Document Number: 30722-000-002

PTC-2124

DOS Software

Guide

PTC-2124
DOS Software Software Guide

Document Number: 30722-000-002

Release Date: November 2000

Technical Writer: Norm Walters

Introduction

This Guide's Purpose and Scope

This manual was written by the Symbol Technical Publications Group. This group is tasked with providing technical documentation for the Symbol PTC-2124 product line that uses the Microsoft® Windows CE® Operating System. Every effort has been made to provide accurate and concise information to you, our customer.

The *PTC-2124 DOS Software Guide* provides information that allows the user to set up and use the PTC-2124. This manual is meant to provide information on the various components of this product, including

- General regulations,
- Overview of the PTC-2124,
- Maintenance and troubleshooting,
- Available accessories.

This manual, however, does not provide instructions on how to perform the tasks specific to your job within your organization. For job-specific information, refer to the instructions provided by your organization.

Contacting Symbol's Support Center

Symbol's Support Center may be contacted to obtain help in resolving any PTC-2124 system problem that you may experience.

If you have a problem running your unit or using your equipment, contact your facility's technical or system support. If there is a problem with the equipment, the system support will contact the Symbol Support Center at 1-800-653-5350.

For additional information on Symbol's products and services, please visit our website at **www.symbol.com**.

Table of Contents

PTC-2124 Overview

The PTC-2124 is a rugged AMD™ SC400 processor-powered, pen-based Portable Teletransaction Computer (PTC). It couples standard PC technology with Symbol's expertise in data collection and radio technology to provide a flexible, high-performance portable system.

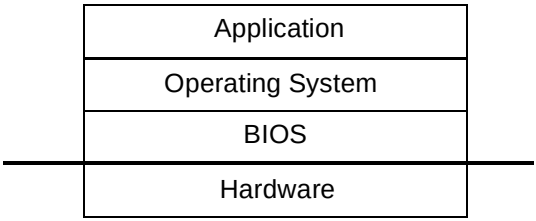
This section of the manual provides a basic overview of the PTC-2124's software environment, including brief discussions of

- the system BIOS,
- the operating system and applications,
- the software kernel,
- the PenRight! operating environment,
- power management, and
- software development kits (SDKs).

Functional Overview

The PTC-2124 uses a PC-type software architecture consisting of three major software layers:

- BIOS (**B**asic **I**nput/**O**utput **S**ystem)
- Operating System
- Application



BIOS

The BIOS is the lowest software layer and typically exists in ROM memory. It provides basic input/output services for the system and an insulating interface between the upper software layers and the hardware layer.

The PTC-2124 uses a customized version of the Phoenix™ BIOS to support PTC-2124 hardware features. To achieve this level of support while maintaining PC compatibility, a system extension called the Symbol BIOS, or TBIOS, is used.

TBIOS is a method for expanding the BIOS functions without reserving a function number that could be used by another application. This is accomplished by forcing the TBIOS code to reside with standard BIOS code, but by providing a different entry point to the TBIOS. The entry point is defined by the contents of a static location in the BIOS Compatibility Area (COMPAS).

Operating System

The operating system provides additional higher-level services to the Application Layer and uses the BIOS interface for performing hardware functions. By using standardized BIOS interfaces, the PC architecture supports standard operating systems. The PTC-2124 supports the MS-DOS 6.22 operating system.

Application

The application is the highest software layer and provides controlling software programs for the unit.

Other Software Components

Software Kernel

The software kernel is a simple program loader that is used to reprogram the PTC-2124 Flash PROM memory areas.

PenRight! Operating Environment

PenRight! is a graphical application development tool used to create forms containing objects such as buttons, list boxes, and writing fields for pen-based computers. These objects are activated with a stylus pen, which may be used to select options, highlight items, enter strings of characters, or enter a signature on a digitized screen.

PenRight! applications are DOS programs that are driven by events. Each depression or release of the stylus on the digitized screen triggers a particular type of programmed action.

Power Management

Power management is a necessity for battery-operated devices such as the PTC-2124. The unit is designed to incorporate battery-saving power management functions directly into the unit hardware and software architecture. The platform hardware incorporates features that allow most internal elements to be slowed down or deactivated. Together, the hardware and software constantly monitor system activity and implement power management activity states that are used for power management functions.

Software Development Kits

Symbol provides application development tools in the form of Software Development Kits (SDKs) to help developers design application programs for the SC400 product family. The SDK is a collection of run-time libraries and TSRs that assist in the development of C application programs on PC-compatible computers. Once an application program is developed, it may be copied to a PCMCIA card and inserted into the PTC-2124 to load or run the application.

Note: *If converting applications from a PTC-1124, any portion of the application software that directly manipulates the SLC (PTC-1124) hardware will have to be rewritten before it will work on the PTC-2124. The internal architecture of the PTC-2124 is not compatible with SLC. Any software (BIOS, drivers, or applications) that manipulates hardware must be written specifically for the PTC-2124.*

PTC-2124 Unit Configurations

The PTC-2124 DOS unit is available in a wide variety of configurations.

This section will cover the configuration options for the following devices:

- Radio,
- RAM,
- ROM,
- Storage,
- PCMCIA Slots,
- Displays,
- IrDA,
- Ethernet, and
- DCD Devices.

Radio Options

Batch

A batch unit does not have an antenna cable integrated in the unit. It is still possible to install a radio in the unit by way of the user accessible PCMCIA slots. However, an antenna must be installed on the radio. Either of the PCMCIA slots may be used for a radio card.

LAN Radio Ready

The radio ready unit has an RSMA antenna cable integrated into the unit. The end of the cable routes through the unit to the externally accessible PCMCIA slot door. A radio may be installed into PCMCIA Slot 0 (closest to the display) or in PCMCIA Slot 1 (slot farthest from the unit's display).

WAN Radio Factory Installed

When ordered with a WAN radio such as a DataTac or Mobitex network radio, the radio module is installed internally in the unit. Because these radio types are not PCMCIA form factor, they require additional space in the unit. The unit therefore has a deeper backshell than non-WAN-radio-equipped units.

Memory

RAM

The PTC-2124 has RAM configurations of 4, 20, 36, or 64 MB on the CPU board. When the 16- and 32-MB memory modules are added, the 4 MB is included. The 64-MB configuration actually contains 68 MB of RAM, but the SC400 is able to address only 64 MB and the 4 MB on the CPU board is disabled.

ROM

The PTC-2124 comes with 512 KB of ROM, which contains the BIOS for DOS.

Storage Options

ATA

The PTC-2124 supports ATA drives, both solid state and rotating. ATA Type II cards may be used in either of the user accessible PCMCIA slots. Symbol offers factory installed card options of 60, 110, and 175 MB.

The 520-MB rotating ATA disks (any Type III) are installed in PCMCIA Slot 0.

SRAM

The PTC-2124 also supports SRAM (static RAM) cards. Generally, these cards are used for special purposes like reloading the ROM images (Genesis procedure) or booting the unit, rather than data storage because the capacities are rather low — 1, 2, and 4 MB. However, these cards may also be used by the application for data storage.

Compact Flash

The PTC-2124 has an internal Compact Flash adapter which is a standard 50-pin port that supports a variety of Compact Flash module sizes.

PCMCIA Slots

The PTC-2124 has two PCMCIA slots that are user accessible. These external slots are controlled by an Intel PCMCIA controller.

External Slots

Slot 0

This is the card slot closest to the display.

Slot 1

This is the card slot farthest from the display.

Display

The PTC-2124 is designed with a 4.7" (11.9 cm) diagonal ¼ VGA screen offering 12 lines x 40 characters of text in DOS mode. The standard monochrome transflective LCD display offers 320 x 240 pixel resolution, 64 levels of gray, and an EL backlight. A scratch- and impact-resistant touch digitizer protects the display.

The PTC-2124 includes a transparent, resistive-touch digitizer mounted above the LCD display. The digitizer is designed to support both stylus and finger activation and incorporates a minimal amount of "palm rejection" to help minimize hand contact. It also uses transparent material and will not interfere with the LCD display.

IrDA

The PTC-2124 supports an IrDA 1.0 port located on the right side of the unit that provides an optical serial communication interface. The IrDA communication port uses an infrared (IR) light beam to support a half-duplex, point-to-point communication link with a peripheral device.

Ethernet

The PTC-2124 supports Ethernet communications using a full 16-bit controller to provide a high-speed Ethernet interface via the cradle contacts for transmit and receive operation at 10 Mbps.

DCD Devices

Auto ID devices are attached to the 30-pin Scanner/Expansion module at the top of the unit.

Scanners

The PTC-2124 supports an optional laser scanner module. Using an attached laser scanner, the unit may be programmed to recognize, read, and discriminate automatically among six 1D barcode types. For instructions on using a laser scanner module, refer to the ***PTC-2124 User's Guide***.

PTC-2124 Assignments

This section provides information on the following PTC-2124 components:

- Resource Map,
- Hardware Interrupts, and
- COM Port Assignments.

Resource Map

PTC-2124 components are assigned various address ranges within system memory.

Memory Allocation Table

PTC-2124 memory is allocated as follows:

Address	Size	Resource	Comments
000000-09FFFF$_h$	640 KB	Base	Base Memory
0A0000-0BFFFF$_h$	128 KB	Video	Video Display Frame Buffer
0C0000-0C7FFF$_h$	32 KB	Video	VGA BIOS ROM
0C8000-0C9FFF$_h$	8 KB	ASIC	Shared RAM area for ASIC
0CA000-0CBFFF$_h$	8 KB	--	Unused
0CC000-0CCFFF$_h$	8 KB	PCMCIA/UMB	PCMCIA Window or Upper Memory Block
0CD000-0CDFFF$_h$	8 KB	PCMCIA/UMB	PCMCIA Window or Upper Memory Block
0CE000-0CEFFF$_h$	8 KB	PCMCIA/UMB	PCMCIA Window or Upper Memory Block
0CF000-0CFFFF$_h$	8 KB	PCMCIA/UMB	PCMCIA Window or Upper Memory Block
0D0000-0DFFFF$_h$	64 KB	UMB	Upper Memory Block
0E0000-0EFFFF$_h$	64 KB	UMB	Upper Memory Block (reserved)
0F0000-0FFFFF$_h$	64 KB	BIOS	System BIOS (shadowed)
100000-xxxxxxxx$_h$	>1 MB	EMS	Extended Memory

Hardware Interrupts

Knowledge of hardware interrupt assignments will be useful for making system configuration decisions.

Interrupt Table

PTC-2124 hardware is assigned to the following interrupts (IRQs):

H/W Interrupt	PC/AT Usage	PTC-2124 Usage	Notes
IRQ 0	Timer 0	Timer 0	
IRQ 1	Keyboard	Keyboard	
IRQ 2	IRQ(15:9) Cascade	IRQ(15:9) Cascade	
IRQ 3	COM 2	COM 2/4	COM 2 = User Port COM 4 = WAN Radio Serial Port
IRQ 4	COM 1	COM 1	IrDA Port I/F
IRQ 5	LPT 2		
IRQ 6	Floppy Disk		
IRQ 7	LPT 1	LPT 1	
IRQ 8	RTC	RTC Alarm	
IRQ 9		PCMCIA	Open for PCMCIA devices
IRQ 10		Ethernet	
IRQ 11		PCMCIA	Open for PCMCIA devices
IRQ 12	Mouse	Digitizer	
IRQ 13	Math Exception	ASIC	
IRQ 14	Hard Disk	IDE ATA SS Hard Disk	
IRQ 15		PCMCIA	Open for PCMCIA devices

COM Port Assignments

The following are the default COM Port settings for the PTC-2124:

- COM1: IrDA,
- COM2: 15 pin serial or cradle serial ports,
- COM3: Scanner, and
- COM4: WAN radio.

Boot Options

The PTC-2124 uses a customized version of the Phoenix BIOS (**B**asic **I**nput/**O**utput **S**ystem) that supports PTC-2124 features. Because the BIOS interfaces with the hardware, it must be aware of the physical devices present in the system. This section provides information on the following:

- The behavior of the BIOS at boot-up,
- Changing the BIOS settings,
- Changing the boot source,
- Drive letter mapping,
- Reflashing the BIOS, and
- Resetting/Rebooting the PTC-2124.

POST

In PC-based systems, after a boot, the BIOS is responsible for testing and initializing all hardware components and boot loading the operating system into memory. This process is known as the Power-On Self Test, or POST.

What Happens During POST

During the POST process, messages must be generated for the system. The POST, initializations, and messages are customized for the PTC-2124.

Once the POST operations are complete, the BIOS loads the operating system. Once a bootable disk is found, the boot loader is loaded into memory and executed. If no bootable disk is found, the system displays the message: **No Boot Device Available**.

Changing BIOS Settings

There are many settings in the BIOS of the PTC-2124 that you would expect to see in a desktop computer system. Once set, the BIOS settings generally do not have to be returned to the previous settings.

To access the BIOS settings, attach a physical keyboard to the PTC-2124. During the memory test portion of the POST process, press **F2**. Upon completion of the memory test, the system will grant access to the BIOS setup menu.

Listed below are several of the settings that can be accessed from the BIOS setup screen.

Power Management

There are four operational states for the PTC-2124: Ship, Full Run, Standby, and Suspend. For more Power Management information, see "Power Management States" on page 31.

Boot Source

This determines which drive is used to boot the system.

CPU Speed

There are three possible settings: 33, 66, and 100 MHz. The default is 100 MHz.

> ***Note:*** *The system's contrast and brightness controls become disabled when in the BIOS setup mode.*

Boot Sources and Drive Letter Mapping

Normal Boot

Under normal conditions, the unit boots from the internal Compact Flash card and loads Card and Socket Services to support SRAM and ATA cards.

Drive Letter Assignments when Booting from a Compact Flash Card

Installation Slot	Drive Letter
Compact Flash	C
0	D
1	E

Note: *If Card and Socket Services are not loaded, the unit will not have access to SRAM or ATA cards.*

Booting from an ATA Card

To prepare an ATA Card for booting the system, perform the following steps:

1. The ATA Card must be formatted first. The **HDFMT** utility located in the **/DOS** directory on the unit may be used for this. This command is identical to the MS-DOS FORMAT.COM utility. For more information, refer to the section titled "Formatting an ATA Card" on page 127.

2. Copy the MS-DOS operating system files onto the unit. This may be done by running the **MS-DOS SYS.COM** utility.

To boot from an ATA Card, perform the following steps:

1. Remove the Compact Flash card from the CF slot inside of the unit.

2. Install the ATA card into PCMCIA Slot 0 (closest to display).

3. Restart the unit.

4. Press **F2** during POST to enter CMOS Setup.

5. Arrow down to access the **Embedded Features** screen.

6. Ensure that the **PCMCIA ATA** option is enabled.

7. Reboot the system.

Note: *Do not attempt to load Card and Socket Services when booting from an ATA card.*

Drive Letter Assignments when Booting from an ATA Card

Installation Slot	Drive Letter
Compact Flash	Not Installed [1]
0	C
1	Not Supported [2]

[1] CF Card must be removed.

[2] Card and Socket Services cannot be loaded. Doing so will cause the unit to lock up.

Booting from an SRAM Card

The PTC-2124 may be configured to boot from an SRAM card instead of its internal Compact Flash Card. This is similar to booting a desktop PC from a floppy disk.

> ***Note:*** *When an SRAM Card is used as the boot source, it must remain in the PCMCIA slot. Do not attempt to remove it.*

For an SRAM Card to be recognized properly by the unit, it should be a 1-, 2-, or 4-MB SRAM Card with No-Attribute Memory. SRAM Cards with Attribute Memory may not be recognized properly by the unit at boot up.

> ***Note:*** *Do not attempt to load Card and Socket Services when booting from an SRAM card.*

To prepare an SRAM Card for booting purposes, perform the following steps:

1. The SRAM Card must first be formatted properly. To do this, the **PCFORMAT** utility in the PCM directory may be used. For more information on **PCFORMAT**, refer to the section titled "Formatting an SRAM Card" on page 127.

2. Copy the **MS-DOS** operating system files onto the unit. This may be done by running the **MS-DOS SYS.COM** utility.

To boot from an SRAM Card, perform the following steps:

1. Insert the SRAM Card into Slot 0 (closest to the display).

2. Restart the unit.

3. Press **F2** during POST to enter CMOS Setup.

4. Arrow down to access the **Embedded Features** screen.

5. Ensure that the **[ROM/RAM Disk 0]** option is set to **[PCMCIA]**.

6. Reboot the system.

Drive Letter Assignments when Booting from an SRAM Card

Installation Slot	Drive Letter
Compact Flash	C
0	A or B
1	Not Supported[1]

[1] Card and Socket Services cannot be loaded. Doing so will cause the unit to lock up.

Note: There is a possibility of a boot error if all video modes are not calibrated. As the PTC-2124 boots up, tapping on the screen as the **config.sys**, **autoexec.bat**, **user_boot.bat**, or application is loading may produce one of the following errors:

- Run-time error R6002 — integer divide by 0.
- Divide overflow — memory allocation error.
- Divide overflow.

Should one of these errors occur, it may be corrected by performing the following steps:

1. Delete **TMOUSE.INI** in the mouse directory.
2. Reboot the unit and allow it to go into the calibration mode.
3. Calibrate all video modes.
4. **Save** and **Exit**.
5. Reboot the unit.

Reflashing the BIOS

TFLASH Utility

The TFlash utility is a DOS command line program that will reflash a system BIOS image into the unit. To use this program:

Step	Action
1	Copy the binary ROM image of the BIOS to be flashed to the directory in which the **tflash.exe** resides.
2	Boot the unit into DOS protected mode (perform a "clean" boot with **NO MEMORY MANAGER LOADED**). This may be done by using a keyboard and pressing **F5** or by renaming the **config.sys** and **autoexec.bat** files (i.e., **config.old**, **autoexec.old**, etc.).
3	Type **TFlash <name>** (<name>=new BIOS ROM Image name)
4	Press **ENTER**.

When the flash process is complete, restart the unit. If the unit displays a blank screen without restarting, or if the restart message is not displayed, the unit should be reset by pressing the **Reset** button and then the **Resume** button. The unit is now operable with the new BIOS image.

Note: *No "crisis mode" is associated with this utility. If a failure occurs during the flash process, the process must be repeated in its entirety.*

The TFLASH utility can also read the BIOS image from ROM and write it to disk. The command line option for this is:

Tflash romimage.bin –r8000
-r8000 is the memory location for the bios. The resulting file size should be 256 KB.

Complete usage for TFLASH is:

ex1: TFlash <Filename>[-o<offset>]
ex2: TFlash <Filename>[-r<length>] [-o<offset>]
ex3: TFlash [-e<length>] [-o<offset>]

Where:

-o <offset>, default=0, from base of ROM.
-r Reads <length> bytes, default=512k, from ROM into a file.
-e Erases <length> bytes, default=512k, ROM area to all FFs.

Automatic Genesis Flash

Perform the following procedure to re-flash the BIOS of a PTC-2124. Note that this procedure may also be used to force the PTC-2124 to boot from a bootable SRAM card in PCMCIA Slot 0.

To perform an Automatic Genesis Flash, follow these steps:

Step	Action
1	Insert the Genesis formatted SRAM card into Slot 0.
2	Turn on the PTC-2124.
3	Press and hold the **Backlight** button.
4	Press and hold the **Resume** button.
5	Release the **Backlight** button.
6	Press and release the **Backlight** button.
7	Release the **Resume** button.
8	The Status light will start to flash and the backlight will turn on. The unit will then beep about 15 – 20 times to indicate that the BIOS is being reloaded.
9	After the unit stops beeping, remove the SRAM card from Slot 0.
10	Perform a Warm Boot on the unit.

Resetting/Rebooting the PTC-2124

There are several ways to reset the PTC-2124. It should be noted that removing the battery will not reset the PTC-2124. The bridge battery maintains all pointers and memory such that when the battery is reapplied to the unit, the unit will resume operations at the point at which the power was removed.

Cold Reboot

Use the following procedure to perform a Cold Reboot. It powers off the PTC-2124 and puts it into ship mode. Turning on the unit at this point (pressing the **Resume** button) will reinitialize the PTC-2124.

Step	Action
1	Remove the battery.
2	Locate the metallic **Reset** button located underneath the battery.
3	Use a thin, insulated device (such as a plastic coffee stirrer) to press the **Reset** button.
4	Replace the battery.
5	Press the **Resume** button.

Console Reboot

The following Console Reboot procedure stops the PTC, resets it, then restarts (boots) it. This procedure erases all programs and data stored in RAM. When the PTC restarts, it returns to the operating system.

Step	Action
1	Turn on the PTC-2124.
2	Press and hold the **Contrast** button.
3	Press and hold the **Resume** button.

Step	Action
4	Release the **Contrast** button.
5	Press and release the **Contrast** button.
6	Release the **Resume** button.

Power Management

APM

As battery conservation measures are dependent upon user operations and preferences, the PTC-2124 allows the user application program to direct and control Power Management operations via the industry standard Advanced Power Management (APM) software interface. This interface is defined by the APM Interface Specification which is currently at Revision Level 1.2. The platform software uses the APM functions and states to place the PTC-2124 into the appropriate platform Power Management Activity State.

To support the APM interface in the PTC-2124, Symbol enhanced the APM BIOS functions supported in the Phoenix BIOS from APM revision 1.0 to 1.2. The APM BIOS functions manage power in the background on the basis of device activity, and are specific to the PTC-2124 hardware platform. The APM BIOS is the software interface to the PTC-2124 platform and its power managed devices and components. This interface allows software applications to take an active part in managing the power consumed by the PTC-2124 unit.

