

Published: June 2006

Author: Infogile Inc.

Most personal computers today are equipped with a parallel port, commonly used to connect the computer to a parallel printer. Because it is available on most personal computers, the parallel port is a perfect choice for connection to other peripheral devices. However, communication to peripherals across the parallel port is limited because the interface is traditionally unidirectional and there is no standard specification for the interface. Additionally, although the performance of the PC has dramatically increased, the parallel port has remained the same. This situation has led to the development of a new parallel port standard – IEEE Standard 1284-1994.

Introduction

From its humble beginnings as a Printer Port on the original IBM PC, the Parallel Port has become one of the most common connectors on the Personal Computer today. Virtually every PC shipped over the past 18 years has had a parallel port interface on it. Originally this port was used just for driving 'fast' dot matrix printers. Today the parallel port is used to interface to almost every PC peripheral in use. Printers, scanners, data acquisition, CD ROM, tape drives, high speed modems and digital cameras are some of the peripherals that can be purchased and used today.

From 1981 until 1994 there was tremendous growth in the power, performance and capabilities of the PC. Unfortunately there was very little change in the performance of the parallel port. In 1994 this all changed with the release of IEEE Std. 1284-1994. This standard defined new protocols and new interface models that would enable a 50 to 100x performance increase in the capabilities of the parallel port. It can do this and still be fully backward compatible with all existing parallel port peripherals and printers. With the efforts of the IEEE 1284.3 and 1284.4 committees these enhancements are continuing today. IEEE 1284 is for the parallel port what the Pentium processor is to the 286.

White Paper Overview

Abstract

This white paper talks about Advances in High-Speed Parallel Port Performance and Port Sharing. It traces the development of parallel ports from traditional printer ports to IEEE 1284 standard. It talks about the data transfer modes in 1284, the on going activities in this field and an introduction to the state and future of the parallel port interface and the associated protocol stack.

Business Situation

Every new technology becomes legacy technology at some point in its life. Most current computer systems contain a combination of new and old technologies. Current legacy technologies include computer buses, interfaces, peripheral devices, and I/O ports that were designed in the early era of PC standards development, and which have not evolved sufficiently to take advantage of the capabilities of today's more manageable, faster, and far more efficient PC architectures. Reducing the impact of legacy technologies on current system architectures is the goal of several industry initiatives that target legacy components for removal.

Benefits of 1284

- This interface provides the capability to send data from the host computer to the peripheral at a higher speed than previously done.
- It provides a path for data to be sent from the peripheral to the host using existing and future parallel port hardware
- The reverse channel capability reduces the amount of user interaction needed to operate the peripheral.

This paper discusses the features in IEEE 1284 that distinguish it from parallel ports, the data transfer modes in this standard and the on going research activities in the field of parallel ports.

What are Parallel Ports

Parallel ports are personal computer interfaces that transfer data (generally) a byte at a time. This contrasts with serial ports that transfer data one bit at a time. When IBM introduced the PC, in 1981, the parallel printer port was included as an alternative to the slower serial port as a means for driving the latest high performance dot matrix printers. The parallel port had the capability to transfer 8 bits of data at time whereas the serial port transmitted one bit at a time. When the PC was introduced, dot matrix printers were the main peripheral that used the parallel port. As technology progressed and the need for greater external connectivity increased, the parallel port became the means by which you could connect higher performance peripherals. These peripherals now range from printer sharing devices, portable disk drives and tape backup to local area network adapters and CD ROM players.

The parallel port, as implemented on the PC, consists of a connector with 17 signal lines and 8 ground lines. The signal lines are divided into three groups:

- Control (4 lines)
- Status (5 lines)
- Data (8 lines)

As originally designed, the Control lines are used as interface control and handshaking signals from the PC to the printer. The Status lines are used for handshake signals and as status indicators for such things as paper empty, busy indication and interface or peripheral errors. The data lines are used to provide data from the PC to the printer, in that direction only. Later implementations of the parallel port allowed for data to be driven from the peripheral to the PC.

Table 1 identifies each of these signals and gives their Standard Parallel Port (SPP) definitions. The signals within these groups are assigned to specific bits within the registers that make up the hardware/software interface to the parallel port. The parallel port is mapped into the I/O space of the PC. The registers consist as a contiguous block of 3 registers starting from the parallel port's base address. These ports are commonly referred to as the LPT ports and have the familiar I/O base addresses of 3BCh, 378h and 278h. Newer implementations of the parallel port, that support the advanced modes of the 1284 standard, use 8 to 16 registers and are located at I/O addresses 378h or 278h, or are re-locatable, as in the case of a Plug and Play compliant parallel adapter.



