is 300 hex (1,2,3 installed). The jumpers can be easily moved to set another address. **SPRINT EXPERT** requires 16 consecutive addresses, from the base address to base + 0x0F.

address	jumpers inserted	
200	4 3 2 1	
220 *	4 3 2	
240 *	4 3 1	
260	4 3	
2A0	4 2	
2C0 *	4 1	
2E0	4	
300	3 2 1	(factory setting)
340	3 1	
360	3	

The addresses marked * should be avoided, since other PC/XT/AT functions are assigned to these ports.

SOFTWARE INSTALLATION

The **SPRINT** software is distributed on two diskettes. The contents are shown on the next pages.

The files with the '.EXE' file extension are the programs that constitute the utilities described in this Manual. The files with the '.HEX' file extension are examples of simple PROM files that are stored in HEX formats. INTEL.HEX is the Intel HEX format, MOTOROLA.HEX is the Motorola S-Record Format and SPACE.HEX is the space format. The files with the PLD extension are examples in the **SPRINT** Macro PLDASM format. The file TST16R4.JED is a sample JEDEC file including test vectors and preload.

The other files on the disk are provided for your information and should be read. The README file will contain release notes for the version of software that you received with your **SPRINT**. PLDNEWS contains updates for new devices in the PLDASM.

Dates and lengths of the files will change as new versions are distributed.

First you should make a backup copy of the programs on these diskettes onto 2 other floppy disks. (The **SPRINT** diskette is not copy protected). If you work in a floppy-only system, you can use the backup disks directly. Keep the **SPRINT** master diskettes in a safe place. (There is a nominal fee for replacement of lost disks).

After plugging in the board, boot the system with your standard MSDOS boot diskette or hard disk. If your system has a hard disk, insert Disk #1 (PROM) into your floppy drive and type INSTALLH. This batch file will install all **SPRINT** programs into a directory on the hard disk called **SPRINT**. It will first process the PROM files on Disk 1, then ask you to insert Disk 2 for the PAL files. The sub-directory UTILITY and EXAMPLES will be created and will be set up. Older versions of **SPRINT** will be replaced automatically. On completion, the TEST utility will be executed and the test menu will be displayed.

For floppy disk only systems, insert the **SPRINT** distribution floppy disk #1. Included on this floppy is a file called TEST2.EXE (**SPRINT^{plus}**) or TESTEX.EXE (**SPRINT EXPERT**) located in the directory UTILITY. Execution of this program will display a menu of test routines.

If the hardware port addresses have been changed, the IOADD.CNF file **must** be changed.

Volume in drive A is **SPRINT**_1
Directory of A:\

UTILITY	<DIR>		14.10.88	9.35
EXAMPLES	<DIR>		14.10.88	9.35
INSTALLH	BAT	634	23.10.88	9.00
PROM	EXE	155562	20.10.88	9.00
SPRINT	EXE	13680	11.09.88	9.00
UNASM	EXE	50405	12.09.88	9.00

Directory of A:\UTILITY

.	<DIR>		14.10.88	9.00
..	<DIR>		14.10.88	9.00
TESTx	EXE	28384	21.09.88	9.48

Directory of A:\EXAMPLES

.	<DIR>		14.10.88	9.35
..	<DIR>		14.10.88	9.35
DEMO	COM	1024	5.08.88	9.47
INTEL	HEX	512	1.07.88	9.00
MOTOROLA	HEX	256	1.07.88	9.00
SPACE	HEX	132	1.07.88	9.00

Volume in drive A is **SPRINT**_2
Directory of A:\

EXAMPLES	<DIR>		14.10.88	9.00
PAL	EXE	208170	22.10.88	9.49
PLDASM	EXE	70220	23.10.88	9.33
TST16R4	JED	5888	1.07.88	9.13
INSTALLH	BAT	634	23.10.88	9.55

Directory of A:\EXAMPLES

.	<DIR>		14.10.88	9.00
..	<DIR>		14.10.88	9.00
20RA10	PLD	1625	31.07.88	9.44
22V10	PLD	6272	31.07.88	9.33
3TO8DMUX	PLD	3328	14.08.86	9.29
7C331	PLD	2376	31.07.88	9.00
EM78C800	PLD	2102	9.10.88	9.44
EX78C800	PLD	1821	9.10.88	9.42
SA78C800	PLD	6027	9.10.88	9.49
T16R4	PLD	1066	8.01.87	9.28
T6L16	PLD	928	31.07.88	9.47
T8L14	PLD	937	31.07.88	9.47
TEP600V	PLD	6250	31.07.88	9.54
EXEL	DOC	4096	27.10.00	9.53

SPRINT EXPERT INITIAL SYSTEM CHECKOUT WITH TESTEX

Once the system hardware and software are installed, the initial checkout can be done. Executing test will automatically run through the following tests:

* readback of all inputs
* test POD type and relay functions
* test internal hardware
* test **SPRINT** bus
* test pin drivers

After these tests are complete, the PASS message will be displayed. The user can then exit the TESTEX utility. If any error messages are displayed, check for I/O address conflicts. If there are no conflicts, and error messages are still being displayed, contact your **SPRINT** distributor.