Monitoring

When full power management is enabled, system activity is monitored at two different levels. The APM Driver software monitors the frequency of certain interrupts, and the APM BIOS monitors the hardware activity directly. Both work together to inform APM-aware applications, Device Drivers, and TSRs about power management events and conditions. The PTC-2124 allows all elements to work together to conserve power.

APM Driver

POWER.EXE

For MS-DOS environments, Symbol provides an APM Driver module, **power.exe**, which connects to the APM BIOS and controls power management policy via function calls to the APM software interface. This DOS driver was modified by Symbol to provide a specific interface to the PTC-2124 pen-based computer. **Power.exe** is a terminate and stay resident (TSR) driver that must be loaded into the **config.sys** file.

Command Line Parameters

The following is a description of the **power.exe** command line parameters. The current settings are always displayed when invoking **power.exe** from the command line.

power [/option]

Options: **/SUSPEND XXXX**

This parameter sets the amount of time that the unit will be in the **Standby** state before entering **Suspend**. There are only 8 valid settings for this parameter. They are 0 (disable), 64, 128, 256, 512, 1024, 2048, and 4096 seconds. Any value of 1 – 4096 entered will be rounded off to the nearest valid value. A parameter value of 4097 or higher will cause an **Invalid Parameter** error. The default setting for this parameter is 16 minutes.

/STANDBY XXXX

This parameter sets the amount of time during which the unit will be in the **High Speed** state before entering **Standby**. There are only 8 valid settings for this parameter. They are 0 (disable), 8, 16, 32, 64, 256, 512, and 1024. Any value of 1 – 1024 will be rounded off to the nearest valid value. A parameter value of 1025 or higher will cause an **Invalid Parameter** error. When **Standby** is disabled, the **Suspend** timer never starts. Therefore, disabling **Standby** effectively disables **Suspend**. The default setting for this parameter is 32 seconds.

/ADV:[MIN:MAX:REG]

This parameter means that **power.exe** has enabled and connected to the APM BIOS. It monitors hardware and applications, then reduces power consumption where possible. The MIN, MAX, and REG parameters are optional. MIN conserves the least power, MAX conserves the most power, and REG gives average power conservation. The default setting for this parameter is REG.

/STD

This parameter means that **power.exe** will enable the APM BIOS, but it will disconnect. Therefore, **power.exe** will not poll for APM events.

/OFF

This parameter means that **power.exe** disables the APM power management.

> ***Note:*** *A help message may be called by entering **POWER /?**.*

Power Management States

There are four operational states for the PTC-2124: **Ship**, **Full Run**, **Standby**, and **Suspend**.

State	Description
Ship	The unit is completely powered off. The resume key must be pressed to power-up the unit. Pressing the metallic Reset Button located underneath the battery puts the unit into this mode.
Full Run	The system CPU and devices are fully awake and functional — also referred to as being fully operational.
Standby	A state generated by software-controlled timers and lack of activity on monitored buses. While in the **Standby** state, the unit's power usage is reduced by up to 50%, and the unit is able to process many functions at a reduced rate (8 MHz).
Suspend	A state in which the system further reduces the power consumption from **Standby** mode. There are several suspend options that are controlled by the BIOS or device drivers. Most of the state changes that occur to get into **Suspend** are due to lack of activity of devices. While in the **Suspend** state, the unit's power usage is reduced by up to 90%; however, the unit is functionally deactivated until it is reawakened.

The PTC-2124 allows the user to help conserve power directly via the Suspend/Resume feature. This feature allows the user to place the unit into **Suspend** state by depressing the **Resume** switch on the unit's top cabinet. This deactivates the unit and most internal hardware elements. The user may reactivate the unit by pressing the **Resume** switch again.

Additionally, the PTC-2124 provides a Standby Timer, which specifies the inactive time before automatically entering the **Standby** state and deactivating most internal elements. The system default setting for this timer is approximately 32 seconds. During this period, the unit may be reactivated via the Digitizer or the Resume switch. Once the unit enters **Suspend** state, the Digitizer will no longer wake the unit.

Magic Packet Mode

Another feature of the PTC-2124's power management is the ability to enable the Magic Packet Wake-Up mode (Magic Packet is an E-Wake utility developed by AMD) when the unit is docked in a cradle and in the Suspend state. Magic Packet mode provides the ability to wake up the unit remotely by using the Ethernet connection established through the cradle. This allows the unit to be accessed remotely and managed even if the **Suspend** state has been entered. The Magic Packet feature does not require that a software network driver be loaded in the unit.

When the unit is put manually into the **Suspend** state (while docked in a cradle), the Ethernet controller will automatically enable Magic Packet mode. While in the **Magic Packet** mode, the unit will monitor all incoming frames to determine whether any of them is a **Magic Packet** frame. A Magic Packet frame is a unit of data that is sent by a network manager, via Ethernet connection, from a remote site with the intent to wake up the unit. When a Magic Packet frame is received and detected, the Ethernet controller will wake up the unit and disable the **Magic Packet** mode. The unit then returns to **Full Run** state, regaining full functionality, including network accessibility.

Features of Power Management in the Standby State

While in Standby, the PTC-2124 has the following features:

- Power usage is reduced (up to 50% savings).
- Processor speed drops to 8 MHz.
- Display is turned off and all other devices remain on.

The following conditions will prevent the unit from entering the **Standby** mode:

- IrDA drivers loaded.
- Cradle Ethernet link with CradTSR loaded.
- Scrnblnk utility loaded.

The following table illustrates wake sources used to bring the PTC-2124 out of **Standby** mode:

Wake Source	Comment
Touch/Digitizer	Unit wakes when the digitizer registers a touch.
RTC Alarm	Unit wakes when a Real Time Comparator Alarm is initiated.
Dock/Undock	Unit wakes when it is connected to or disconnected from a cradle.
Keyboard	Unit wakes with input from a keyboard that is attached to a cradle.
Resume Button	Unit wakes when the Resume Button is pressed.

Features of Power Management in the Suspend State

While in the Suspend mode, the PTC-2124 takes on the following conditions:

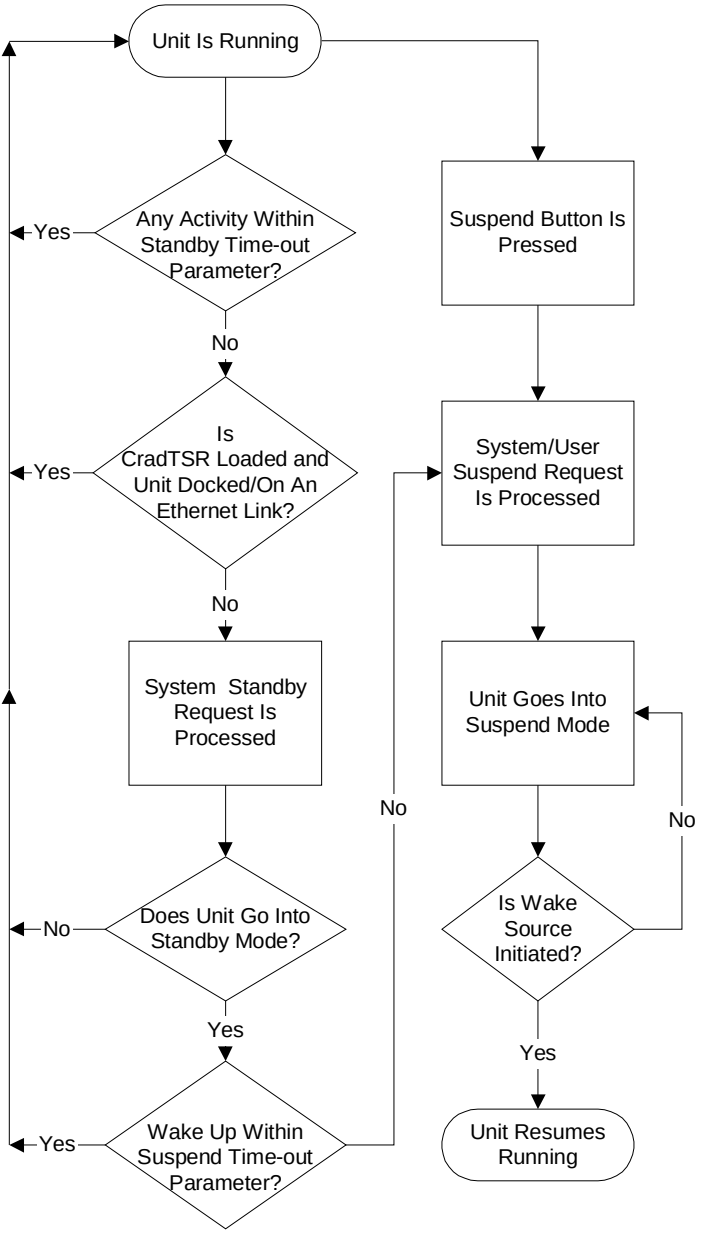
- Power usage is reduced (up to 90% savings).
- Processor speed drops to 8 MHz.
- All devices are turned off, except for non-PCMCIA-based WAN radios.
- Ethernet is in Low Power.

The following conditions will prevent the unit from entering the **Suspend** mode:

- IrDA drivers loaded.
- LAN radio and drivers loaded.
- Cradle Ethernet link with CradTSR loaded.
- Scrnblnk utility loaded.
- The following table illustrates wake sources used to bring the PTC-2124 out of the Suspend mode:

Wake Source	Comment
User Serial Port	Unit wakes in response to a Ring Indicator.
Cradle Serial Port	Unit wakes in response to a Ring Indicator.
WAN	Unit wakes in response to a Ring Indicator.
Ethernet	Unit wakes in response to E-Wake.
Dock/ Undock	Unit wakes when it is connected to or disconnected from a cradle.
Resume Button	Unit wakes when the Resume Button is pressed.

SC 400 Power Control Flow



Driver Support

The PTC-2124 provides support for Symbol drivers. Supported Symbol drivers are described in the following sections.

Cradle Information

Cradle Overview

The Symbol PTC-1124 Desktop/Vehicle Cradle is a specialized docking station that is used for the PTC-2124. The cradle provides the following services:

- External Serial Port Connection,
- Battery Recharging Connection,
- External Ethernet Connection, and an
- External Keyboard Connection.

The SC-1124 is designed for use on a flat horizontal surface, such as a table or desk. The VC-1124 mounts securely inside a vehicle's cab (using a vehicle mount).

Each cradle holds one PTC and one spare battery pack at a time and works with the PTC in two ways:

1. It acts as a communication link. Through the cradle, the PTC can send data to and receive data from a host computer or other serial devices.
2. The cradle provides power for rapidly recharging the PTC's lithium-ion battery pack and a spare battery pack when the PTC and the spare pack are installed in the cradle.

The cradle may be connected via cable to a network through its Ethernet port or to external serial devices via its three 9-pin RS-232 serial ports. In addition, a keyboard can be connected to the cradle for use with the installed PTC.

PTC-2124 Cradle Interaction

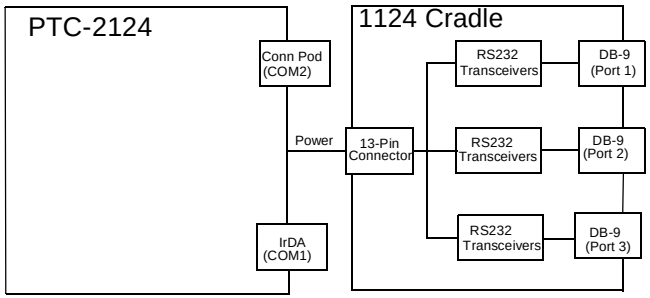
The PTC-2124 has only one COM port available for use by peripherals. The PTC-2124 uses an electrically controlled serial switch-box with a 9-wire interface to provide additional serial ports for this single COM port.

The PTC-2124 provides IrDA on COM1 and wired serial on COM2. An infrared sensor on the PTC-2124 provides the IrDA outlet. A Connector Pod on the PTC-2124 provides the serial outlet.

The following table summarizes the possibilities for a PTC-2124. When out of the cradle, COM1 is available as IrDA and COM2 is available through an attached Connector Pod. When docked, COM1 is available as IrDA and COM2 is rerouted to one of the three DB-9 serial connectors.

Cradle	COM1 Outlet	COM2 Outlet
Undocked	IrDA on PTC-2124	Connector Pod on PTC-2124
Docked	IrDA on PTC-2124	DB-9 on cradle

PTC-2124 Plugged Into A Cradle



Cradle Serial Interface

The cradle serial interface is implemented using a communication chip and cradle electronics to create four multiplexed RS232 serial ports: one internal port and three external ports. Since the serial ports are multiplexed, only one port may be active at a time.

Serial Port 1	DB9 Connector (Male Pins).
Serial Port 2	DB9 Connector (Male Pins).
Serial Port 3	DB9 Connector (Male Pins).
Control Port	No Connector, Internal to unit.

The same COM 2 Port is used for both the Vehicle/ Desktop-1124 Cradle's Serial Connectors and the serial port in the PTC-2124 unit's Connector Pod. Therefore, only one COM interface may be used. When the PTC-2124 is placed in a cradle, the cradle's serial ports can be used if the unit's application program uses the cradle serial routines from the PTC-2124 SDK (refer to the PTC-2124 SDK for software details).

> ***Note:*** *When the connector Pod is connected, the Serial connection is a 4-wire implementation only. When attached to a cradle, the active port is a full 9-wire serial port implementation.*

The Optical Serial Signals use the cradle contacts on the PTC-2124 unit as shown below:

Pin	Signals	Description
1	OTXD	Optical Transmit Data
2	ORXD	Optical Receive Data
3	OTXS#	Optical Transmit Status
4	ORXS#	Optical Receive Status

In the Optical Serial Interface, two lines are used to receive and transmit data and two lines are used as status lines. The two status lines (OTXS#, ORXS#) provide communication status information via Time-Division-Multiplexing. In this technique, the status information is embedded in the status data of each line.

The Status Word, like a normal Data Word, begins with a Start Bit and ends with a Stop bit. However, the Status Word uses three Stop bits to provide an easy identification of the end of the word.

The Status Lines operate at a 38.4 KBPS data rate, where as the Data Lines operate at the rate programmed for the internal communication chip UART controlling the Serial Port. The Serial communication chip UART can be programmed to support the following standard communications functions:

- Data Rate 75 to 115.2 Kbps.
- Parity Even, Odd, None.
- Data Width 5, 6, 7, or 8 bits.
- Stop Bits 1 or 2.

DTR and RTS Latching

The Vehicle/Desktop-1124 Cradle provides DTR and RTS latching on Serial Port 3 to prevent the connected device from dropping the communication link with the PTC-2124. Prior to switching to another Serial Port, the cradle will latch the DTR and RTS lines at their current levels. For example, if the signal is currently high, the line will be latched high, or if the signal is currently low, the line will be latched low. This feature is particularly useful when connecting to devices such as WAN radios.

Ring Indicator

To allow the Ring Indicator (RI) signal to reach the PTC-2124 unit from any serial port (active or not), the RI signals from all Serial Ports are logically Ored together. The RI signal can be used to wake the PTC-2124 unit from the **Standby** or **Suspend** modes.

DB-9 Serial Connector Pinout

The pinout of the RS232 port DB-9 (male pins) connectors are as shown in the tables below:

Pin	Signal
1	CD
2	RXD
3	TXD
4	DTR
5	GND

Pin	Signal
6	DSR
7	RTS
8	CTS
9	RI

Ethernet Port

The 10 Mbps Ethernet Port is available via an RJ-45 connector. The Ethernet Port is located on the right side of the cradle and interfaces to the Cradle Contacts. The cradle RJ-45 port will be wired as DCE accepting a standard male LAN cable. The Ethernet RJ-45 connector pinout is as follows:

PIN	SIGNAL
1	TXD+
2	TXD-
3	RXD+
4	NC

PIN	SIGNAL
5	NC
6	RXD-
7	NC
8	NC

Note: *The Ethernet Port is located on both the Desktop and the Vehicle configurations; however, the port will typically be used only in a Desktop configuration.*

Keyboard Port

The Vehicle/Desktop-1124 Cradle supports an external PS/2 Keyboard Port which is located on the right side of the unit. The PS/2 Keyboard Port supports the following connector pinout:

Pin	SIGNAL	DESCRIPTION
1	KB_DATA	Keyboard Data
2	N/C	Not Connected
3	GND	Ground
4	5VSW	Keyboard Power (+5 VDC)
5	KB_CLOCK	Keyboard Clock
6	N/C	Not Connected

Note: *Permanent Keyboard damage may occur if the user connects or disconnects the keyboard from the cradle's keyboard connector while the unit is docked, unless the unit is suspended first.*

Crad TSR Overview

CradTSR is a Terminate and Stay Resident (TSR) program which controls how the PTC-2124 operates while in a cradle.

Specifically, CradTSR does the following:

- Interacts with the unit's on-board Ethernet controller to enable the controller when the unit is in the cradle and disables the controller when the unit is out of the cradle.
- Rejects APM system **Standby** and **Suspend** requests if Ethernet link activity is detected.
- Reroutes COM2 from the DB-15 on the bottom of the unit to one of the two serial ports on the cradle.
- Configures the baud rate, parity, and stop bits on the three serial ports on the cradle.
- Provides cradle status information to user applications via API calls.

This section discusses in general what happens when **CradTSR** is invoked, both for the first time and at subsequent times.

For more information on Ethernet functions within **CradTSR**, see the section titled "Cradle TSR Function" on page 52.

Invoking CradTSR

When **CradTSR** is invoked for the first time after a reboot, it queries the Symbol BIOS extensions to determine the model of PTC on which it is running. If it finds any model other than the one it is designed for, it exits.

Specifying Interrupt Vector

There is no need to specify the software interrupt vector used for communicating with the TSR. **CradTSR** has been assigned vector 0 x 63 and uses that by default. Optionally, one can override this value using the /V option. Additionally, one can specify the /S option, which will cause **CradTSR** to automatically locate a free vector by scanning the IDT between 0 x 060 and 0 x 06F.

Binary Interface of CradTSR

When **CradTSR** has already been loaded and is resident and **CradTSR** is invoked again, the command line options may be used as a binary interface to the TSR. Many of the **CradTSR** SDK functions are exposed via the binary interface, including options for changing COM2 configuration options, querying current status, etc.

When **CradTSR** is invoked for the N^{th} time, where $N \neq 1$, it searches the software interrupt vector range looking for a preloaded version of **CradTSR**. When it finds one, it uses the **CradTSR** SDK interface to query its major and minor version numbers and reports them to the user.

The arguments /C2P, /C2C1, etc. may be used to alter the current settings of COM1 or COM2. If specified when the PTC-2124 is outside of the cradle, the change takes place immediately. If the PTC-2124 is in the cradle, the COM1 parameters are not immediately altered.

Besides the Device parameters, the /M argument may be used to monitor the in-cradle state, and the /Q argument will dump all of the Device settings and other state variables.

Command Line Arguments

This section describes specific command-line arguments available when invoking **CradTSR**.

When the **CradTSR** is invoked, the syntax is

Cradtsr [options].

There are more options available for the N^{th} invocation ($N \neq 1$) than for the 1st. In the table on the next page, options available only for later invocations are marked with the phrase “if preloaded” (meaning valid only if **CradTSR** is already loaded).

In this table, **BR** stands for a decimal baud rate between 300 and 115200. **D** stands for the number of data bits (5 – 8), and **S** stands for the number of stop bits (1 or 2). **P** stands for a parity code: *E* (even), *O* (odd), and *N* (none) are supported.

The “**-u**” option for /C2P, /C2C1, etc. tells **CradTSR** to stop altering the system state to support the given configuration.

The table below provides a description of each option and specifies when each option is available.

Option	Available:	Description
/V:hexVec	If not loaded	Load the TSR at hex vector *hexVec* if available.
/?	Always	Print summary of command-line options.
/C2C1:BR,D-P-S	Always	Setup of COM2 on Cradle DB1 (when in cradle).
/C2C2:BR,D-P-S	Always	Setup of COM2 on Cradle DB2 (when in cradle).
/C2C3:BR,D-P-S	Always	Setup of COM2 on Cradle DB3 (when in cradle).
/DB1	Always[1]	Route COM2 to DB1 when in cradle.
/DB2	Always[1]	Route COM2 to DB2 when in cradle.
/DB3	Always[1]	Route COM2 to DB3 when in cradle.
/M	If pre-loaded	Monitor in-cradle status.
/Q	If pre-loaded	Query current configuration.
/U	If pre-loaded	Unload the TSR (if possible).
/C2C1: -u	If pre-loaded	Disable COM2 on Cradle DB1.
/C2C2: -u	If pre-loaded	Disable COM2 on Cradle DB2.
/C2C3: -u	If pre-loaded	Disable COM2 on Cradle DB3.
/C2C1: +u	If pre-loaded	Enable COM2 on Cradle DB1.
/C2C2: +u	If pre-loaded	Enable COM2 on Cradle DB2.
/C2C3: +u	If pre-loaded	Enable COM3 on Cradle DB3.

[1] Cradle port must first be enabled with one of the [/C2Cx: +u] enable options.

Purpose

The purpose of the TSR is to promote longer device life by turning off the video driver and backlight after a determined period of idle time has elapsed. Idle time is based on monitored events.

SCRNBLNK is useful in situations in which the unit is not allowed to go into system **Standby**/**Request** states (such as when there is an active Ethernet link on the cradle, or when a radio is installed in the unit).

The goals of the TSR are as follows:

- Prolong the useful battery time,
- Automatically resume processing upon events, and
- Allow the user to easily select the time-out period.

User Interface Functions

The user interface is a standard DOS command line with options to reject **Standby** messages, specify the time-out period in seconds, unload the TSR, and use a selected vector:

scrnblnk /r /t[seconds] /u /v[vector].

Reject Standby (/r option)

The user may choose to reject **Standby** messages. This will prevent the unit from moving to **Suspend** from a power timer.

Time-out (/t option)

The user may change the time-out period by reentering the command line with a different time. The time-out period may be set to a value within the range of 1 – 3600 (seconds).

The user may disable the TSR by entering **/t0** seconds for a time-out period.

Unload (/u option)

The user may unload the TSR and recover the memory if no vectors have been changed since the TSR was installed.

Vector (/v option)

The user may choose a vector in the range 60_h through $6f_h$ for the TSR to use. If the vector is already in use, an error message will be displayed.

The user may access a usage screen by entering the argument **/?**.

TSR Internal Functions

The TSR has the form of a Finite State Machine (FSM). The events and states relationships are described below.

Events

The events that the TSR senses are

- Keypad — through INT 75_h chain.
- Mouse — through INT 33_h chain.
- Timer — through INT $1C_h$ chain.

INIT State

When **Enter** is pressed to end the command entry, the TSR enters the **INIT** state. If the TSR is not in memory, it is loaded, and the state is changed to **SBL_ON**. If it is already in memory, a new time-out value may be entered, and the resident state remains **SBL_ON**.

SBL_ON State

The TSR monitors the 55 mSec clock to calculate its elapsed time. When the time-out period has elapsed and no event has occurred, the current state of the backlight is saved; the backlight and video are turned off; and the state is changed to **SBL_OFF**. If an event occurs, the timer is reset to its initial value and the time-out countdown is restarted.

SBL_OFF State

The backlight cannot be turned on when the TSR is in the **SBL_OFF** state; the state must be changed to **SBL_ON** to turn on the backlight.

When either a mouse or a keypad event occurs, the backlight is restored to its saved state, and the video is turned on. The timer is then set to its initial value and the time-out countdown is restarted; and the state is changed to **SBL_ON**.

INACTIVE State

When a delay of 0 seconds is entered, the TSR is effectively disabled but remains in memory, and the state is changed to **INACTIVE**. When a non-0 delay is entered, the timer is set to the new delay value and the state is changed to **SBL_ON**.

Standby/Suspend/Resume

The Standby/Suspend/Resume messages are a special set of circumstances. Their activity is sensed from a broadcast message through INT $2f_h$.

Standby messages may be rejected when the TSR state is either **SBL_OFF** or **SBL_ON**.

Resume messages are always recognized. The timer is set to its initial value. If a **scrnblnk** time-out occurs, the backlight is restored to its saved state, the video is turned on, and the TSR state is changed to **SBL_ON**.

State Machine

States	Events					
	Initial Command Entry	Time-out	Event	Suspend or Resume	Enter Zero Delay	Enter Non-Zero Delay
INIT	SBL_ON					
SBL_ON		SBL_OFF			INACTIVE	
SBL_OFF			SBL_ON	SBL_ON		
INACTIVE						SBL_ON

Cradle TSR Function

If the cradle driver (CRADTSR) is loaded, the Ethernet driver (TEP) can optimize its power management significantly.

It does this in two ways:

1. When the unit is undocked, the Ethernet interface is put into a low-power mode called **hardware suspend mode**. This mode extends the battery lifetime by reducing power consumption.

2. When the unit is docked in a cradle with an attached Ethernet, it will not automatically go into **Standby** mode or **Suspend** mode. This keeps the unit from turning itself off while the user is downloading large files.

If CRADTSR is not loaded, TEP will continue to function as a normal packet driver. However, this mode of operation is not recommended.

Ethernet Drivers/Utilities

Installing Ethernet Drivers

To install Ethernet drivers, perform the following steps:

1. Edit the **autoexec.bat** file and uncomment the lines that call **ethernet.bat** and **cradle.bat**.

2. If the FTP TCP/IP stack is to be loaded, copy the necessary files to Ethernet/Pkt (these are not included on the unit). Edit **ethernet.bat** and uncomment the two FTP lines at the bottom.

3. Uncomment the **scrnblnk** line if screen blanking is needed to conserve power.

ETHERCTL.EXE

Etherctl.exe is a power control utility that turns the power to the ethernet chip on and off. This utility is part of the **ethernet.bat** file.

etherctl.exe [-f] {suspend | wake}

-f = ignore packet driver link status

TEP.COM

Tep.com is the ethernet packet driver and is loaded when **ethernet.bat** is run.

tep [options] <packet_int_no> <io_addr> <hardware_irq> <Ethernet

– i = Force driver to report itself as IEEE 802.3 instead of Ethernet.

– d = Delayed initialization. Used for diskless booting.

– n = NetWare conversion; converts 802.3 packets into 8137 packet.

– w = Windows hack, obsoleted by winpkt.

– p = Promiscuous mode disable.

– u = Uninstall.

– s = Scan I/O space even if a plug and play card is found.

– f = Fake hardware; used for debugging.

ETHERNET.BAT

Ethernet.bat is a batch file that turns on the power to the ethernet chip, loads the ethernet driver, and then loads the TCP stack.

> ***Note:*** *The TCP/IP stack drivers are not included on the base image and must be installed by the user. Once installed,* ***ethernet.bat*** *must be edited to load the stack.*

IrDA Power Management

IrDA power is controlled by two simple utilities. Neither of these utilities requires parameters.

JIRDAON.EXE

Jirdaon.exe is the IR power on utility for the PTC-2124.

JIRDAOFF.EXE

Jirdaoff.exe is the IR power off utility for the PTC-2124.

LP20.EXE

lp20.exe is the executable file that loads the LitePlus drivers.

LitePlus 2.0 IrDA Print Driver

Even though the LitePlus 1.10 DOS driver provides the ability to print to IrDA compliant printers with APM support, it can send data only in one direction (to printers). It is also not strictly IrDA-compliant in that it lacks the ability to act as a secondary station. The LitePlus 2.0 DOS driver is designed to provide bidirectional data transfer capability along with many other enhancements. These new features include a user-configurable buffer size and the ability to enable/disable IrDA stack, change IrDA COM ports on-the-fly, support the application callback function, and provide a high-efficiency data exchange between the driver and the application.

LitePlus 2.0 IrDA DOS Driver Architecture

The LitePlus 2.0 IrDA DOS driver is based on a completely different architecture than that of LitePlus 1.10, even though its functions are natural extensions of that program.

LitePlus 2.0 still uses the format of the DOS TSR program and supports only single point-to-point data exchange. However, the IrDA state machine will be driven by the timer interrupt rather than by the BIOS interrupts. The entire implementation will be purely interrupt driven. This will ensure that there is no waiting under most circumstances inside the IrDA driver, therefore providing maximum efficiency and consuming minimum system resources.

LitePlus 2.0 still controls INT 17H to support traditional DOS printing functions such as Ctrl-P, Print-Screen, and redirection to an LPT driver.

LitePlus 2.0 also controls INT 14H, adding new extension APIs to support two-way data transfer effectively. Direct redirection to COMx ports is no longer supported. The new APIs provide applications with direct access to IrDA stack information and allow high efficiency data exchange between the application and LitePlus 2.0.

IrDA Related Specification

LitePlus 2.0 is an IrDA primary station that can act as a secondary station. Whether it plays the primary or secondary role depends on which IrDA device initiates the connection procedure.

Physical Layer

LitePlus 2.0 supports the standard IrDA SIR Physical Layer Specification 1.0 (except for 2400 bps-only devices). Implementing 2400 bps support increases the complexity and code size without benefit; all IrDA devices start with a 9600-bps discovery and a connection sequence. A 2400-bps-only station starts its discovery with the initial command frames beginning with a minimum of five SOP delimiters (0 x C0) at 2400 bps. A station that supports 2400 bps hears the SOPs sent from the 2400 bps as a sequence of characters “0 x 77, 0 x 77, 0 x FF.” After recognizing this signature sequence, it shifts over to 2400 bps and starts from there.

LitePlus 2.0 supports two types of hardware IR implementations:

- internal transceivers, and
- external dongles.

Dongles may be attached to serial ports directly or indirectly (through a cradle). The internal transceiver design usually requires the IrDA infrared transceiver baud rate to be controlled by the baud rate registers of the COM port it occupies. Internal implementations that require additional register control may be supported by treating them as external dongles. The only external dongle used currently supports SIR speeds of 9600, 19200, 57600, and 115200 bps only.

To make an IrDA connection to an external dongle, follow these steps:

1. Install infrared drivers onto a Windows 95 PC. Connect IrDA dongle to a free COM Port.
2. Start hyperterminal, pointing the opening of the port to the assigned IrDA COM Port (i.e., COM4).
3. From the PTC-2124 DOS, run **type config.sys >lpt1**. The hyperterminal screen should display the contents of **config.sys**.

Link Access Control Layer (IrLAP)

LitePlus 2.0 supports Link Access protocol IrDA specifications 1.0 and 1.1, with the following limitations:

- Connectionless Data Services are not supported. Also, LitePlus 2.0 does not expose such services to the application.
- Sniff services are not supported. Since LitePlus 2.0 supports only single Service Access Points (SAP), such services are redundant and are not available to the application.
- A window size of only one frame is supported. For the single SAP TSR design, supporting window sizes of more than one significantly increases the amount of memory required for buffers and the complexity of state machine implementation.
- Since LitePlus 2.0 resides in main memory as a TSR, a trade-off between communication efficiency and memory resources is required. Large buffers (more memory) means less overhead when transferring large amounts of data. However, it also means less memory is available to run applications. All IrLAP APIs are not directly available to applications.

The IrLAP summary for LitePlus 2.0 parameter negotiation is provided in the following table:

Negotiation Category	LitePlus XID Response	Preferred Receipt Value
Baud rate	9600, 19200, 38400, 57600, 115200	Anything but 2400, 38400
Maximum turn around time	500 ms	500 ms
Data size	Up to 2048 bytes	Any
Window size	1 frame window	1 frame window
Number of BOFs	2 @ 115, 200 bps	Any
Minimum turn-around time	10 ms	Any
Link disconnect	40, 30, 25, 20, 16, 12, 8,3 seconds	Any

Link Management Layer (IrLMP)

LitePlus 2.0 supports IrLMP 1.0 and 1.1 with the following exceptions:

- Only one Service Access Point is supported. With a single SAP, LitePlus 2.0 can support two-way data exchange.
- LitePlus 2.0 does not include the architectural component described in the Link Management Protocol Document section 2.2.3 — Transport Protocol. LitePlus 2.0 does not support multiple IrLMP connections simultaneously.
- **Exclusive** Mode is not supported. The **Multiplexed** mode supports only simultaneous connections of IAS (Information Access Service) and data transfer channel.
- Connectionless data are not supported.
- Sniffing primitive is not supported. LitePlus provides only normal discovery services.
- The hint byte is set to 0 x 04, which identifies the IrDA driver as a Computer. The printer bit is not set even though DOS printing data can be redirected to the driver. This byte may also be used by another computer to decide/display how the unit containing the IrDA driver is used.
- Device nickname — the device nickname is exchanged during discovery sequence. LitePlus 2.0 allows the user to change nicknames through a command line option.
- The DISCOVER and RESOLVE ADDRESS states described in section 3.5.2 and 3.5.2.3.1 of Link Management Protocol Document are not implemented internally in IrLMP but rather in the IrLAP layer.
- Only the CONNECT, DISCONNECTED and TRANSFER READY states are supported in the diagram in section 6.3.1 of Link Management Protocol Document. Other states are not used by LitePlus 2.0.

- The only supported IAS primitive is LM_GetValueByClass. Other primitives, including LM_GetinfoBaseDetails, LM_GetObjects, LM_GetValue, LM_GetObjectInfo, and LM_GetAttributeNames should not be used. The IrLMP API is not exposed to the application, and all IAS services are handled by LitePlus automatically.

All IrLMP APIs, including those for IAS, are not directly available to applications. The applications will use extended INT14H functions to access the IrDA stack directly.

Classes and Attributes

LitePlus 2.0 will support the following three IAS classes and attributes:

Class	Attribute	Value
Device	DeviceName	{0,1,0,0,3,0,22, "IrDA Driver by ACTiSYS"}
	IrLMPSupport	{0,1,0,0,2,0,3,1, 0,0}
IrCOMM*	Parameters	{0,1,0,1,2,0,6,0, 1,1,1,1}
	LSAPSel	0,1,0,1,1,0,0,0,2}
IrLPT**	Parameters	{0x81,0}
	LSAPSel	{0,1,0,4,1,0,0,0,2}

* Only a three-wire-raw portion is supported for IrCOMM. Three-wire-raw is sufficient for LitePlus 2.0. More complicated protocols, such as nine-wire-cooked, target different usage models and require additional protocol support on top of IrLAP and IrLMP. TinyTP is such a protocol and is not a part of LitePlus 2.0 design.

** Supported for backward compatibility. Even printing will use three-wire-raw in the IrCOMM object instead of the IrLPT object.

IrDA Compatibility

LitePlus 2.0 is based on the ACT1 IrDA stack, which has passed IrDA compatibility tests conducted by **ACTiSYS**. Those tests include both test suites from Genoa for IrDA compatibility and interoperability tests, including many third-party implementations.

DOS BIOS INT17H Function Extension

One of the key features of LitePlus 1.10 is support for data printing through the IrDA driver under a DOS prompt. This feature is achieved by replacing the INT17H interrupt routine.

When LitePlus is enabled, all calls to INT17H are redirected to the IrDA stack only when DX is zero. This implies that LitePlus 2.0 distinguishes between different LPTx ports, and only LPT1 is mapped to the IrDA port. The reason LPT1 is always used is because all DOS printing functions, such as PRINT SCREEN, are always redirected to LPT1. When LitePlus is disabled or DX is not zero, all calls are directed to the original INT17H routine. Following are the details for the new INT 17H.

Print Character

The print character function sends the byte in AL to LitePlus 2.0's internal buffer if a connection has already been configured. If a connection has not been established, LitePlus 2.0 waits for a full discovery cycle before it returns.

Called with

AH = 0
AL = character to print
DX = 0 Only LPT1 is mapped to IrDA port.

Returns

AH = 0x90 — If successful.
0x31 — If failed.

Initialize Printer

This function initializes the IrDA driver by initializing discovery/connection.

Called with

AH = 1
DX = 0 — Only LPT1 is mapped to IrDA port.

Returns

AH = 0 x 90 — If successful.
0 x A8 — If failed.

Get Printer Status

This function always returns successfully.

Called with

AH = 2
DX = 0 — Only LPT1 is mapped to IrDA port.

Returns

AH = 0 x 90

Tests to support the following DOS printing functions will be conducted and demonstrated before release:

- Redirection of TYPE, DIR or COPY command.
- Print screen Sys Rq.
- Ctrl-P.

DOS BIOS INT14H Function Extension

Although LitePlus 1.10 supports printing through redirection to COMx (where x is the number of IrDA COM ports), LitePlus 2.0 no longer supports such usage. All printing data should be redirected to the LPT port instead. One of the major features of LitePlus 2.0 is support for two-way, high-efficiency data transfers.

Because of the well-known latency of DOS serial BIOS calls, most DOS programmers bypass the BIOS call and access the registers directly. Unfortunately, DOS does not have the capability to catch direct access to registers and make legacy programs work transparently. To use the new IrDA capability, use the extended APIs that LitePlus supports.

Instead of supporting DOS redirection to COMx similar to LPT1 ports, LitePlus 2.0 actually extends INT14H to support two-way, high-efficiency data transfer. Applications can directly access IrDA stack buffers and status through these extensions. This method also provides application software a way to disable/enable the entire IrDA stack.

Upon installation, LitePlus 2.0 installs its own INT14H routine. All calls without the correct COMx port specified, or with a function number smaller than 0 x 80, are passed over to the old INT14H. The following are details of the new INT14H functions.

Buffer Control Block (BCB)

To increase the efficiency of data transfer, structures called Buffer Control Blocks (BCBs) are exchanged between IrDA stacks and application software. The definition of a BCB is as follows:

```
typedef struct BufferControlBlock {

    int      Status; /* Internal buffer usage indicator */

    struct  BufCntlBlck* pNext;/*Pointer to next BCB structure*/

    unsigned int BufferSize;/* Size of the buffer pointed by
    pPacket */

    unsigned int DataLength;/* Length of total frame data */

    unsigned int DataOffset;/* Offset to the start of useful
    data */

    unsigned char pPacket;/* Pointer to the start of the
    buffer */ } BCB;
```

Two BCB queues are maintained independently by the LitePlus 2.0 driver: the send queue and the receive queue. For receiving data, the application can either poll the LitePlus 2.0 driver to get a BCB that LitePlus 2.0 received successfully or directly use the pointer returned by the callback function.

After the data in the BCB is consumed, the application returns the BCB to LitePlus 2.0's receive queue. For sending data, the application first requests a free BCB from LitePlus 2.0. If successful, it fills up the BCB and passes it back to LitePlus via an INT14H call.

Status

This field is used internally by LitePlus 2.0 for buffer management. The application software sets it to zero before returning it to the IrDA driver to send.

pNext

This field is also used internally by LitePlus 2.0 for buffer management. The application software should ensure not to change this field for both incoming and outgoing buffers.

BufferSize

This number is the size of the buffer pointed by pPacket and is the maximum number of bytes of data that this BCB can carry. The application should always check this field to ensure that it does not write too much data into a BCB. For future consideration, applications should assume that all BCBs could have different BufferSize values. For LitePlus 2.0, this value is set for all BCBs during the driver load.

DataLength

For incoming buffers, this field indicates the length of the data bytes in the IrDA I frame received. The number of “real” data bytes is this number minus four.

For outgoing buffers, this field indicates how many bytes are to be sent and should be set correctly by the application. The number of “real” data bytes is this number plus four.

DataOffset

This field indicates the starting index of real data in the Data array pointed by pPacket. For all BCBs, this field is four and has been included for future compatibility with more complicated protocols such as TinyTP.

For outgoing BCBs, this field has no effect.

pPacket

This is the pointer to where real data is stored. For incoming BCBs, the effective data starts with pPacket[DataOffset] with length of DataLength.

Initialize IrDA Stack

Called with

AH = 0 x 80

AL = 0 —Disable IrDA stack. Discards all data buffers that have not yet been sent. All data buffers received but not yet consumed will also be lost. This is necessary to maintain the integrity of buffer, as there could be a buffer occupied by printing data from INT17H. Once disabled, the only way to enable the stack again is to call INT 14H with AH = 0 x 80 and AL = 1.

= 1 — Enable IrDA stack with callback function. Discards all data buffers that have not yet been sent. All data buffers received but not consumed will be lost.

EX:BX (AL = 1) Address of a callback function. The callback function contains the void CallBack (char far * fpData, int iDataLength) prototype. If EX:BX was set to 0:0, no callback occurs. The received data buffer is available through other calls.

CX = 0 — Internal transceiver used on the COM port.

= 1 — ACT220L dongle is used.

0 x FFFF Default dongle. This is either the default transceiver (internal) or the one specified on the command line.

DX = Serial port 0 to 3. This parameter allows users to change the IrDA serial port "on the fly." A value of $FFFF_h$ indicates that the default port (or the one specified on the command line) should be used.

Returns

AH = 0 x 90 — Successful. This value is always returned.

ES:BX Pointer to a data structure that contains all the IrDA-related information. The definition of the data structure is listed below. After the first initialization call, the application software should be able to get most of the useful information directly from this structure.

```
typedef struct IRDA_INFO {
        unsigned int    BaudRate;
        unsigned int    LAP_Status;
        unsigned int    EngineType;
        unsigned int    IAS_Selection;
        unsigned longs  DeviceAddress;
        unsigned long   dDeviceAddress;
        unsigned char   LocalDiscoveryInfo[32];
        unsigned char   RemoteDiscoveryInfo[32];
        unsigned long   ReceivedPackets;
        unsigned long   ReceivedBytes;
        unsigned long   SentPackets;
        unsigned long   SentBytes;
        unsigned long   DataInSendQueue;
        unsigned long   DataInReceiveQueue;
        unsigned long   BadPackets;
        unsigned long   TotalReSend; } IrDA_info;
```

Although the main purpose of this call is to disable/ enable the LitePlus 2.0 driver, the returned pointer points to a data structure that allows the application to retrieve many of the counters and statistics of the underlying LitePlus 2.0 driver while the IrDA connection is active.

The LitePlus 2.0 driver is enabled by default after installation.

BaudRate

The current Baud rate used by the IrDA transceiver. The following are the possible values for this field:

1	115.2K bps
2	57.6K bps
3	38.4K bps
6	19.2K bps
12	9.6K bps

LAP_Status

This field indicates the current status of the LitePlus 2.0 driver. The following are the possible values for this field:

0	Disconnected. The driver has not yet set up a connection with another IrDA device. LitePlus 2.0 may either be idle or in the middle of a discovery/ connection procedure.
1	Connected. The driver is already connected with another IrDA device. Data transfer occurs only when LitePlus 2.0 is in this state.
2	Disabled. The driver will reject any data transfer request. This implies that if a BCB is requested, a NULL pointer will be returned.