TESTEX can also be executed in calibration mode. This menu mode instructs the user in all actions to test and adjust the card. Note that calibration should be performed every six months.

SPRINT^{plus} INITIAL SYSTEM CHECKOUT WITH TEST2

Once the system hardware and software are installed, the initial checkout can be done. Executing test will automatically run through the following tests:

* readback of all inputs
* test POD type and relay functions

After these tests are completed, the 'PASS' message will be displayed. The user can then exit the TEST2 utility. If any error messages are displayed, check for I/O address conflicts. If there are no conflicts, and error messages are still being displayed, contact your **SPRINT** distributor.

TEST2 can also be executed in calibration mode. This menu mode instructs the user in all actions to test and adjust the card. Note that calibration should be performed every six months.

SPRINT EXPERT CALIBRATION UTILITY

TESTEX can also be used for detailed test and calibration of the **SPRINT** system. A good voltmeter is required for calibration (commands 2, 3, 5, 9). An 50 MHz oscilloscope is needed for timer and output driver testing (4, 6, 8, 9, E).

To do this, start the program with the command:

>TESTEX -c

The following menu will be displayed:

1 = test all inputs	self test
2 = set Vcc voltage	allows adjustment of Vcc levels
3 = set Vpp voltages	allows adjustment of other levels
4 = pulse all outputs	observe pin driver waveforms
5 = test Vcc drivers	observe Vcc drivers
6 = test delay	verify on-board timer functions
7 = test pod	self test TOP and base functions
8 = test Vcc pulse driver	test pulse drivers
9 = set current limits	allows adjustment
E = test oscillator	observe XTAL emulator waveform
S = test **SPRINT** bus	self test
V = test PIN drivers	self test and Vth display
q = exit to DOS	

All tests contain on-screen messages detailing how the test is used.

Tests 1, 7, S and V are done as part of the automatic test on installation. These tests verify correct system installation. The user can run these tests individually. The message 'no errors detected' in test 1 should appear on the screen. If test 1 is ok, run the test 7 to test the installation of the socket pod and check the results for no errors.

For tests 2, 3 and 9, use a multimeter with pin 20 of the TOP40 dip as reference. For tests 4, 5, 6, and E, use the metal case of the computer as Ground. Never connect an Oscilliscope ground to any socket pin during TEST.

SPRINT^{plus} CALIBRATION UTILITY

TEST2 can also be used for detailed test and calibration of the **SPRINT** system. A good voltmeter is required for calibration (commands 2, 3, 5, 9). An 50 MHz oscilloscope is needed for timer and output driver testing (4, 6, 8, 9, E).

To do this, start the program with the command:

>TEST2 -c

The following menu will be displayed:

1 = readback all inputs	self test
2 = set Vcc voltage	allows adjustment of Vcc levels
3 = set Vpp voltage	allows adjustment of VPP channel 1
4 = pulse all outputs	observe pin driver waveform
5 = test Vcc drivers	observe Vcc drivers
6 = test delay	verify on-board timer function
7 = test pod	verify socket pod functions
8 = test Vcc pulse driver	test pulse drivers
9 = set up Vpp2 voltage	allows adjustment of VPP channel 2 (SN > 800)
q = exit to DOS	

Tests 1 and 7 are done as part of the automatic test on installation. These tests verify correct system installation. The user can run these tests individually. The message 'no errors detected' in test 1 should appear on the screen. If test 1 is okay, run test 7 to test the installation of the socket pod and check the results for no errors. If there are any errors, check first the positions of the filter switches on the socket adapter, then check if any other boards in the system conflict with the address 300-31f Hex used by **SPRINT** as shipped from the factory. If there is an address conflict, the base address of the **SPRINT** board may be changed.

SPRINTplus and SPRINT EXPERT are covered by a 6-month warranty on hardware, and a 6 month warranty on software. For 6 months from the date of delivery, we will repair your SPRINT hardware free of charge.

In addition, within 6 months of the date of purchase, you may update your software to the latest release. This release will typically include new devices and new features. Note that more devices, features and automatic updating are provided by the SMP - Software Maintenance Plan. The Software warranty date is shown on the board (for example SW 12/90), until this date, you will have free access to all new devices added into the SPRINT software. Thereafter, SMP is required for new devices.

HOW TO REQUEST SERVICE OR UPDATES:

a) The following items are required to qualify for free repair of your SPRINT programmer:

1. Fax or write first to explain the type of problem and to receive a Return Authorization Number. Provide complete details of the type of computer used, the device type selected and the error message/problem you have.

2. Provide a copy of shipping documents.

3. After authorization, send your SPRINT back to your local distributor.

b) The following items are required to qualify for a free software update in order to correct a software problem:

1. Write a description of the error and send it to your local SPRINT distributor.

2. Include enough information for us to reproduce your problem (failed devices, and matching Disk files.

3. If the problem directly affects your ability to correctly program devices, and we have enough information to correct the problem, then you will receive a corrected version free of charge.

4. Changes in manufacturers specifications are not considered as errors.

c) The procedure to receive your free update within 6 months of the date of delivery is to request it via mail or fax from your local distributor. Please include a copy of your invoice, the serial number of your SPRINT board, and the present software version you are using at the time.