EngineType

This field indicates what role the LitePlus 2.0 driver is currently playing. Following are the possible values:

0	Secondary station role. In this role, LitePlus 2.0 responds passively.
1	Primary station role. In this role, LitePlus 2.0 actively attempts to connect with another device.

2 Undecided. The LitePlus 2.0 is idle in disconnected mode. Which role it will play depends on whether another IrDA device tries to initiate a connection first.

IAS_Selection

This field indicates how IAS responds to inquiries. There are two possible values:

0 LitePlus 2.0 supports IrLPT only (used only by LitePlus 1.x).

1 IrCOMM, three-wire-raw (used by LitePlus 2.0).

DeviceAddress

This long word is the MAC address used by the IrDA LAP of LitePlus 2.0 (source address).

dDeviceAddress

This long word is the MAC address used by the IrDA LAP of the IrDA device that communicates with LitePlus 2.0 (destination address).

LocalDiscoveryInfo

This string contains the nickname of the local station/IrDA stack. LitePlus 2.0 always sets it to **LitePlus**.

RemoteDiscoveryInfo

This string contains the nickname of the remote station/IrDA stack.

ReceivedPackets

This long word counter contains the total packets received by LitePlus 2.0. This counter is reset each time the stack makes a transition from the connected to the disconnected mode.

ReceivedBytes

This long word counter contains the total bytes received by LitePlus 2.0. Please note that this counter is NOT the total number of bytes of effective data carried over the infrared link. It includes all the bytes in all packets and is a good indication of infrared channel quality.

SendPackets

This long word counter contains the total packets sent by LitePlus 2.0. This counter is reset each time the stack makes a transition from connected to disconnected mode.

SentBytes

This long word counter contains the total bytes sent by LitePlus 2.0. Please note that this counter is the NOT the total number of bytes of effective data carried over the infrared link. It includes all the bytes in all packets.

DataInSendQueue

This long word counter contains the total number of bytes already in the LitePlus 2.0 send data queue but not sent over the infrared link. Packets that have already been sent but not yet acknowledged are also included.

DataInReceiveQueue

This long word counter contains the total number of bytes already received by LitePlus 2.0 but not yet consumed by the application.

BadPackets

This long word counter contains the total bad packets received by LitePlus 2.0. Bad packets are those complete packets with bad CRC or partial packets. This counter will be reset each time the stack makes a transition from connected to disconnected mode.

TotalResend

This long word counter contains the total number of times a packet has been resent. A resent condition could occur for several reasons: (1) The other station does not receive the packet; (2) The other station received the packet but the received packet does not have a valid CRC; (3) The response from the other station is not received by this station; (4) The response is not valid. This counter is reset each time the LitePlus 2.0 makes a transition from connected to disconnected mode.

Send/Return BCB

Called with

AH = 0 x 81

AL = 0 — The BCB has valid data and is to be sent.

= 1 — The data in this BCB has been consumed. The application is returning the BCB to LitePlus 2.0.

Returns

AH = 0 x 90 — Success. This value is returned so long as LitePlus 2.0 is enabled.

= 0 x A0 — Fail. LitePlus 2.0 is currently in a disabled mode.

This function has two uses: (1) the application may use this function to request LitePlus 2.0 to send a BCB that has been filled with data or (2) it may return a BCB with received data already used by the application.

Receive/Get free BCB

Called with

AH = 0 x 82

AL = 0 — Ask LitePlus 2.0 for possible incoming BCB.

= 1 — Ask LitePlus 2.0 for possible free BCB.

Returns

AH = 0 x 90 — Success. This value is returned as long as LitePlus 2.0 is enabled.

= 0 x A0 — Fail. LitePlus 2.0 is now in disabled mode.

EX:BX Pointer to the BCB. It could be a BCB that contains the data LitePlus 2.0 successfully received over the infrared link or a free BCB for application to place send data. A NULL pointer may be returned in both cases. A NULL pointer means no data has been received if the application is asking for incoming BCB. It also means no free BCB is available if the application is requesting free BCB.

Get Status

Called with

AH = 0 x 83

AL = 0 — Discovery status.

= 1 — Connection status.

Returns

AX = Corresponding status.

If the application asks for discovery status, the possible return values are as follows:

0	Never discovered.
1	Already discovered, but LitePlus 2.0 is idle.
2	In the middle of a discovery procedure.
4	Already discovered, probably connected.

If the application requests connection status, the possible return values are as follows:

0	Disconnected. LitePlus 2.0 is in NDM state.
1	Connecting. LitePlus 2.0 is in the middle of a connection procedure.
2	Connected. LitePlus 2.0 is already connected with another station.
4	Disabled.

Please note that this status information is used by the application to display the status of the underlying LitePlus 2.0 driver.

Unload LitePlus driver

Called with

AH = 0 x 84

Returns

BX = 0 — Unable to unload the LitePlus driver. Before unloading the LitePlus driver, it checks to ensure that APM interrupt support is installed. If yes, it verifies that the current interrupt vector is the same as in the LitePlus driver. If not, it means another routine replaced the APM driver after LitePlus was loaded. It is not possible to unload the driver.

BX = otherwise — LitePlus has recovered all the interrupt vectors. BX now contains the _psp variable of the LitePlus driver. The caller can then use this variable to unload the driver.

Miscellaneous

Command Line Options

All command line options start with "/". The following command line options are available for LitePlus 2.0. The synopsis for LitePlus 2.0 is as follows:

LitePlus [/COM:x] [/BAUD:xxx] [/EXT:xx] [/BUFFER:x] [/NOAPM] [/SUSPENDOK] [/NAME:ssss] [/HELP] [/UNLOAD]

Note: *Options are not case sensitive.*

/COM:x This option allows the user to choose the UART port to which the IrDA hardware is connected. The “x” may be any value between 1 and 4. LitePlus 2.0 uses DOS default register addresses, and the default COM port is COM3. LitePlus 2.0 also assigns an IRQ number accordingly. The default values for DOS are as follows:

COM port	Base Register Address	IRQ
COM1	0x3F8	IRQ4
COM2	0x2F8	IRQ3
COM3	0x3E8	IRQ4
COM4	0x2E8	IRQ3

/BAUD:xxx This option allows the user to choose the highest baud rate LitePlus 2.0 supports during IrDA connections. LitePlus 2.0 itself can support up to 115.2 Kbps. Unless there is a specific reason for a lower baud rate (strong interference, background lighting), it is normally unnecessary to limit the highest speed. The default highest speed is 115.2 K.

The valid values for xxx are as follows:

96	9600 bps
192	19200 bps
384	38400 bps
576	57600 bps
1152	115.2 Kbps

Note that 38400 bps is listed here, even though ACT IR220L does not support 38400 bps. The 38400 bps is listed because the default dongle is assumed to be an internal dongle.

/EXT:x This option tells LitePlus 2.0 what external dongle is attached to the serial port. Following values are currently supported.

0	No external dongle (Default)
1	ACT IR220L

/BUFFER:x This option tells LitePlus 2.0 the maximum size for a single buffer. Following values are supported:

0	64 bytes (default)
1	128 bytes
2	256 bytes
3	512 bytes
4	1024 bytes
5	2048 bytes

This is the only control over memory allocation for buffers. Send queue and receive queue, each contain two BCBs. Each BCB contains a data buffer of the size defined here.

/NOAPM LitePlus 2.0 supports the Symbol APM (Advanced Power Management) driver. By default, LitePlus 2.0 attempts to detect APM driver and install APM support accordingly. However, if the APM driver is not detected or the user chooses to install with the /NOAPM option, LitePlus 2.0 will not install APM support. By supporting APM, LitePlus 2.0 is able to recover from a suspend/resume cycle.

/SUSPENDOK By default, LitePlus 2.0 rejects a request to suspend if it is in the middle of a communication section. However, if the driver is installed with this option, LitePlus 2.0 always honors the request.

/NAME:ssss	This option allows the name of the IrDA stack to be changed. This name is displayed when the device is discovered by Windows 95. The default value is **PTC-2124**. Please note that Windows 95 displays only the first 18 characters of the string when disconnected and can display at least 28 characters when connected. The longest name that LitePlus 2.0 can handle is 28 characters. LitePlus 2.0 truncates the string if it is longer than 28.
/HELP	If this option is used, LitePlus will simply display the following message and quit without further action.

Usage

LitePlus [/COM:port][/BAUD:baud-rate][/EXT:dongle] [/BUFFER:size][/NOAPM] [/SUSPENDOK] [/NAME:nick-name][/HELP][/UNLOAD].

Port

1 = COM1, 2 = COM2, 3* = COM3 and 4 = COM4.

Baud-Rate

Only 96, 192, 384, 576, and 1152* are valid.

Dongle

0* = internal, 1 = ACT220L.

Size

0* = 64, 1 = 128, 2 = 256, 3 = 512, 4 = 1024 and 5 = 2048.

Name

Any string. Only the first 28 characters are used. Default is PTC-2124.

Note

In the preceding parameters, "*" indicates a default setting value.

/UNLOAD — LitePlus attempts to recover all the interrupt vectors it changed and unloads itself. If successful, the message **LitePlus uninstalled** is displayed. If unsuccessful, the message **APM interrupt prevent unloading LitePlus** is displayed, and the driver stays unchanged. Notice that when the **/UNLOAD** option is used, all other options are ignored.

Initial Detection

Before installing the LitePlus 2.0 TSR driver, the software performs the following detections:

- Whether the required COM port has been detected by DOS. This is achieved by checking the BIOS data area at 0 x 40:0. The associated error message is **Serial port not found. Driver not installed**.
- Whether the LitePlus 2.0 TSR has already been installed. This is achieved by checking the **LitePlus** identification string. The associated error message is **Driver already loaded**.

If either of the detections fail, LitePlus 2.0 aborts the installation procedure with an appropriate error message.

Discussions

- Callback functions can easily disrupt or disable the system if the application terminates without first disabling the IrDA driver.
- Since callback functions are actually called from inside the interrupt routine, the application should return as soon as possible. Long delays in the callback routine could cause the behavior of the driver to be unpredictable.

NTMOUSE

This section describes NTMOUSE, Symbol's digitizer driver for the Nissha touchscreen controller.

What is NTMOUSE?

NTMOUSE is the Symbol digitizer driver for the Nissha™ touchscreen controller. This section describes the relationship among digitizers, the NTMOUSE digitizer driver, and the PENCAL pen calibration utility.

The Nissha digitizer (or touchscreen controller) generates interrupts when the screen is touched with a stylus. The data provided by the digitizer are in a very raw format, consisting of X and Y coordinates in "digitizer units." These digitizer units do not correspond to pixels; the digitizer simply converts analog X and Y voltages from the touchscreen into a digital value and generates an interrupt to the digitizer driver.

The NTMOUSE digitizer driver (sometimes called the *pen driver* or *mouse driver*) reads the stylus position in digitizer units and makes it available to applications as "mouse" input. The main roles of NTMOUSE are to (a) emulate the DOS interrupt 33h mouse programming interface, and (b) convert digitizer unit coordinates to mouse coordinates.

The conversion from digitizer units to mouse units is simple once NTMOUSE is calibrated properly. NTMOUSE monitors the current video mode of the PTC, giving it the range of valid X and Y coordinates in mouse units. For example, Video Mode 18 is 640 x 480 pixels in X and Y dimensions and has 16 colors. The digitizer simply has to convert digitizer units into a 640 x 480 range for each digitizer interrupt.

Pencal

PENCAL is the tool that configures NTMOUSE with the minimum and maximum X and Y digitizer unit coordinates for each video mode. It does this by displaying a configuration screen with four crosshairs and asking the user to tap each one in sequence.

For each pen tap, PENCAL obtains the digitizer coordinates of the crosshair. Because it knows the pixel coordinates of the crosshair, PENCAL can calculate the formula needed to convert between the two. It saves this information in a configuration file (generally **tmouse.ini**) that is read by NTMOUSE when it loads.

Using NTMOUSE

To run NTMOUSE, invoke it at the DOS command prompt using command-line arguments appropriate to the PTC display type. NTMOUSE contains the following command-line:

ntmouse [/Q] [/B] [/?]

These arguments are optional:

- **/Q** indicates the display is a 320 x 240 quarter-VGA.
- **/B** disables the beep on APM resume (available in version 3.02.02 and later).
- **/?** displays the usage line.

By default, NTMOUSE assumes a 640 x 480 “full” VGA display and expects to read calibration information from **tmouse.ini**.

The path to NTMOUSE's calibration file may be configured using environment variables. When it loads, the file looks for an environment variable named TMOUSE. This variable should contain a pointer to the configuration file (for example, TMOUSE = **c:\mouse\tmouse.ini**). If this variable does not exist, NTMOUSE reads its program segment prefix to determine its installation directory. It then tries to open **tmouse.ini** in its installation directory. If this fails, NTMOUSE will still load but will use default calibration values that may be incorrect.

The Symbol PTC-2124 has a ¼ VGA display which shows a "slice" of the full VGA display. Applications running on such systems must ensure that they do not draw outside the visible part of the display. NTMOUSE's command line arguments control how the digitizer driver must scale the digitizer units. This ensures that the units appear to be mouse events on the "visible" portion of the full VGA display.

NTMOUSE Interaction with APM

The NTMOUSE driver is APM aware. This means that it gets a notification when the PTC on which it is running is suspended or resumed. During **Resume**, NTMOUSE reinitializes the touchscreen hardware and generates a beep. The beep is NTMOUSE's signal to the user that the touchscreen is active again and is ready for input.

Units with the mouse, Ethernet, and cradle driver loaded that are left in a cradle with an active Ethernet link will emit a beep at regular intervals, whenever the unit attempts to go into APM standby mode and the Ethernet driver rejects the attempt. The reason for this is that a rejected APM standby request results in an APM resume notification to all drivers, and this causes NTMOUSE to generate a beep. This may be disabled with the **/B** command mentioned in the "Using NTMOUSE" section.

Special Notes for Programmers

NTMOUSE does not support INT 33h functions 7 and 8 (Set Horizontal Limits for Pointer and Set Vertical Limits for Pointer, respectively).

Some applications (such as the PenRight! runtime) routinely use these calls to set the horizontal limits to 0 through 640 and the vertical limits to 0 through 480. Internally, NTMOUSE uses the pointer limits to scale digitizer coordinates to mouse coordinates, so these calls interfere with its ability to provide accurate information to applications. NTMOUSE will silently ignore attempts to set the pointer limits using these two functions.

2124POP

2124POP is a pop-up keyboard utility that allows users to use the pen-based screen to touch the keys in the pop-up keyboard with a stylus to key in text on the command line or in other applications. 2124POP also has a “gas gauge” function to show battery life. This function will display a meter that displays the amount of life remaining in the battery as a percentage of the original life.

Command Line

The command string for the 2124POP is as follows:

2124pop [/option]

Options

Individual parameters within the string influence different features or operations as follows:

/D enables the Direct Video I/O Mode.

/N disables the Drag feature.

/B enables the budget Keyboard operation.

/E enables the Keyboard when pop-up is invoked.

/X:n sets the number of video columns.

/Y:n sets the number of video rows.

/K:n n = 0 – 9999, sets the standard keyboard delay (default = 25).

/S:n n = 0 – 9999, sets the secondary keyboard delay for **action** keys.

/T:n n = 1 – 5, sets the triple-tap activation time threshold (default = 2).

/R:n n= 0 – 60, sets the time-out value for the Resume Time-out macro.

/A:n n = 1 – 5, sets triple-tap activation area threshold (default = 2).

/G[:n] enables Battery Gas Gauge. n = 0 – 8 default = 2, sets the gas gauge alarm level (optional).

/M:path specifies a Macro Definition File.

/F:path specifies a file containing “action” key definitions.

PENCAL

This section describes PENCAL, Symbol's pen calibration utility. PENCAL is used in conjunction with Symbol's DOS-based digitizer drivers, such as NTMOUSE.

What is PENCAL?

To understand PENCAL, it is necessary to understand Symbol's digitizer drivers. This section gives a brief overview of how the digitizer, the digitizer driver, and PENCAL work together.

The digitizer (or touchscreen) generates interrupts when the screen is touched using a stylus. The data provided by the digitizer is in a very raw format, consisting of X and Y coordinates in "digitizer units." These digitizer units do not correspond to pixels in any way; the digitizer simply converts analog X and Y voltages from the touchscreen into a digital value and generates an interrupt to the digitizer driver.

The digitizer driver reads the stylus position in digitizer units and makes it available to applications as "mouse" input and is sometimes called the pen driver or mouse driver.

The main roles of the digitizer driver are to

- emulate the DOS interrupt 33h mouse programming interface, and
- convert digitizer unit coordinates to mouse coordinates.

The conversion from digitizer units to mouse units is simple once the digitizer driver is properly calibrated. The digitizer driver monitors the current video mode of the PTC-2124, giving it the range of valid X and Y coordinates in mouse units. For example, video mode 18 is 640 x 480 pixels in the X and Y dimensions and has 16 colors. The digitizer simply has to convert digitizer units into a 640 x 480 range for each digitizer interrupt.

PENCAL is the tool that configures the digitizer driver with the minimum and maximum X and Y digitizer unit coordinates for each video mode. It does this by displaying a configuration screen with four crosshairs and asking the user to tap each one in sequence.

For each pen tap, PENCAL obtains the digitizer coordinates of the crosshair. Since it knows the pixel coordinates of the crosshair, it can calculate the formula needed to convert between the two. It saves this information in a configuration file (generally **tmouse.ini**) that is read by the digitizer driver when it loads.

Why Use PENCAL?

PENCAL is useful because not all digitizers use the same mapping scheme between digitizer units and mouse units. Several possible reasons exist for this, including the following:

- Not all touchscreens have the same electrical characteristics. As a result, two touchscreens may report different digitizer coordinates for the same physical location on the display.
- During manufacture, the display may not be identically seated on two similar units.

If two units have the same **tmouse.ini** file, and only one displays properly, a calibration problem is likely the fault.

Using PENCAL

To run PENCAL, invoke it at the DOS command prompt using command-line arguments appropriate to the PTC display type. PENCAL contains the following command line:

pencal [/x xpixels] [/y ypixels] [/r rows] [/c cols] [/Q] [/d] [file]

All of these arguments are optional:

- **/x** indicates the number of X pixels visible in 640 x 480 mode.
- **/y** indicates the number of Y pixels visible in 640 x 480 mode.
- **/r** indicates the number of rows visible in 80 column by 25 row text mode.
- **/c** indicates the number of columns visible in 80 column by 25 row text mode.
- **/Q** is an alias for /x 320 /y 240 /r 15 /c 40 (PTC screen size).
- **/d** indicates that the BIOS is configured to use "double height" characters.
- **file** is the name of the digitizer driver calibration file.

By default, PENCAL assumes a 640 x 480, 80 column by 25 row full VGA display. Its default configuration file is **tmouse.ini**. Note that invoking PENCAL with the "**/?**" argument or "**/help**" will produce a usage line.

The Symbol PTC-2124 has a ¼ VGA display which shows a "slice" of the full VGA display. Applications running on such systems must ensure that they do not draw outside the visible part of the display. PENCAL's command line arguments tell it how the digitizer driver must scale the digitizer units it so that they appear to be mouse events on the "visible" portion of the full VGA display.

Some units use a double-height character set as the default text mode configuration, effectively reducing the number of rows on the display by half but improving readability considerably. For these units, the "**/d**" option tells PENCAL to adjust its placement of targeting crosshairs in text mode.

During calibration, PENCAL offers users the choice of calibrating several video modes. It is important that all modes used during PTC-2124 operation be calibrated. The most commonly used modes are 80-column text mode (mode 7) and 640 x 480 graphics mode (mode 18), which are the first two choices in the mode menu.

If 640 x 480 graphics mode is calibrated when PENCAL is invoked, PENCAL displays the main menu. If 640 x 480 graphics mode is NOT calibrated, PENCAL requires calibration of this mode before displaying the main menu.

PENCAL changes the digitizer driver's calibration values for each video mode after the user taps on each of the 4 crosshairs and selects the **accept** menu item. However, the digitizer driver configuration file is not updated until the user selects **save and exit** from the main menu. If settings are not saved, the calibration for each affected video mode will revert to its previous value following a reboot.

Note the following requirements for running PENCAL:

- NTMOUSE or another Symbol digitizer driver must be loaded before running PENCAL.
- The **helvb.fon** file must be in the PATH when PENCAL is invoked. When PENCAL exits successfully, it writes a configuration file containing calibration information for the various video modes. By default, this file is called **tmouse.ini**.

PENCAL Usage Notes

The following is a list of miscellaneous notes regarding PENCAL usage.

- To determine the version of PENCAL currently being used, use the "**/?**" option, which will print a usage line, including the version number and build date/time.
- If **tmouse.ini** does not exist, and the mouse is being calibrated for the first time, PENCAL requires the user to calibrate the 640 x 480 mode before displaying the main menu. This allows it to process menu selections correctly.
- During calibration, PENCAL enforces a pause between crosshair taps and before the first tap. If the user attempts to rush through calibration, PENCAL will appear to ignore the pen tap. Pause 3 or 4 seconds between pen taps and before the first tap.
- PENCAL displays a message if all video modes are not calibrated. However, not all video modes can be calibrated using PENCAL.
- If only the top two crosshairs are visible while the user is calibrating a text mode, the BIOS may be in double-height character mode. In these instances, PENCAL should be invoked with the "**/d**" option.
- When PENCAL is invoked with the "**/d**" option, the cursor may appear to be in the wrong place when the user clicks the **accept** button after calibrating a text mode. However, PENCAL should still process the **accept** button click correctly and return immediately to the main menu.

This section describes the graphical application development tool, PenRight! Pro. Throughout this document, the name “PenRight!” is used in place of “PenRight! Pro.”

PenRight! Overview

PenRight! is a software development tool as well as an operating environment for pen applications written using PenRight! PenRight! applications are window-based (form) programs. A form contains objects such as buttons, list boxes, writing fields, text, and images. A pen touches buttons to activate, select items, highlight and scroll lists, write characters or draw signatures. PenRight! provides all these building blocks, in a simple point-and-drag-then-drop fashion, to create PenRight! applications.

Features

The PenRight! application environment provides the following features:

- **Event Management**.

  PenRight! applications are driven by events that are directly or indirectly caused by user actions, such as touching the pen to a certain part of the display.

- **Resource Management**.

  Resources are used to store messages, fonts, forms, graphics, changeable code, and any other type of data that is best stored separately from the compiled application. Resources have the advantage of being easily modified without having to change and recompile the application code.

- **Font Management**

  This allows a variety of display fonts including proportionally spaced fronts. Microsoft Windows fonts may be used in the PenRight! environment.

- **Extensible Applications**

  These are code resources that are pieces of code and are stored separately from the application in the resource files. This provides the ability to attach pieces of code to the main application. They provide flexibility because they can be easily changed without having to recompile the application.

- **Digitizing Overlay Interface**

  This allows the pen's position to be displayed on the screen.

PenRight! applications consist of two components: executable code and resources.

- Executable code is like any other executable program that has been compiled, linked, and executed.
- Resources contain data or code that are created and stored separately from the application's executable code. Forms, buttons, lists, fields, fonts, graphic images, and text strings are all resources. They are compiled by the PenRight! resource compiler, then attached to the executable code to form a program.

MobileBuilder

MobileBuilder is the Rapid Application Development (RAD) tool designed to create PenRight! applications. MobileBuilder features include

- A window-based display that allows multiple windows to be handled both on and off the screen.
- Integrated development environment (IDE) with a visual form designer. Forms to collect user input through a variety of mechanisms including fields, controls, and lists. Fields are areas in which text or graphics may be written and edited. Controls are graphical objects that the user can point at and set by using the pen. Lists are scrollable lists of choices from which an item may be selected by pointing with the pen.
- Cross-platform capabilities that allow a single application design to run on Windows NT, 98, 95, 3.1, MS-DOS, DPMI, Palm OS, and Windows CE.
- A built-in code assistant and code editor.
- Bit-mapped graphical output providing maximum flexibility in displaying objects on the screen. Functions exist to draw a variety of objects including lines, rectangles, characters, patterns, etc. The output may be rotated to face any of the four sides of the display to accommodate any orientation of the computer.
- Handwriting recognition capabilities that translate hand printing into ASCII character data. “Electronic ink” is displayed as the user writes on the screen with the pen; when the user pauses, the handwritten bit image is converted to handle a wide variety of personal writing styles. Individual user training is not required.
- Flexible “C-based” architecture with over 350 built-in routines for pen-based/touch screen functionality.
- Any third-party C or C++ library, C source code, of DLL may be linked directly into application designs.

System Requirements

The pen-based computer should have the following minimum system resources to run PenRight! applications:

- VGA display,
- 192 Kilobytes of extended memory (to load runtime system files if option **/e** is specified at installation),
- MS-DOS v3.3 or later.

System Software

The following PenRight! Runtime system software must be installed on the PTC-2124 to run PenRight! applications:

- **PENR!API.EXE** — PenRight! API functions module (TSR),
- **PENR!HWP.EXE** — PenRight! handwriting recognition system module (TSR).

Installation

There are three installation routines for the PenRight! operating system. It may be loaded onto an IBM compatible PC, a RAM disk, or a PCMCIA card drive in a PTC.

PC Installation

To install the PenRight! SDK tools, insert diskette Number 1 into a diskette drive, change to that drive, and type **install** to begin the installation process. The install program guides the user through the installation process. PenRight! Pro files are loaded onto the hard disk into a directory named \Penright.

For more information about installing the PenRight! SDK programs, refer to Chapter 1 in the ***PenRight! PRO Learning Guide***.

RAM Disk Installation

To install PenRight! into a RAM disk:

1. Create a RAM disk using **rdiskcrc.exe**. Follow the procedures in the ***Programming Tools*** manual on the use of **rdiskcrc.exe**.

2. Copy PenRight! system files to the destination directory.

3. For manual loading, type the following commands from the DOS command line. For automatic loading, add them to the **autoexec.bat** and reboot the PTC-2124.

   ```
   <drive:path>PENR!API /i=64 /e /l
   <drive:path>PENR!HWP /I:65
   ```

   Where ***/e*** specifies to load in extended memory, ***/l*** to set video mode to reverse video for proper view of application screen, and ***/i*** and ***/I*** set the software interrupt vector of the TSR.

PCMCIA Card Installation

To install PenRight! onto a PCMCIA memory card:

1. Format the PCMCIA memory card using **pcformat.exe** and copy PenRight! system files to the desired directory. Follow the procedures in the ***Programming Tools*** manual on the use of **pcformat.exe**.

2. For manual loading, type the following commands from the DOS command line. For automatic loading, add them to the **autoexec.bat** and reboot the PTC-2124.

   ```
   <drive:path>PENR!API /i=64 /e /l
   <drive:path>PENR!HWP /I:65
   ```

Where ***/e*** specifies to load in extended memory, ***/l*** to set video mode to reverse video for proper view of application screen, and ***/i*** and ***/I*** set the software interrupt vector of the TSR.

Refer to a DOS manual for proper **emm386.exe** configuration.

Command Line Parameters

/b nnnn	***nnnn*** specifies a starting value for the generated numerical IDs. Normally, ID values are assigned to symbols beginning with 32768 (8000_h). Specify a decimal or hexadecimal number in the range 1 – 65535. Note that PenRight! normally reserves IDs from 1 to 32767, so do not use numbers within this range. (Hexadecimal numbers must be preceded by 0x (zerox) as in C.)
/h inputHeaderFile	***inputHeaderFile*** specifies a header file containing C language macro statements that equate symbols to fixed numerical values. The numerical values can be decimal, unsigned, or hexadecimal values up to 65,535. Any lines in the ***inputHeaderFile*** not beginning with **#define** are ignored. The following are examples of valid statements: `#define ExitButton 438` `#define FirstForm 379U` `#define FileList 0x100` `#define EditForm 0xA` ***inputHeaderFile*** may contain a complete or partial list of symbols used in the resources and forms. The ***outputHeaderFile*** generated does not include any symbols listed in ***inputHeaderFile***. Both ***inputHeaderFile*** and ***outputHeaderFile*** should be included in the application source code. ***Note:*** *The **/h** switch cannot be used with the **/i** switch.*

/i	Used for intermediate resource builds prior to the final build. Symbol IDs are not resolved, but are stored in a symbol table inside the resulting resource file. This allows the symbols to be resolved later in another resource build. When this switch is used, the output header file is not generated. ***Note:*** *The **/i** switch cannot be used with the **/h** switch.* ***Note:*** *The **/i** switch is used for building resources that will later be included in another resource file by using the RESOURCE token.*
/p	Causes the Resource Builder to pause after building the resource. Press any key to continue. RSCBUILD sets the MS-DOS exit code when it finishes. This exit code may be tested within a batch file by using the **IF *errorlevel*** statement. An exit code of zero indicates successful completion with no errors. Any non-zero exit code indicates an error.

Radio Types and Functions

The following sections provide information on the various radio types and radio functions used with the PTC-2124.

Included in the following sections are

- An overview of the RadioID function,
- Available radio types,
- Radio utility features and their usage,
- Available options, and
- The radio's power control utility.

RadioID Overview

This is a DOS utility that will detect and identify either the Radio Type or Radio Module in a particular computer. The Radio Module information is valid only on a PTC with an internal Direct Sequence Radio.

Radio Type

The radio type will be one of the following:

- Internal 900 MHz Direct Sequence,
- Internal 2.4 GHz Direct Sequence,
- PCMCIA Direct Sequence,
- PCMCIA Frequency Hopping 802.11 (3500),
- PCMCIA Direct Sequence 802.11 (4500),
- PCMCIA Direct Sequence 802.11 with TMA (2500), and
- PCMCIA Direct Sequence 802.11 11 Mbit (4800).

Utility Features

The following features are included in the RadioID utility:

- Detect direct sequence, frequency hopping, 802.11 and mininet radios.
- Can run in verbose or silent mode.
- May be executed from DOS prompt, called from a batch program, or spawned from an application.
- Sets DOS ***errorlevel*** to a code identifying the radio type or radio module.
- Can be run without the radio on.
- Can be run with the radio on (e.g., after a packet driver has been loaded).

The following are features that are unavailable with this utility:

- Will not distinguish between PCMCIA Direct Sequence 900-MHz and PCMCIA Direct Sequence 2.4-GHz radios.
- Will not detect multiple radios.
- Will not detect radios if using the PC simulator.

Usage

The radioid function uses the following command line:

radioid/[option]

Options

Options may be preceded by either "**-**" or "**/**" and may be specified in either upper or lower case:

radioid/? Help screen displays.

radioid/h Help screen displays.

radioid/s Silent mode.

radioid/r Detect radio module (093/095/024/025 etc.)
instead of "PCMCIA Radio" type.

Help Screen

The help screen usage options are as follows:

RadioID (option)

/s Silent mode,

/r Radio module,

/? Usage.

Silent Mode

This utility will not display a banner message or a radioid message identifying the radio. However, the radio type or module is still returned as a DOS error level.

Verbose Mode

This is the default mode of operation. If the **-s** operation is not used, the verbose mode will be assumed. When invoked, the utility will display the following banner:

Radio Detect Utility PCN

PCN 98-0176 V1.24

This will be followed by a message displaying the radio type or module detected.

Radio Type Identifying Messages

If the **r** switch is NOT used, then the possible radio identifying messages will be as indicated below:

Error Return	Radio Type	Displayed Message
0	N/A	No Radio Detected
1	Internal Direct Sequence 900 MHz	DS 900 MHz
2	Internal Direct Sequence 2.4 GHz	DS 2.4 GHz
3	PCMCIA Direct Sequence	Direct Sequence PCMCIA Radio
4	PCMCIA Frequency Hopping Radio (3000/3200)	FH Radio (3000/3200)
5	PCMCIA Frequency Hopping 802.11 Radio (3500)	FH 802.11 Radio (3500)
6	Direct Sequence 802.11 Radio (4500)	DS 802.11 Radio (4500)
7	Direct Sequence 802.11 Radio with TMA (2500)	DS 802.11 Radio with TMA (2500)
8	Direct Sequence 11 Mbit 802.11 Radio (4800)	DS 11 Mbit 802.11 Radio (4800)
10	PCMCIA Frequency Hopping Peanut Radio (3100)	FH 3100 Radio
240	Spectrum 24 802.11 Radio	Spectrum 24 802.11 Radio
241	Mininet Radio	Mininet Radio

Note: *Text descriptions are displayed only in verbose mode.*

DPOWCTRL.EXE

Dpowctrl.exe is the utility that is used to control power to the WAN radio on Symbol's PTC-2124. It is written to operate on any DOS-based SC400 system with either a T506 or a T130 ASIC interface to the radio.

Dpowctrl.exe replaces the **srpon.exe** and **srpoff.exe** utilities.

To execute **dpowctrl.exe**, enter the following DOS command:

dpowctrl radio xxx <**ENTER**>.

Where xxx = On, Off, or Inq(uiry).

Valid command line parameters are as follows:

On — Turns the device on. The device will remain on until the utility is run with the off parameter.

Off — Turns the device off. The device will remain off until the utility is run with the on parameter.

Inq — Displays the current device status. Returns either "Radio is on" or "Radio is off." If an invalid parameter is entered, the "ERROR: invalid command" message is displayed.

Valid command line parameters must be entered upon execution or the usage screen will be displayed and the utility exits without taking any action on the radio.

Packet Drivers

Introduction to Packet Drivers

Definition

A packet driver is the layer responsible for the physical interface to the RF card and has no transport associated with it. The packet driver is a TSR program that provides an interface between the radio and the transport layer. This TSR program implements the packet driver interface specification and isolates the upper layer protocols from the specific hardware.

The PTC-2124 packet driver is of the 802.11-compliant (DS and FH) type.

802.11-Compliant (DS and FH) Packet Driver Configuration

List of packet drivers

The following is a list of the 802.11 (DS and FH) packet drivers:

- 802.11 **DS** (2.4-GHz **direct-sequence** 802.11-compliant spread spectrum),
- 802.11 **FH** (2.4-GHz **frequency-hopping** 802.11-compliant spread spectrum).

Initialization file

The 802.11 DS and 802.11 FH packet drivers are configured by an initialization file, **pkt.ini**, which can be created and modified using any ASCII text editor. Although there are many keywords, only one SSID is necessary to cause the packet driver to operate using its built-in defaults. The SSID must match the access point's SSID.

Example

The following is an example of a minimum **pkt.ini** file:

```
[AWCPKT]

SSID= "zephyr."
```

How to Load an 802.11-Compliant (DS and FH) Packet Driver

Packet driver requirements

Here is a list of the packet driver requirements:

- A different packet driver is required for each type of wireless terminal. Normally, the packet driver is loaded by executing **radio/pkt/loadpkt.bat** then uncommenting the line for the radio that is installed in the unit.

  Example:

  To load the **awcp35c.com** driver (for the LM3500 radio) on a PTC-2124, the following line would be uncommented in **loadpkt.bat**:

  awcp35c -i pkt35.ini 0 x 68.

- Each driver requires a configuration file.

  Successive configuration overrides prior settings, and any configurations done within the application overrides the configuration file settings. The packet driver will parse a **pkt.ini** file if the file passed with the **-i** option has an **.ini** extension.

Classification of Devices

Below is a list that shows the packet driver/device association:

Packet Driver Name	Device
AWCP35C.COM	PTCs (21x4) with LM3500 radio cards.
AWCP45C.COM	PTCs (21x4) with LM4500 radio cards.
AWCP690.COM	PTCs (21x4) with Aironet 690/695 and IRM900 radio cards.

802.11-Compliant (DS and FH) Configuration Structure

Example

The following is an example of an 802.11 configuration structure:

```
typedef struct
{

    unsigned int SSIDLength;
    unsigned int SSID1Len;
    unsigned char SSID1[32];
    unsigned int SSID2Len;
    unsigned char SSID2[32];
    unsigned int SSID3Len;
    unsigned char SSID3[32];
    unsigned int SpecAPLength;
    unsigned char SpecifiedAP1[6];
    unsigned char SpecifiedAP2[6];
    unsigned char SpecifiedAP3[6];
    unsigned char SpecifiedAP4[6];
    unsigned int ConfigLength;
    unsigned int Infrastructure;
    unsigned int RxMode;
    unsigned int FragThreshold;
    unsigned int RTSThreshold;
    unsigned char NetAddress[6];
    unsigned char DataRate1;
    unsigned char DataRate2;
    unsigned char DataRate3;
    unsigned char DataRate4;
    unsigned char DataRate5;
    unsigned char DataRate6;
    unsigned char DataRate7;
    unsigned char DataRate8;
    unsigned int ShortRetryLimit;
    unsigned int LongRetryLimit;
    unsigned int TxMSDULifetime;
    unsigned int RxMSDULifetime;
    unsigned int Stationary;
    unsigned int Ordering;
    unsigned int ScanMode;
    unsigned int ProbeDelay;
    unsigned int ProbeEnergyTo;
    unsigned int ProbeResponseTo;
    unsigned int BeaconListenTo;
    unsigned int JoinNetTo;
    unsigned int AuthTimeOut;
    unsigned int AuthType;
```

```
    unsigned int AssociationTo;
    unsigned int SpecifiedApTo;
    unsigned int OffScanInterval;
    unsigned int OffScanDuration;
    unsigned int LinkLossDelay;
    unsigned int BeaconLostTime;
    unsigned int RefreshInterval;
    unsigned int PowerSaveMode;
    unsigned int SleepForDtim;
    unsigned int ListenTime;
    unsigned int FastListenTime;
    unsigned int ListenDecay;
    unsigned int FastListenDelay;
    unsigned int BeaconPeriod;
    unsigned int AtimDuration;
    unsigned int DwellPeriod;
    unsigned int CurrentSet;
    unsigned int CurrentPattern;
    unsigned int DtimPeriod;
    unsigned int RadioType;
    unsigned int Diversity;
    unsigned int TxPowerLevel;
    unsigned int RSSIThreshold;
    unsigned char NodeName[16];
    unsigned int ARLThreshold;
    unsigned int ARLDecay;
    unsigned int ARLDelay;
    unsigned char MagicPacketAction;
    unsigned char MagicPacketControl;
    unsigned int WakeMask;
} Pkt80211Cfg;
```

Packet Driver Loading Options

Packet Driver Options List

The most commonly used packet driver options for loading and unloading are listed in the table below. These options apply to all packet drivers.

Packet Driver Option	Purpose
<-c>	Forces the CLASS returned by driver_info to 11. This is needed to support **ieeedrv.exe** from FTP software.
-i	Tells the packet driver where to locate the configuration file. If this option is not present, the driver looks for **pkt.cfg** in the current directory. If **pkt.cfg** resides in a different directory or if the configuration file is not named **pkt.cfg**, then this option must be used. If the filename extension is **.ini**, the file is assumed to be in ASCII format.
<int_number>	Used as an interface for the software interrupt packet driver. It must be between 0 x 60 and 0 x 7F.
-m	Places the packet driver into monitor mode for use with monitor mode firmware.
-n	Allows usage with BYU Novell-Packet Driver Converter.

Packet Driver Option	Purpose
-p	Causes the received packets to be padded to 64 bytes before being passed to the upper layer. The length in the packet header is not affected, and the padding is done with indeterminate characters because of some protocol layers' requirements.
-t	Adds or subtracts Symbol TXP types to packets (used with old **srf_tsr** programs).
-u	Unloads the packet driver TSR. The driver cannot be unloaded if it is in use or being used by another device/driver.
-w	Enables use with Windows.
<xxx>	Specifies pkt, ptc, pen0, pen4, or p690, depending on the machine.

Startup Reason Codes

Code List

Reason codes are error codes that are displayed when the packet driver fails to load, and they apply to all of the packet drivers. Below is a list of startup reason codes that apply to all packet drivers.

Reason Code	Message	Description
0 x 80	“Reset timeout”	The coprocessor could not be reset. This message generally indicates a hardware failure.
0 x 81	“Reset error”	The coprocessor could not be reset and usually implies a hardware failure.
0 x 82	“Nop command failed”	The no operation command sent to the coprocessor was not received. This message normally suggests faulty hardware or coprocessor firmware.
0 x 83	“Config failed”	The config command sent to the coprocessor was not received, and it indicates faulty hardware or coprocessor firmware.
0 x 84	“Config timeout”	The config command timed out while waiting to complete. Generally, this message implies faulty hardware or coprocessor firmware.

Reason Code	Message	Description
0 x 85	“Config error”	The data in the configuration file is invalid for this coprocessor type. Verify that the correct default information exists in the config file of the coprocessor in question.
0 x 86	“Address timeout”	The coprocessor was unable to formulate a valid MAC address. This message usually means faulty coprocessor hardware or firmware.
0 x 87	“COP timeout”	The coprocessor timed out while waiting for a command to complete, and it generally indicates faulty coprocessor hardware or firmware.
0 x 88	“Nop did not complete”	The no operation command failed to complete. This message typically indicates faulty coprocessor hardware or firmware.
0 x 89	“Nop command interrupt failure”	The no operation command failed to generate the appropriate hardware interrupt and usually implies faulty coprocessor hardware.

ARLAN Diagnostic Reason Codes

List of Codes

The following is a list of the ARLAN diagnostic reason codes that apply to all packet drivers.

Reason Code	Description
0 x EC	EEPROM error on radio module.
0 x ED	Too many TX enable commands.
0 x EE	T410 chip failure.
0 x EF	On-chip failure.
0 x F0	Invalid parameter found in command.
0 x F1	Invalid command received by coprocessor.
0 x F2	Reserved.
0 x F3	Load code error.
0 x F4	Invalid bit rate or channel.
0 x F5	Reserved.
0 x F6	A radio ID may need to be specified.
0 x F7	Missing SS code.
0 x F8	Checksum error.
0 x F9	No more address space.
0 x FA	Transceiver not found.
0 x FB	Backbone failure.
0 x FC	SCC failure.
0 x FD	Local RAM test failed.
0 x FE	EPROM checksum failed.
0 x FF	OK.

Bar-Code System

This section presents the characteristics of various bar-coding systems currently supported by Symbol PTC-2124s. It also illustrates the configuration of bar-code devices and information gathering from scanning devices.

Bar Code

A bar code is composed of a series of bars and spaces grouped together to form a label. Standard bar-code specifications define the following:

- The ratio between wide elements and narrow elements,
- Specific patterns that represent certain characters and identify the beginning or end of code,
- The character set represented by the code.

Various types of bar codes are in use today. Since each bar-code type requires its own decode module, several decode modules are provided in the SDK. This section will address the bar codes supported by Symbol products.

Plessey (WNDCD01.EXE)

Plessey is used predominantly in the grocery industry. Plessey uses two sizes of elements: wide and narrow. The ratio of wide to narrow elements is generally 2:1.

Characters are represented by eight elements (four bars and four spaces). The character set contains the ten numeric characters 0 through 9. Plessey's start pattern is a wide bar followed by a narrow space. The stop pattern is a narrow bar, a narrow space, and a narrow bar. Two check digits are recommended for Plessey labels.

Universal Product Code (WNDCD02.EXE)

The Universal Product Code (UPC) has a fixed and automatic check digit scheme and is used mostly in the retail industry. UPC uses four sizes of elements, which are expressed as multiples of the smallest size: x, 2x, 3x, and 4x.

Characters are represented by four elements (two bars and two spaces). The length of each character is 7x.

Two forms of UPC code exist: UPCA, which encodes twelve characters per label, and UPCE, which encodes eight characters per label. The character set for both UPCA and UPCE contains the ten numeric characters 0 through 9.

Codabar (WNDCD03.EXE)

Codabar is used by the medical industry, photo industry, and libraries. Codabar uses two sizes of elements: wide and narrow.

Each character is represented by seven elements (four bars and three spaces). Characters are separated from each other by an intercharacter gap. The ratio of wide to narrow elements is 2:1 to 3:1.

The character set contains twenty characters: **$**, **-**, **+**, **.**, **/**, and **:**, **0** through **9**, and the start/stop characters **A**, **B**, **C**, and **D**.

Code 39 (WNDCD04.EXE)

Code 39 (also known as Code 3 of 9) is used in both government and industry. For example, the Department of Defense uses Code 39 exclusively. Code 39 uses two sizes of elements: wide and narrow. The ratio of wide to narrow elements is 2:1 to 3:1.

Characters are represented by nine elements (five bars and four spaces). Three of the nine elements are wide; the remaining six are narrow. Each character is separated by a space, called an intercharacter gap. The character set contains 43 alphanumeric and special characters, including **A** through **Z**, **0** through **9**, and the following special characters: **-**, **$**, *****, **.**, **%**, **/**, and **+**.

The asterisk (*****) is used as a distinct start and stop character.

Code 39 commonly uses no check digit; if one is required, the standard AIM code 39 check digit is supported and is selectable through the label options byte.

Interleaved 2 of 5 (WNDCD05.EXE)

Interleaved 2 of 5 is frequently used in warehousing and heavy industrial applications. Interleaved 2 of 5 uses two sizes of elements: wide and narrow.

Each character is represented by five elements. Two of the five elements are wide, and the other three are narrow. Characters are made up of either five bar elements or five space elements. The ratio of wide to narrow elements is 2:1 to 3:1; thus, wide bars are two to three times the width of narrow bars. The character set contains ten numeric characters (0 through 9).

The start pattern consists of two narrow bars and two narrow spaces. The stop pattern consists of one wide bar, one narrow space, and one narrow bar.

Because Interleaved 2 of 5 uses a start/stop pattern rather than a distinct start/stop character, it is prone to short reads. For this reason, Interleaved 2 of 5 should be used only for fixed-length applications.

Code 128 (WNDCD07.EXE)

Code 128 uses four sizes of elements, which are expressed as multiples of the smallest size: x, 2x, 3x, and 4x.

Characters are represented by six elements (three bars and three spaces); the length of each character is 11x. Code 128 uses three different start characters. Selecting a start character generates one of three possible character sets:

- All standard uppercase alphanumeric characters plus control and special characters.
- All standard uppercase alphanumeric characters plus lowercase alphabetic characters and special characters.
- All one hundred digit pairs from 00 through 99 and special characters.

The character set contains the complete set of 128 ASCII characters, four special function characters, three start characters, four code subset selection characters (which allow a label to use all three subsets), and one stop character. The stop character consists of four bars and three spaces.

Code 128 uses a fixed and automatic check digit scheme.

Ames Code (WNDCD10.EXE)

Ames Code (Intelus) is printed on file folders by the Medical Record Systems Division of Ames Color-File Corporation. Ames Code is a secure numeric bar code with structural similarity to Codabar. Ames Code is of variable length, from 1 to 60 characters. The range of scanning speed depends on code density.

Code 93 (WNDCD09.EXE)

Code 93 uses two sizes of elements: wide and narrow.

Characters are constructed from 9 modules arranged into 3 bars with their adjacent spaces. Bars may be 1, 2, or 3 modules wide (except for the start/stop character, which contains a 4-module wide bar). Spaces may be 1, 2, 3, or 4 modules wide.

Every Code 93 symbol consists of a leading quiet zone, a start symbol character, symbol characters representing data, two check characters, a stop symbol character and a trailing quiet zone. Quiet zones are spaces preceding the start character and following the stop character. Nominally, each quiet zone is at least 10x wide. The start and stop characters consist of 3 bars and 3 spaces each.

The following chart demonstrates the symbology used for Ames Code:

Character	Bars	Spaces
1	0 0 0 1	1 0 0
2	1 0 0 0	0 0 1
3	0 0 0 1	0 0 1
4	0 0 1 0	1 0 0
5	0 1 0 0	0 0 1
6	0 0 1 0	0 0 1
7	0 1 0 0	1 0 0
8	1 0 0 0	0 1 0
9	0 0 0 1	0 1 0
0	1 0 0 0	1 0 0
-	0 1 0 0	0 1 0
Start/Stop	0 0 1 0	0 1 0

The “0” elements represent narrow bars or spaces, and the “1” elements represent wide bars or spaces.

Wide elements are normally three times the width of narrow elements. The spaces between characters are at least as wide as a narrow space.

The start/stop character (denoted by a printed period) must be the first and last character of every message. The start/stop is not transmitted.

Code 16k (WNDCD12.EXE)

Code 16K, developed by Ted Williams of LaserLight Systems in 1989, is a multiple-row variable-length symbology encoding the full ASCII 128 character set.

Williams also developed Code 128, and the structure of Code 16K is based on existing UPC and Code 128 character patterns. There are between 2 and 16 adjacent rows, each divided by a separator bar. Rows are identified by the use of unique start/stop patterns.

Code 16K symbols may be read by modified moving-beam laser scanners or CCD scanners. Rows may be scanned in any order. Labels may be printed by standard printing technologies.

SuperCode (WNDCD13.EXE)

SuperCode was invented by Ynjiun Wang in 1994 and is in the public domain. This symbology uses a packet structure, a variant of a multirow symbology. There are precise rules for the horizontal placement of symbol characters in a packet, but greater freedom in placing packets vertically and horizontally than offered by a matrix of columns and rows in a multi-row symbology. The packet structure of SuperCode ensures that each symbol character encoding a data or error correction code word is adjacent to a symbol character encoding the packet address.

The sequence of code words is known regardless of how the packets are arranged. Not only does this allow for nonrectangular symbol shapes, but also the packets do not have to abut one another physically.

WANDDRVR.EXE

To use scanner functions, wand device drivers and other interface programs (TSRs) must be installed. The wand device driver (**wanddrvr.exe**) TSR provides application program interface (API) services. To scan bar-code labels and translate bar and space information into valid characters, two other TSR programs, **wandcdxx.exe** and **wndlib.exe**, are required. Once installed in memory, these programs provide a simple but powerful means of entering data to the application via a scanning device.

Several wand service functions allow applications to modify parameters, such as the device configuration and bar-code type, from within the program. By default, these functions are accessed through TSD Service 7Ah after the **wanddrvr.exe** is installed. To use wand services, an application calls the TSD Service 7Ah with the AL register set to the desired function number. Library functions are also used to access these services.

Loading Wand Drivers

To aid in loading wand drivers, **wand.bat** may be used. To use **wand.bat** to load the wand drivers, follow these steps:

1. Edit **/scanner/S1D/wand.bat** and uncomment the WNDCDxx drivers for the necessary symbologies.
2. Modify **/util/which.bat** by uncommenting the components of **which.bat**.

Installation Priority

The TSR programs must be installed in the following order:

1. WNDLIB.EXE,
2. WNDCDxx.EXE,
3. WANDDRVR.EXE.

> ***Note:*** *The xx in the file names above indicate the type of bar code supported.*

WANDT130.EXE

In addition to **wanddrvr.exe**, **wandt130.exe** is also required if scanner functions are to be used. **wandt130.exe** is loaded through **config.sys** and needs to be uncommented for the scanner to work.

Command Line

DEVICE=WANDT130.EXE[Option].

Command Line Option:

[-VXX] Where XX = optional new INT VECTOR.

WNDCDxx.EXE

When a bar-code label is scanned, data must be converted to characters. **wndcdxx.exe** performs the decode operation which translates the data into characters for **wanddrvr.exe**. Each bar-code type has its own decode module. The Symbol SDK includes the following modules which support the most popular bar-code types:

Decode	Module Bar-code Type
WNDCD01.EXE	PLESSEY ALPHA_PLESS ISBN_PLESS PURE_PLESS SAIN_PLESS
WNDCD02.EXE	UPC EAN UPC_EAN
WNDCD03.EXE	CODABAR
WNDCD04.EXE	CODE_3_9
WNDCD05.EXE	CODE_2_5 DISC_2_5 INTERL_2_5 INDUST_2_5 MATRIX_2_5
WNDCD07.EXE	CODE_128
WNDCD09.EXE	CODE_93
WNDCD10.EXE	AMES

These decode modules must be installed at the same time as the **wanddrvr.exe** TSR is installed. They must be available either in the directory that contains **wanddrvr.exe** or directed by the PATH environment variable to the directory containing the TSR.

The PCMCIA Card

A PCMCIA Card is a small form factor device about the size of a credit card. The card provides superior expansion capability to portable and notebook computers.

The PTC-2124 uses an Intel PCMCIA controller to access the two PCMCIA slots (Slot 0 and Slot 1) which are accessible through the access door on the top of the unit. Slot 0 is closest to the display and Slot 1 is farthest from the display.

Location of External PCMCIA Slots

The PCMCIA slots in the PTC-2124 will support a variety of cards. Most of the card types that have been installed seem to work. This includes spinning drives, memory, modem, and network cards.

PC cards can be one of the following types:

Type	Description
Type I	3.3 - mm thick
Type II	5 - mm thick
Type III	10.5 - mm thick
Type I Extended	3.3 - mm thick, with 40 - mm extended length
Type II Extended	5 - mm thick, with 40 - mm extended length

With the architecture provided by the PCMCIA standard, many special types of PC cards are available. Some examples are listed below:

- Data storage cards,
- Pager receivers,
- Global Positioning System (GPS) cards,
- Audio cards,
- Data encryption cards,
- Serial interface cards,
- A/D and D/A conversion cards.

Formatting an SRAM Card

Note: *Do not attempt to format a PCMCIA SRAM card using the DOS FORMAT command; use* ***pcformat.exe****.*

An SRAM data card can hold up to 4 MB of data. PTC-2124 PCMCIA drives accept memory cards ranging from 1 MB to 4 MB in size. If using a new memory card, it must be formatted prior to writing data to it. To format the card, it is necessary use **pcformat.exe**, which is included in the unit's PCM directory. Command line options for **pcformat.exe** are listed later in this section.

Once formatted, the memory card functions as a typical DOS file disk. DOS commands may be used to copy or delete files and to write or read data to/from files on the card.

Formatting an ATA Card

An ATA data card can be of solid state or rotating media type. PTC-2124 PCMCIA drives accept a wide variety of ATA storage cards. If using a new memory card, it must be formatted prior to writing data. To format the card, it is necessary to know the card's storage capacity and how to use **hdfmt.exe**, which is included with the standard image. Command line options for **hdfmt.exe** are listed later in this section.

Once formatted, the memory card behaves as a normal DOS file disk. DOS commands can be used to copy or delete files and to write or read data to/from files on the card.

Installing a PCMCIA Modem

A PCMCIA Modem is typically installed in either Slot 0 (the slot closest to the display) or Slot 1 on the unit. For the unit to recognize the modem properly, **config.sys** must be edited and the following line must be uncommented:

```
device=c:\pcm\pcmscd.exe
```

The modem is addressed as COM4.

PCMCIA Card System Software

This section discusses the PCMCIA device drivers included on the unit. These drivers allow the formatting of an SRAM card, reading and writing data to the card, and the use of other types of PC cards for the PTC-2124.

The following device drivers are described in this section:

- CNFIGNAM.EXE
- PCFORMAT.EXE
- HDFMT.EXE
- PCM.EXE
- PCMATA.SYS
- PCMCS95.EXE
- PCMSCD.EXE
- PCMSSIT.EXE
- DPMS.EXE

CNFIGNAM.EXE

Description

Specifies which PCM configuration parameters to use based on the selected boot configuration.

Command

```
DEVICE = <drive:\path\>CNFIGNAM.EXE
         [/config_name]
```

Note: *The **/config_name** parameter identifies the PCM configuration to use.*

*The **pcm.ini** file may contain multiple setup configurations for PCM. Each of these configurations is identified and separated by square brackets.*

PCFORMAT.EXE

Description

Formats the PCMCIA SRAM card to a DOS FAT file structure using the PC card drive in the PTC-2124. This program works exclusively on the PTC-2124 PCMCIA drive. The “drive:” parameter specifies the drive in which the card resides which drive is to be formatted.

Command

PCFORMAT <drive:>

Formatting Procedure

1. Insert the PCMCIA card to be formatted into Slot 0 or 1. Formatting will destroy all data on the card.

2. At the DOS prompt, enter **pcformat** and the drive letter that is to be formatted. Press **Enter** and the following message displays:

> Insert a new PCMCIA SRAM card
> into the PCMCIA slot and press
> **Enter** when ready

Press **Enter** and the SRAM card formatting begins.

Upon completion, the formatting results will be displayed.

HDFMT.EXE

Description

Formats the PCMCIA ATA card to a DOS FAT file structure using the PC card drive in the PTC-2124. This program works exclusively on the PTC-2124 PCMCIA drive. The “drive:” parameter specifies which drive is to be formatted.

Command

HDFMT <drive:> [/option]

Options

/V[:label]

Specifies the volume label.

/Q

Performs a quick format.

/B

Allocates space on the formatted disk for system files.

/S

Copies system files to the formatted disk.

PCM.EXE

Description

Displays information about the PCMCIA card system.

Command

<drive:\path\>PCM.EXE

To use the PCM Plus configuration program for DOS, follow these steps:

1. At the DOS prompt, use the CD command to move to the PCM Plus directory.

2. Type PCM and then press **Enter**.

I/O Card Configuration

Although the configuration program displays information about all card types, it can configure only I/O cards. PCM cannot configure other types of cards, such as ATA or flash memory cards. The PCM will report a non-I/O card as being not configured. Equally important, PCM Plus cannot reconfigure I/O cards that were originally configured by their own proprietary client driver.

When running the configuration utilities, PCM Plus displays various windows and menus that contain two types of fields:

- Fields for viewing only,
- Fields that can be modified.

These fields are described in the sections below. Refer to these sections when using the configuration utilities and their corresponding options.

Read-Only Fields

Note: *While in the DOS mode, some screens may be too large for the display.*

Socket Number is the number of the socket in which the card currently resides. If the computer possesses more than one PCMCIA socket and presently has more than one card inserted, use the PgUp/PgDn keys to display data about the card in the neighboring socket.

Config. Number is the configuration number assigned to the current list of parameters and the total number of configurations available to program. Use the arrow keys to switch among them.

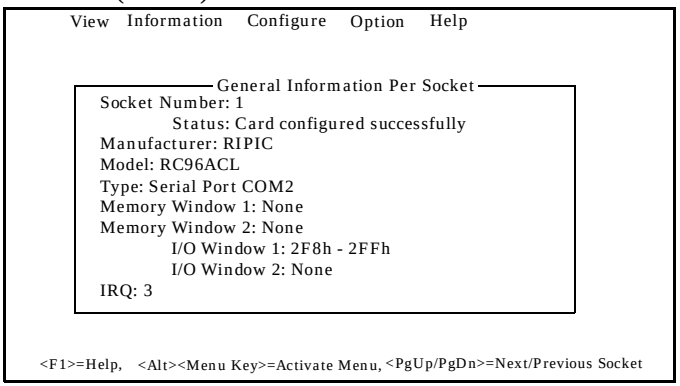
Manufacturer is the manufacturer's name.

Model is the model type or number. For instance, if this is a LAN card, this field identifies whether it is an Ethernet or Token Ring card. This field can also show a serial number.

Type is the card type, such as network LAN adapter, flash memory, modem/fax, etc.

Compliance represents the version of the PCMCIA standard to which the card complies. Configuration Loaded indicates that PCM Plus uses this set of configuration values to operate the card.

Configuration Registers are the starting addresses for the register(s) where the configuration information is stored (in hex).



Editable Fields

Memory Window controls the address range in which the memory window resides for this card when applicable. This field is followed by an 8- or 16-bit descriptor for the access type. The descriptor cannot be changed.

I/O Window indicates the I/O window address range for this card. This field is followed by an 8- or 16-bit descriptor for the access type.

IRQ controls the interrupt request level for this card.

The following fields will be displayed:

Status In the case of an SRAM card, this field may indicate that the battery is either good, dead, or low.

In the case of a flash card, this field indicates that write protect is either Off or On.

Technology Will identify the use of either Flash or SRAM.

File System Will identify whether the file system is unformatted or unknown, FFS2, or TFFS.

If more than one socket and card are installed in the computer, use the PgUp/PgDn keys to view them.

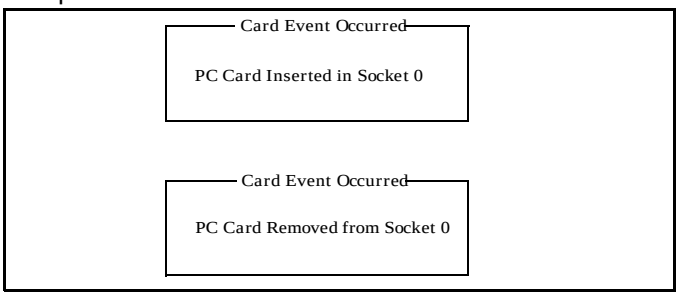
Check the Status line. This line displays the current status of the resident card. If PCM Plus configured an I/O card automatically, the status reads "Card configured successfully." If the program did not recognize an I/O card or if the I/O card has no card information structure (CIS), the line reads "Unconfigured," and the card must be configured manually with the PCM configuration program.

To perform an I/O card configuration manually, edit an existing configuration, or define an I/O card that PCM Plus cannot recognize, see the section titled **"Advanced Information" on page 137**.

In the case of an I/O card being "Unconfigured," the Status line may also indicate that the requested configuration (I/O, IRQ, or memory) is unavailable. These resources are unavailable if PCM Plus excluded them to avoid hardware conflicts. The Status line displays the detailed cause of the failure when PCM Plus recognizes an I/O card but cannot configure it.

When inserting or removing a card while PCM Plus is running, the program displays one of these two messages:

Sample PCM Screen



To exit, press **Alt+V** and select the **Exit** option, either by highlighting **Exit** and pressing **Enter** or by simply pressing **X**.

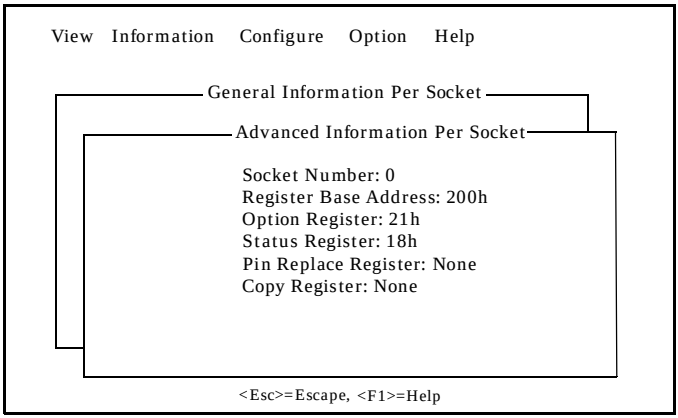
Advanced Information

Not all information about a specific card is available on the General Information screen. For additional information, refer to the Advanced Information screen, which includes the following:

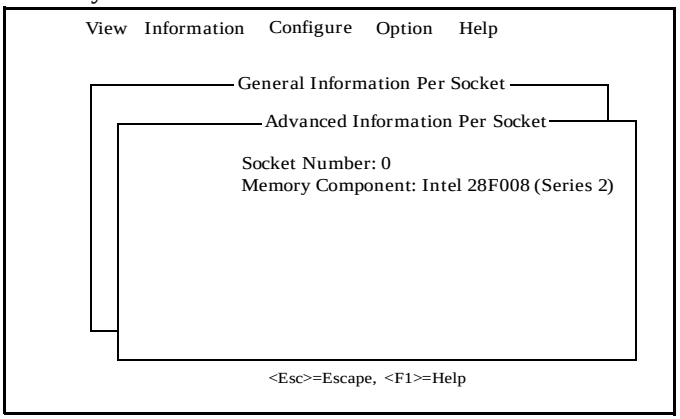
- Register base address,
- Option register,
- Status register,
- Pin replacement register,
- Copy register.

To access the Advanced Information screen, press **Alt+V** at the PCM Main Menu and select **Advanced Information**.

I/O Card Advanced Information Screen



Memory Card Advanced Information Screen



The above screen displays for a memory card. The Memory Component field indicates the technology recognized by PCM Plus. This field indicates SRAM for an SRAM card and Intel, AMD, or Mitsubishi for a memory card.

After reviewing the Advanced Information screen, press **Esc** to return to the Main Menu.

Configure

To configure or reconfigure a card, follow these steps:

At the Main Menu, press Alt+C and select one of the following options:

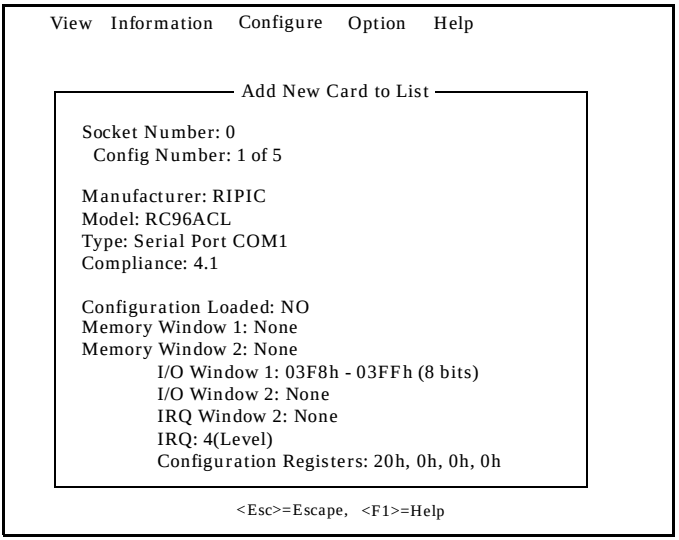
- Select **Add Card to List** and follow the **"Add Card to List"** section before assigning a set of configuration values to an inserted card that does not appear in the card list.
- To edit the configuration values of a card that is already in the card list, select **Edit Config Parameters** and then go to the **"Edit Config Parameters"** section and follow the instructions.

Note: *Before a card's configuration values can be modified, the card must be in the card list.*

Add Card to List

Select **Add Card to List** if PCM cannot identify a card or to define custom card configuration values.

Add New Card to List Screen



An installed card often has several configuration possibilities, with each configuration defining a different set of values for memory, I/O, or IRQ settings.

1. Press the up and down arrow keys to display the configuration settings for the card. Each field shown in the Add New Card to List screen is defined in the **"I/O Card Configuration"** section.

   Within the Add New Card to List screen, the I/O Window and Memory Window fields may display a wild card. The acceptable ranges for these fields are listed below:

   Memory Window: $A000_h$ to $EF00_h$

   I/O Window: 0100_h to $03FF_h$

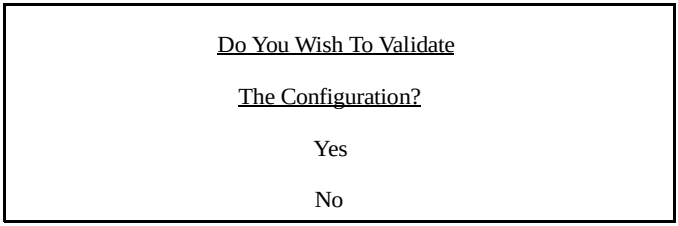
If the values are not entered in these fields, PCM Plus displays an error message when it attempts to validate the configuration and requests that a value be entered.

2. Select the desired configuration by pressing **Enter**. The PCM then asks for validation of the configuration.

3. Select **Yes** to test the selected configuration. If there are any conflicts, PCM reports them and allows the user to change the conflicting value. If there are no conflicting values, the program reports the configuration as successful.

4. Select **No** to bypass validation and testing.

   PCM then displays the following message:

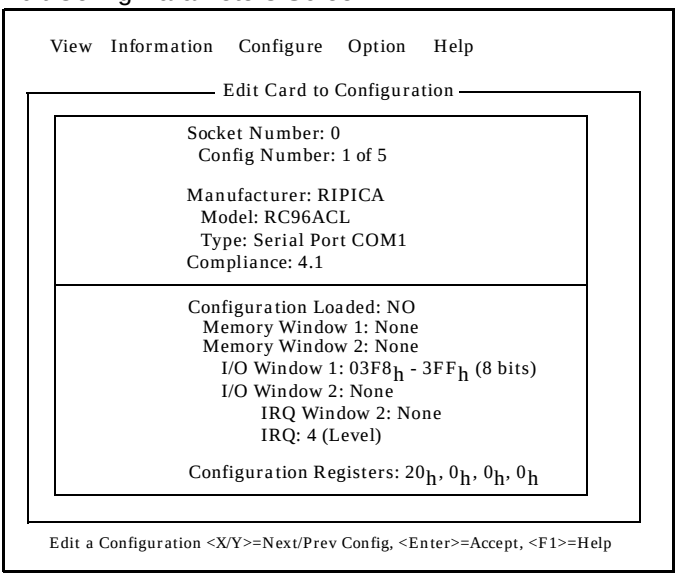


5. Select **No** to return to the main configuration menu without saving the new values, or select **Yes** to save the new values and have these parameters take effect immediately. PCM Plus automatically writes the configuration information to the PCM.INI initialization file.

Edit Config Parameters

1. Select **Edit Config Parameters** to edit the configuration values of a card already added to the card list. PCM displays the following screen:

Edit Config Parameters Screen



Some of the lines in the Edit Config Parameters screen may be edited, but some lines are read-only. See the **"I/O Card Configuration"** section for definitions of the fields.

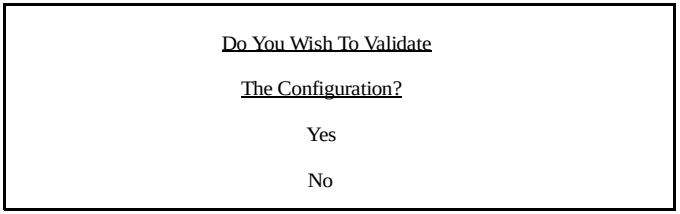
2. At the Edit Config Parameters screen, press the **up** and **down** arrow keys to display configuration settings already defined for the card.

3. Press the **Tab** or **down arrow key** to move into the fields that can be edited. Use the **left** and **right** arrow keys to move within a field. Press the **up** arrow or **Shift+Tab** keys to move backwards through the fields.

4. Change a field by typing in new values.

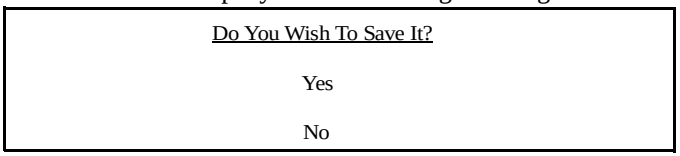
5. Press **Esc** at any time to return to the Config. Number field.

6. When all the values required to configure a card have been defined, press **Enter**. PCM then asks for validation of the configuration and displays the following message:



7. Select **No** to bypass the validation and testing process or select **Yes** to test the configuration values. If there are any conflicts, PCM reports them and allows the conflicting value to be changed. If there are no conflicting values, the program reports the configuration as being successful.

   PCM then displays the following message:



8. Select **No** to return to the main configuration menu without saving the new values, or select **Yes** to save the new values, and these parameters will take effect immediately. PCM Plus automatically writes the changes to the **pcm.ini** initialization file.

Information

PCM Plus also permits access to a list of I/O cards, the configuration sets that have been saved to the **pcm.ini** initialization file, and a list of active PCMCIA client drivers. Such access makes it easy to quickly review configuration information on all I/O cards previously configured under PCM Plus as well as information about the PCMCIA client drivers.

Card List

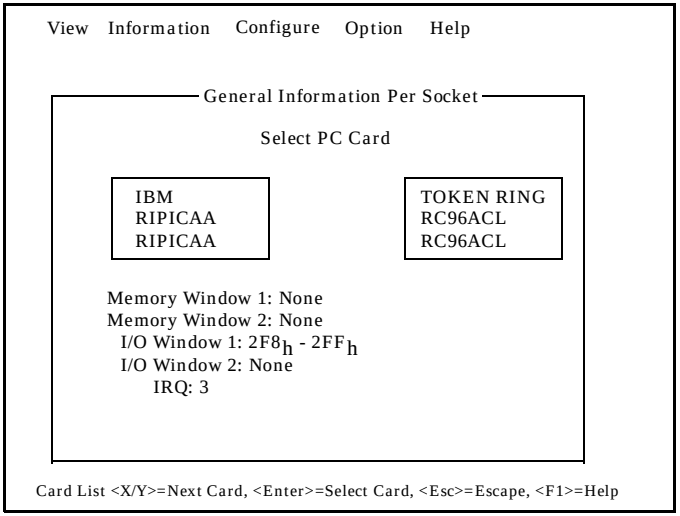
The card list displays all PCMCIA cards currently enabled for use under PCM Plus along with specific configuration information. Not all cards need to be listed in this database to be used by PCM Plus. Normally, LAN, fax/modem, and other cards that use multiple system resources should be added to the configuration list. To add a card to the configuration list, follow the instructions in the **"Configure"** section.

To review the card list, follow these steps:

1. At the PCM Main Menu, press **Alt+I**. The following options display:

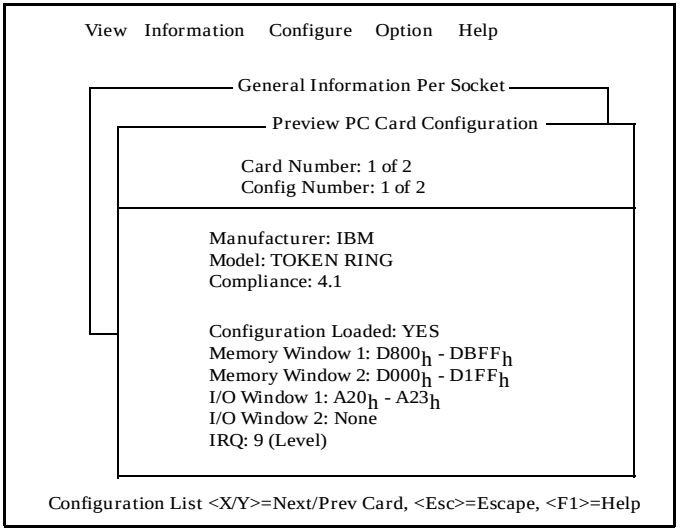
2. Select Card List to display the Card List screen:

Card List Screen



3. Press the **up** or **down** arrow keys to highlight the desired card and press **Enter**. PCM then displays the following screen:

Preview Card Configuration Screen



4. Use the **up** and **down** arrow keys to review other sets of configuration values for the selected card.

5. After reviewing the list, press **Esc** to return to the Main Menu.

Client Information

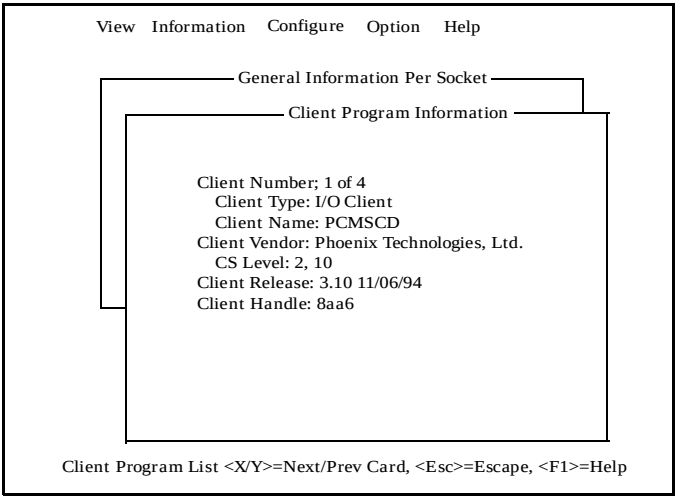
The Client Program Information screen lists the PCM Plus and third-party client drivers currently loaded and provides specific information on each client. The following information is provided: client number, client type, client name, client vendor, CS level, client release (and release date), and client handle. To review the client information, follow these steps:

1. At the PCM Main Menu, press **Alt+I**. The following options display:

Card List
Client Info

2. Select Client Info to display the Client Program Information screen:

Client Program Information Screen



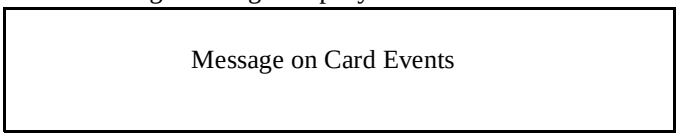
3. Use the **up** and **down** arrow keys to review other clients.

4. After the list has been reviewed, press **Esc** to return to the Main Menu.

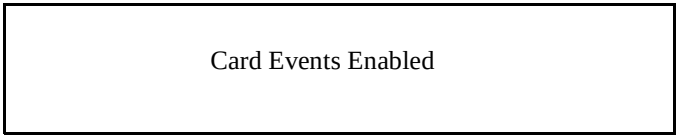
Option

Option enables the program to confirm whenever a PCMCIA card is inserted or removed from the computer. To activate this option, follow these steps:

1. At the PCM Main Menu, press **Alt+O**. The following message displays:

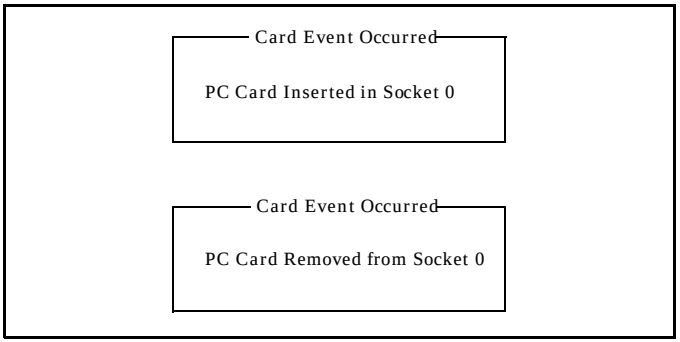


2. Press **Enter**. The following message displays:



Once this feature is enabled, PCM Plus displays one of the following messages each time a card is inserted into or withdrawn from a PCMCIA socket or when a card is configured successfully:

Card Events Screen



When Option is activated, a check mark appears before the pull-down message. To disable this option, repeat Steps **1** and **2** above. PCM displays a message to indicate that the CardEvents feature has been disabled, and the check mark disappears from the pull-down message.

PCMATA.SYS

Pcmata.sys, the Phoenix ATA IDE device driver, acts as an I/O client to Card Services. **Pcmata.sys** enables the system to access ATA-configured PCMCIA cards as IDE hard drive devices using an IDE partition table and maps the first socket (Socket 0) to the first available drive letter (usually D:), the second socket (Socket 1) to the next available drive letter (usually E:), and so on.

Pcmata.sys loads from the **config.sys** file, and the pcmata= line appears in the **pcm.ini** file.

Command

DEVICE = <drive:\path\>PCMATA.SYS

Parameters

The following options are supported. Refer to the **pcm.ini** file for recommended parameter specifics:

/Sn=x

Where:

n specifies the socket number,

x designates the number of partitions allocated for that socket.

Each socket requires its own argument. For example, /S1=2 indicates that Socket 1 has two partitions.

When no socket is specified, the program configures all installed sockets with one partition. Socket numbers start at zero.

/SRAM

Enables PCMATA to access SRAM cards that use 16 KB of upper memory to configure properly.

/ADDR=nn

nn specifies the base address of the 16-KB memory-mapped window required to configure SRAM cards. When no address is specified, Card Services assigns an available address.

/SD=mm

Spins down the drive after **mm** minutes. When no time is specified, Card Services assigns no spin down.

/RM

Forces real mode and disables DOS Protected Modes Services (DPMS). As a default, DPMS loads most of its executable code into extended memory to maximize available conventional memory.

PCMCIA Plus enables both the ATA driver (**pcmata.sys**) and the flash file driver (**pcmmtd.exe**) to read SRAM cards. Adding the /SRAM option to PCMATA and loading PCMMTD permits access to one SRAM card with two drivers. As long as the card has been formatted properly with DOS FORMAT or contains a FAT-compatible disk structure, PCMATA can access one of the following PCMCIA cards:

- A PCMCIA ATA IDE fixed disk that has been properly configured (if necessary, use PCMFDISK to assign partitions; then format the disk).
- A PCMCIA 1.0/2.0-compliant SRAM PC card (read/write).

If the system has more than one socket, all the sockets can be used for ATA disk cards so long as not more than one card lacks a CIS.

Error Messages

Syntax Error in Command Line

Make sure the proper syntax is used for command line parameters.

Card Services Not Found

Pcmata.sys requires Card Services (**pcmcs95.exe**) to run. Make sure Card Services is loaded before loading **pcmata.sys**.

Cannot Open System Window

Make sure a valid value for the **/ADDR=** option is supplied when loading **pcmata.sys**. CC, CD, CE, and CF are valid values. Use a value not already in use by Card Services.

Drive Not Ready

Socket controller initialization failed. The hardware is not working properly.

Drive Time Out

The ATA driver could not verify the status for the socket. The hardware is not working properly.

PCMCS95.EXE

Description

This is a DOS-loadable device driver as well as a TSR utility. It can be installed through either the **config.sys** or the **autoexec.bat** file. However, to support all card drivers, including the SRAM card, **pcmcs95.exe** must be installed before the device-dependent driver.

Because of the above installation requirements, **pcmcs95.exe** must be installed by the **config.sys** file to support a fixed disk ATA card or a flash memory card. To support I/O cards such as modem, fax, or Ethernet, this driver can be installed by either the **config.sys** or the **autoexec.bat** file.

For instance, when a PCMCIA card is inserted into a socket, Card Services determines if the requested resources can be provided for the card. If **pcmscd.exe** is loaded and the **/BEEP** option is enabled in **pcm.ini**, audible tones signal the configuration change and the attempt to allocate resources. **pcmscd.exe** is the Phoenix Super Client Driver that supports the configuration of many PC cards such as network and modem cards. Card Services also informs registered client drivers about card events such as card insertion and extraction. It also indicates status changes such as low battery, card ready, and card locked.

Pcmcs95.exe can be loaded either as a device driver using **config.sys** or as a TSR program via **autoexec.bat**. To load it as a driver, use the first command line statement above in the **config.sys** file. To load it as a TSR, use the second command line statement above in the **autoexec.bat** file.

Command

DEVICE = <drive:\path\>PCMCS95.EXE

See also: **pcmata.sys**, **cd.exe**, **pcm.ini**

Arguments

/ADDR=nn

Required entry. Defines the starting segment address for Card Services configuration data space. It must be within the first 1 MB of address space (xx00:0000) and must specify the start address of a two-digit hexadecimal segment address. The valid values are CCh, CDh, CEh, and CFh. For example, /ADDR=CCh means that a 4-KB window starting at CC00h is used as the Card Services parameter area.

/CLIENTS=nn

Allows **nn** clients to register. Specifies the maximum number of client drivers that can be registered with Card Services. Each client driver requires Card Services to allocate 60 bytes of memory. The default value is 10.

/IRQ=n

Enables IRQ **n** for status-change interrupts. Specifies the Card Services IRQ level. Defines the IRQ resource that Card Services uses for PCMCIA card events such as insertion and removal. This IRQ is unrelated to the communication (COM) ports.

Interrupt range: 8 – 15.

Default value: 10 (if not specified).

/MCA

Enables MCA compatibility mode. This option handles interrupts and card events as if it were working on a microchannel bus. Use this option if Card Services cannot automatically detect MCA.

/NOPM

Ignores all power management events. This option disables all Card Services power management and prevents Card Services from monitoring power management events. Use this option to prevent collisions with non-APM-compliant power management schemes.

/NOBEEP

Turns off the tone emitted when cards are inserted or removed.

/PMOFF

Disables sending the power management option provided with Card Services. If /PMOFF is specified, Card Services unilaterally powers down PCMCIA cards on Suspend messages. On Resume messages, Card Services powers up and reconfigures PC cards (if there is an associated client driver) by sending an artificial insertion message.

/POLL

Required entry. Enables poll card events. Enables sense-driven, rather than interrupt-driven, card events. This switch is useful for noisy hardware environments or when an IRQ is not available. This mode is automatically enabled when an MCA is detected or when the /MCA option is used. When more than one adapter is present, use /POLL to avoid multiple interrupts from monitoring card events.

/REGIONS=n

Uses up to **n** different memory regions at a time.

/RM

Forces real mode operation. As a default, PCM Plus loads **pcmata.sys** into extended memory. Though not recommended, the /RM option can be used to load **pcmata.sys** into conventional memory and disable DPMS. If using the /RM parameter, PCM Plus does not require Novell's **dpms.sys**.

/WAIT=n

Specifies the card settle time in system timer ticks (18.2 per second). These ticks determine the required delay period from the time a PC card is inserted to the time at which it becomes accessible. This option is useful for PC cards that have longer settle times.

Default value: 1

Range: 1–100

/XIRQ=xxxx

Specifies system IRQs to exclude. On a bare IRQ system as shown below, Card Services can use only IRQ10, IRQ11, IRQ12, or IRQ15 for card events. Fully loaded systems such as multimedia systems often require these same IRQ levels. In such cases, use /IRQ to exclude these IRQs from Card Services consideration and to avoid conflicts.

The xxxx variable decodes to these IRQs:

/XIRQ=0400 Excludes Card Services from using IRQ10.

/XIRQ=0800 Excludes Card Services from using IRQ11.

/XIRQ=8C00 Excludes Card Services from using IRQ10, IRQ11, and IRQ15, leaving only IRQ12 available.

The PCM Plus Card Services driver (**pcmcs95.exe**) functions as an extension to the operating system by coordinating access to the PCMCIA cards and allocating system resources among client drivers. The Card Services driver must be loaded from **config.sys** directly after Socket Services (**pcmssit.exe**). Card Services has the following responsibilities:

- Manage all available resources for PCMCIA cards. When a PCMCIA card is inserted into a socket, Card Services determines whether it can provide the resources required by that card's configuration.
- Manage client drivers that are written for specific PC cards. These client drivers are registered with Card Services during initialization (system boot-up). Card Services then provides the registered client drivers with pertinent information when PC card events occur (card insertion or extraction) as well as status changes, such as battery low, card ready, and card locked.

PCMSCD.EXE

Description

The Phoenix Super Client Driver, a Card Services client, increases the efficiency of resource acquisition by requesting system resources (memory, I/O, IRQ) from Card Services. When requested in this way, the resources appear to the system as resident for use by software.

PCMSCD must reside in the same directory as the active **pcm.ini**. PCMSCD reads **pcm.ini** to determine the boot options and the configuration information used for enabling PCMCIA cards.

Command

DEVICE = <drive:\path\>PCMSCD.EXE

When PCMSCD is booted, selected card configurations are loaded from the initialization file into a list maintained by PCMSCD. When a PC card is inserted, PCMSCD identifies the card and looks for a possible configuration in that list. If it finds a configuration that matches the card, PCMSCD attempts the configuration. If it cannot find a matching configuration, PCMSCD examines the CIS configurations on the card. If the card is a generic modem/fax/serial card, PCMSCD tries to enable the card as such. If a configuration is successful, PCMSCD beeps once.

Note: *As a default, the program emits a high-pitched beep when a card is inserted or configured successfully. When a card is extracted, PCMSCD emits a low-pitched beep.*

PCM Plus automatically loads **pcmscd.exe** into the **config.sys** file and writes the PCMSCD= argument to the **pcm.ini** file.

Parameters for PCM.INI

/NOBEEP

Disables beeps whenever a LAN or fax/modem PC card is configured.

/NOMS

Does not save modem state information upon power down.

/NODB

Disables the internal card information database.

/NOMODEM

Disables the generic configuration for modems. The system no longer recognizes fax/modem PC cards automatically.

/NW

Does not wait on an error/warning message.

/RM

Forces real mode and disables DOS Protected Mode Services (DPMS). As a default, PCMSCD uses DPMS and loads most of its executable code into extended memory to maximize available conventional memory.

PCMSSIT.EXE

Description

Pcmssit.exe is a Socket Services device driver loaded through **config.sys**. To modify the device driver options for Socket Services, use PCMSETUP advanced mode or edit the PCMSSIT= argument in **pcm.ini**.

Command

DEVICE = <drive:\path\>PCMSSIT.EXE

Parameters for PCM.INI

/RS=n

Reserves socket **n**.
No argument indicates none.

/RW=n

Reserves socket window controller **n**.
No argument indicates none.

/APOFF

Disables hardware auto-power mode.

/LOCK

Enables socket-locking features.

/RS=n1n2 specifies the number of reserved sockets. The Reserve Socket option prevents Socket Services from colliding with other dedicated PCMCIA software that directly access the socket hardware. Also, use this option to limit access to a socket that may not be accessible in the system. /RS reserves a maximum of two sockets.

/RW=n1n2 reserves a socket window controller. Reserve either I/O or memory windows to prevent conflicts with software that are not allocating resources from Card Services; **n** specifies the window number starting at zero on the first socket controlled window. There is no distinction between I/O and memory windows because the number and type of windows are controller specific. /RW reserves a maximum of two windows.

/APOFF disables auto-power mode on some socket controllers. Use this option for maximum compatibility on MCA-bus machines or for controllers that have problems with auto-power mode.

/LOCK enables socket locking. This switch works only if the PCMCIA hardware has socket-locking capability.

PCMAPM

Description

PCMAPM requires that **power.exe** be loaded and that a v1.1 compliant APM BIOS is used. PCMAPM is a utility used to help control Advanced Power Management of PCMCIA cards.

Command

pcmapm [/?] [/info]

/? This help screen.

/info Advanced Power Management information.

DPMS.EXE

Command

[DEVICE =][d:path]DPMS.EXE [/?|/H]

/? /H displays this help text

Description

DPMS (DOS Protected Mode Services) can also be used to map key PCMCIA functions above 1 Mb. In this case, the software memory footprint is also reduced by more than a third of traditional PCMCIA software. This leaves the PTC-2124 with more memory available for running vertical or horizontal software applications.

The conventional memory region is the area in which most drivers and software must be loaded to be accessed by the OS. Memory management software, such as DPMS, actually loads a very small component into this region while loading additional components into extended memory.

PCM automatically installs the PCM components (card services, super client driver, and ATA driver) into the extended memory region. When PTC card activity occurs, the DPMS server responds to a request from **dpms.exe** to activate the appropriate response from PCM.

PCM.INI

A typical **pcm.ini** file is illustrated below:

```
; PCM.INI - Initialization File for PCM+ Version 3.10

[COM]

1=0x3f8,0x8,0x4,0x1

2=0x2f8,0x8,0x3,0x1

3=0x3e8,0x8,0x4,0x1

4=0x2e8,0x8,0x3,0x1

COMORDER=4

[DEFAULT]

pcmss=/CMDevID=FFFFFFFF

;

If you are having problems with pcmcia devices not working
properly and you are using a emm driver (emm386,qemm,...) add:

PCMCS95=/ADDR=CC/poll/WAIT=12

pcmscd=

pcmrman=

If you are using a two-slotted device, you will need to add to the
following line ;/S1=1; if you want to have both slots support ATA
type devices:

pcmata=/ADDR=CF/SD=5/S0=1/SRAM

pcmffcs=

UseCards=

SetupMode=1

FFSPath=C:\PCM

;

If you are using the umb space cd00 for other devices, you will need
to change this line to another umb space

;

FFSWindowBase=cd00

FFSWindowSize=4

FFSNoOfPartitions=1
```

```
FFSNoOfEraseQueues=1

ComPort=4

ATAWindowBase=cf00

ATASpinDownTime=5

ATANoOfPartitions=1

IRQ=10

;

;if you are using the umb space cc00 for other devices, you will need
to change this line to another umb space

;

CISWindowBase=cc00

TokenRingSpeed=4

CardSettleTime=660

[DEFAULT.Resources]

include_mem=0xcf-0xcf
```

AUTOEXEC.BAT

Append the following information to the **autoexec.bat** file:

```
PATH = C:\PCM;%Path%
```

CONFIG.SYS

A typical **config.sys** file is illustrated below. It is used as a reference for Socket Services and the PCMCIA slots. Refer to **config.pcm** in the SDK\PCM directory.

```
DEVICE=DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE M5 X=CCOO-CDFF
DOS=HIGH,UMB
FILES=20
DEVICE=POWER.EXE
DEVICE=C:\PCM\DPMS.EXE
DEVICE=C:\PCM\CNFIGNAM.EXE /DEFAULT
DEVICE=C:\PCM\PCMSSIT.EXE
DEVICE=C:\PCM\PCMCS95FUL.EXE
DEVICE=C:\PCM\PCMSCD.EXE
DEVICE=C:\PCM\PCMATA.SYS
```

Miscellaneous Utilities

The PTC-2124 supports the following utilities:

- **GETID.COM**
- **ROWMGR.COM**
- **DOS INTERLNK.EXE**
- **TXRX.EXE**
- **WHICH.BAT**
- **USERAPP.BAT**

This section provides information on these utilities.

GETID.COM

This utility returns hardware IDs based on the request on the command line. Only one request can be issued per call. Command line options are as follows:

Command Option	Return Code Type	Return Code	Return Code Explanation
getid /s	SCANNER ID	1	se1200, se1200h, or se900 Scanner
		3	M2000 Scanner
		5	se1200, se1200h, or se900 Scanner
		6	se1200, se1200h, or se900 Scanner
		11	M2000 Scanner
getid /r	RADIO ID	3	IRM900 or 690 radio
		5	3500 radio
		6	2500 or 4500 radio
getid /v	VIDEO ID	0	Transflective Monochrome LCD (640 x 480)
		1	Transmissive Monochrome LCD (640 x 480)
		2	Electroluminescent Panel (640 x 480)
		3	Color TFT LCD (640 x 480)
		4 – 7	TBD
getid /t	ASIC ID	23	T130 ASIC
		3	T506 ASIC

Command Option	Return Code Type	Return Code	Return Code Explanation
getid /b	BOARD ID	16	PTC-2134
		17	PTC-2124
		18	PTC-960M
		19	PTC-870IM
getid /h	KEYBOARD ID (for PTC-960M)	59	27-key keyboard
		95	41-key keyboard
		47	53-key keyboard
getid /?	HELP	N/A	Lists all parameters and the type of return code that each parameter will give.

ROWMGR.COM Version 1.0

Rowmgr.com Overview

Rowmgr.com is a Symbol DOS utility that allows applicable Symbol products to customize the number of video rows on screen. Because of the proprietary nature of Symbol's computer systems and video architecture, it became apparent that a video row utility program would be needed to allow the user to work more efficiently at the DOS command line prompt.

Rowmgr.com is also a (TSR) Terminate and Stay Resident program. In the case of this utility, this means that the program stays in memory and ensures that after the user exits a program, the video row parameter will remain unchanged. This serves to eliminate any cursor confusion at the command line prompt.

Although most programs do a sufficient job of saving and restoring environment variables, the video row variable is often left unchecked and can cause cursor confusion for the user at the DOS command line prompt. **Rowmgr.com** resolves this by allowing the user to return to the DOS command line upon exiting a program and to customize the video row parameters after the program has been initially loaded. **Rowmgr.com** does this without using any additional conventional memory.

Command Line Syntax

Rowmgr.com command line syntax is common among industry standard DOS utility programs. A list of **rowmgr.com** parameters are shown below.

/nn Indicates number of rows to be displayed. Valid values are 10 – 25.

/nnr Change number of video rows after Row MGR TSR has been loaded.

/u Uninstall **rowmgr.com**.

> ***Note:*** *This feature is not available in* **RowMGR.com** *version 1.0*

/p A brief text description of **rowmgr.com**.

/? List available options.

Installation

Rowmgr.com was designed to load in the **autoexec.bat** file but can be loaded from the command line prompt. See below for loading instructions:

1. Edit the **autoexec.bat** file.
2. Add the line **RowMGR/15**.
3. **Save** and **Exit**.
4. **Reboot** the system.

Once **rowmgr.com** has been installed properly, the user can then change the number of video rows at any time via the command line prompt.

DOS INTERLNK.EXE

Overview

DOS Interlnk is a standard DOS utility that provides the user the ability to use one computer to access data and run programs on another computer.

The computer used to type commands is called the *client*. This is the PTC unit. The computer connected to the client is the *server*. This is usually a desktop PC. The client uses the server's drives and printers, and the server displays the status of the connection between the computers.

When the computers are connected using Interlnk, the drives on the desktop PC (the server) appear as additional drives on the PTC unit (the client).

Requirements

To use Interlnk, the following items are needed:

- A free serial port on both computers or a free parallel port on both computers.
- A null-modem serial cable.
- MS-DOS version 6.0.
- 16 K of free memory on the client and 130 K of free memory on the server.

Configuring the Client

To set up **interlnk.exe** on the computer that is to be used as the client, these steps should be followed:

1. Verify the existence of the **interlnk.exe** file on the computers' hard disk. This file is located in the DOS directory.

2. Open the **config.sys** file by using a text editor such as MS-DOS Editor.

3. Add a device command that specifies the location of the **interlnk.exe** file. Options for redirecting drives and printers can also be specified. For more information about these options, type **help interlnk.exe** at the command prompt.

   The following example specifies that the **interlnk.exe** file is located in the DOS directory on drive C. The command also specifies that **interlnk.exe** should redirect five drives instead of the default three drives:

   device=c:\dos\interlnk.exe/drives:5.

4. Save the changes to the **config.sys** file, then quit the text editor.

5. Restart the PTC unit by pressing **CTRL**+**ALT**+**DEL** or the Symbol Reset button.

After the device command for **interlnk.exe** has been added to the **config.sys** file, Interlnk displays the status of redirected drives and ports each time the user starts the client computer.

The status of redirected drives and ports can also be viewed by typing **interlnk** at the command prompt.

For more information about the **interlnk.exe** device driver, type **help interlnk.exe** at the command prompt.

Starting the Server

No changes need to be made to the **config.sys** file to start the Interlnk server. To start the server, type **intersvr** at the command prompt of the desktop PC.

Interlnk will then display two columns of information about redirected drives and printer ports on the server's screen:

- The column labeled "This Computer" lists all drives and printer ports on the server.
- The column labeled "Other Computer" lists the drives and printer ports on the client, in addition to the drives and ports on the server that are now available on the client.

A status bar at the bottom of the screen displays information about the Interlnk connection using the following fields:

- The Transfer field indicates whether the client is reading from or writing to the server. When the client reads from or writes to the server, Interlnk displays an asterisk (*****) next to the drive letter that indicates which server drive is being affected.
- The Port field shows which server port Interlnk is using to connect to the client.
- The Speed field shows the baud rate at which Interlnk is transferring information.

For more information about the Interlnk server, type **help intersvr** at the command prompt.

Establishing a Connection Between Client and Server

Interlnk establishes connections between all redirected drives and ports when one of the following actions is performed:

- The client computer is restarted while the server is running.
- **interlnk** is typed at the command prompt of the client computer.
- One of the redirected drives on the client computer is made the active drive.

Breaking the Connection Between Client and Server

To break the Interlnk connection between the computers, stop the server by pressing **ALT+F4** on the server's keyboard. To restart the server, type **intersvr** at the server's command prompt.

TXRX.EXE

TXRX File Transfer Utility Overview

TXRX is a serial communications utility that will perform a XMODEM / YMODEM protocol file transfer. The transfer rate at which **TXRX** can communicate is anywhere from 1200–38400 baud. It can run on either PTCs, PBCs or PCs. TXRX can be either command, file, or menu-driven.

Using TXRX.EXE

Usage

Command line parameters can be in either upper or lower case. The prefix to the command can be either '-' or '/'.

TXRX [filename]

or

TXRX -C# -B##### -O -N -I - Y

Where:

filename = the configuration file name to use. If no file extension is used, .DAT will be assumed as the extension.

-C Is the communications port to use. The # can be either 1, 2, 3, or 4. Default is 1.

-B Is the baud-rate to use. The ##### can be either 9600, 19200, or 38400. If a value of 0 is used, then the baud rate will be left alone. It will be up to an external program to set it prior to running **TXRX**.

-O This will enable the optics port on capable PTCs.

-N This will keep **TXRX** from displaying its transfer progress (this option is needed for a DR3092).

- I This will display extra file progress statistics.

-Y When this option is used while sending data, 1024 byte blocks will be used for the transfer.

If **TXRX** is invoked without the optional filename, the program will be menu driven. After a file has been transferred, the program will return to the main menu to either transfer another file or exit the program. When certain command line options such as **-B**, **-C**, or **-Y** are used, **TXRX** will not prompt the user for their options.

If a filename is given in the command line, all information needed to transfer file(s) is taken from this configuration file, requiring no user intervention. When all file(s) have been transferred, the program will exit with a return code to DOS.

DOS Return Codes

DOS return codes are errorlevels that are returned to the DOS environment once a program terminates. The errorlevel can be tested in a batch file or through the 'C' call spawn.

OK	0	Successful Acknowledgement.
TIMEOUT_ER	2	Not acknowledged.
CAN_ER	3	Other cpu cancelled.
USER_ABORT	4	User aborted.
FOPEN	5	Error opening file.
FIO	6	File error during I/O rd/wt.
CONFIG	7	Configuration file error.
SYNC_ER	8	Protocol out of sync.
NO_ACK	9	Timed out waiting for ack.
RETRY_ER	10	Too many retries.
NO_EOT	11	End of Xmit not received.
CHKSUM_ER	12	Block does not match chksum.
SHORT_BLK_ER	13	A short block was received.
DISK_FULL_ER	14	The disk if full.

Execution — Menu Driven

If the program is being menu driven (no configuration file specified in the command line), the following main menu will appear:

XYmodem - Ver #.#

Connect Optics? (PTC ONLY)

Y or N (PTC ONLY)

0:Receive file

1:Transmit file

2:Exit

Selecting Option 2 will return to the DOS prompt. If Options 0 or 1 are selected, the next menu appears:

Select Baud Rate

0: 38400 Baud

1: 19200 Baud

2: 9600 Baud

The same baud rate should be selected on both the transmitting and the receiving computer. Once the baud rate has been selected, the program will prompt with:

(If transmitting:)	(If receiving:)
Enter Protocol	
0: Xmodem	
1: Ymodem	
Xm Transmit	Xm Receive
Enter filename:	Enter filename:

If running on a PTC, all file access will be assumed on A:. Once the filename has been given, one of the following messages will appear:

(If transmitting:)	(If receiving:)
To cancel, Press **ESC**	To cancel, Press **ESC**
Sent = 0000	Rcvd = 0000

If the -I command line option is being used, the following status will be displayed:

(If transmitting:)	(If receiving:)
To cancel, Press **ESC**	To cancel, Press **ESC**
Sent=0000 ST:## RE:##	Rcvd=0000 ST:## RE:##

Where:

ST:## Is the current status of the last block received. This will be a 1 or 2 digit code. Refer to "DOS Return Codes" on page 173 for a description of the status codes.

RE:## Is the total retries that were encountered during the file transfer.

It may take a few seconds for the two computers to connect. When they do, transmission will begin and the block count (128 byte/block for XMODEM and 1024 byte/block for YMODEM) will be updated continuously. If connection fails, the program will time out. In either case, a final status message is displayed and the program will return to the main menu. Once transmission has started, the **ESC** key may be used to interrupt execution.

Execution — Using Configuration File

If many files are to be transmitted, the user may wish to use a configuration data file. This file specifies either send or receive, the baud rate, and a list of files to be transferred. If no extension is given for the configuration file in the command line, the program will automatically look for that filename with a .DAT extension.

When all files are transmitted successfully, the program will exit. If an error occurs during transferring one of the files, the program will terminate with a return code to DOS. No other files will be processed.

The configuration file must be of the following format:

operation baud

filename1

filename2

filenameN

Where:

Operation is the string “SEND” or “RECV.” *Baud* is either1200, 2400, 4800, 9600, 19200, or 38400.

Filename1 through filenameN is the list of files to be transferred. There must be one space between operation and baud.

As each file is being processed, the following will be displayed:

(If transmitting:)	(If receiving:)
filenameN	filenameN
To cancel, Press **ESC**	To cancel, Press **ESC**
Sent=0000	Rcvd=0000

The block count will be displayed continually until the file is completely transferred.

If a configuration file is being used, then it is a good idea to use a configuration file on both computers. Otherwise, one side might time out while waiting for the information to be entered on the other side.

Cable Requirements

The cable requirement is a NULL modem cable between the serial devices. The bare minimum NULL modem cable configuration is as follows:

PTC-2124	
9-Pin Connector	
Pin	**Assignment**
3	TXD
2	RXD
5	GND
9-Pin Connector	
Pin	**Assignment**
3	TXD
2	RXD
5	GND

PC	
25-Pin Connector	
Pin	**Assignment**
3	RXD
2	TXD
7	GND
9-Pin Connector	
Pin	**Assignment**
2	RXD
3	TXD
5	GND

TXD - Transmit Data

RXD - Receive Data

GND - Signal Ground

Note: *Any other style of NULL modem cable should work. The above shows the minimal configuration needed for TXRX version 3.2 and greater.*

WHICH.BAT

WHICH.BAT uses Symbol utilities **getid.com** and **radioid.com**. These programs determine the proper pen unit panel installed. **Which.bat** sets the proper environment variable for the TMOUSE and then loads TMOUSE. **Which.bat** components must first be uncommented to take effect.

> ***Note:*** *The new pen calibration settings do not take effect when running Pencal /H (invoked when running **which.bat**) after boot up.*

USERAPP.BAT

Userapp.bat is a utility that allows user installed applications to run. To have the PTC-2124 automatically run a user installed application (i.e., PenRight! application), a file named **userapp.bat** should be created in the root directory on the unit's C drive. Once all necessary drivers have been loaded at boot up, this batch file will be executed.

Accessories

A list of PTC-2124 accessories, and their part numbers, may be found in the ***PTC-2124 User's Guide*** or obtained from a Symbol Sales Representative.

References

The following are additional reference materials available for the PTC-2124:

- **PTC-2124 User's Guide**
  (Part Number: 30833-000-002)
- **SC400 DOS SDK Programming Guide**
  (Part Number: 30419-000-01)
- **2X34 SC/VC User's Guide**
  (Part Number: 24907-701-01)
- **1124 Connector Pod Read-Me-First Sheet**
  (Part Number: 24830-701-01)

Note: *Reference guides may be downloaded in the PDF format from Symbol's website at* ***www.symbol.com****.*

Index

Symbols

Numerics

A

B

C

D

E

S

T

U

V

W

Symbol Technologies, Inc.

One Symbol Plaza

Holtsville, NY 11742-1300

(Document No: 30722-000-002)

Printed in the U.S.A.