Fig. 1

Table 1 -- SPP Signal Definitions

Group	SPP Signal	In/Out	Signal Description
Control	nSTROBE	Out	Active low. Indicates valid data is on the data lines.
	nAUTOFEED	Out	Active low. Instructs the printer to automatically insert a line feed for each carriage return
	nSELECTIN	Out	Active low. Used to indicate to the printer that it is selected.
	nINIT	Out	Active low. Used to reset the printer.
Status	nACK	In	A low asserted pulse used to indicate that the last character was received.
	BUSY	In	A high signal asserted by the printer to indicate that it is busy and cannot take data.
	PE	In	Paper Empty
	SELECT	In	Asserted high to indicate that the printer is online.
	nERROR	In	Asserted low to indicate that some error condition exists
Data	DATA[8:1]	Out	8 data lines- output only in older SPP

Why IEEE-1284

The problems faced by developers and customers of these peripherals fall into three categories. First, although the performance of the PC has increased dramatically, there has been virtually no change in the parallel port performance or architecture. The maximum data transfer rate achievable with this architecture is around 150 kilobytes per second and is extremely software intensive. Second, there is no standard for the electrical interface. This causes many problems when attempting to guarantee operation across various platforms. Finally, the lack of design standards forced a distance limitation of only 6 feet for external cables

In 1991 there was a meeting of printer manufacturers to start discussions on developing a new standard for the intelligent control of printers over a network. These manufacturers, which included Lexmark, IBM, Texas Instruments and others, formed the Network Printing Alliance. The NPA submitted a proposal to the IEEE for the creation of a committee to develop a new standard for a high speed bi-directional parallel port for the PC. It was a requirement that this new standard would remain fully compatible with the original parallel port software and peripherals, but would increase the data rate capability to greater than 1M bytes per second, both in and out of the computer. This committee became the IEEE 1284 committee.

The IEEE 1284 standard, "Standard Signaling Method for a Bi-directional Parallel Peripheral Interface for Personal Computers", was approved for final release in March of 1994.

Data Transfer Modes in IEEE-1284

The 1284 standard defines 5 modes of data transfer. Each mode provides a method of transferring data in either the forward direction (PC to peripheral), reverse direction (peripheral to PC) or bi-directional data transfer (half duplex). The defined modes are:

Compatibility Mode



This mode defines the protocol used by most PCs to transfer data to a printer. It is commonly called the "Centronics" mode and is the method utilized with the standard parallel port. In this mode, data is placed on the port's data lines, the printer status is checked for no errors and that it is not Busy, and then a data Strobe is generated by the software to clock the data to the printer. Figure 2 describes this transfer.

As can be seen, in order to output one byte of data it requires four I/O instructions and at least as many additional instructions. The net effect of this is a limitation on the bandwidth capabilities of the port on the order of 150K bytes per second. This bandwidth is sufficient for communicating with dot matrix and many older laser printers, but a limitation when communicating with pocket LAN adapters, removable disk drives, and the newest generation of laser printers, to name a few. Of course, this mode is for the forward channel only and must be combined with a reverse channel

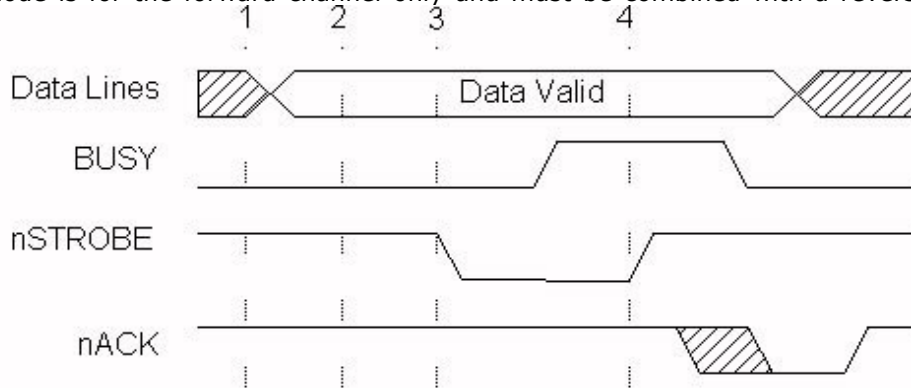


Figure 2 -- Compatibility Mode Data Transfer Cycle

mode in order to have a complete bi-directional channel. This mode was included as a way to provide backward compatibility with the huge base of installed printers and peripherals. The other modes are used to provide the reverse channel and high performance communication links. Many of the integrated 1284 I/O controllers have implemented a mode that uses a FIFO buffer to transfer data with the Compatibility mode protocol. This mode is referred to as "Fast Centronics" or "Parallel Port FIFO Mode". When this mode is enabled, data written to the FIFO port will be transferred to the printer using hardware generated strobes for the handshaking. Since there is very little latency between transfers, and the software does not have to do any of the strobing or handshake checking, data rates over 500K bytes per second are achievable with some systems. This mode, however, is not an IEEE 1284 defined mode and is not described in the standard.

Nibble Mode

The Nibble mode is the most common way to get reverse channel data from a printer or peripheral. This mode is usually combined with the Compatibility mode or a proprietary forward channel mode to create a complete bi-directional channel. All of the standard parallel ports provide 5 lines from the peripheral to the PC to be used for external status indications. Using these lines, a peripheral can send a byte of data (8-bits) by sending 2 nibbles (4-bits) of information to the PC in two data transfer cycles. Unfortunately, since the nACK line is generally used to provide a peripheral interrupt, the bits used to transfer a nibble are not conveniently packed into the byte defined by the Status register. For this reason, the software must read the status byte and then manipulate the bits in order to get a correct byte. Table 2 identifies the signal names for the Nibble mode



Table 2 -- Nibble Mode Signals

SPP Signal	Nibble Mode Name	In/Out	Description -- Signal usage when in Nibble Mode data transfer
nSTROBE	nSTROBE	Out	Not used for reverse data transfer
nAUTOFEED	HostBusy	Out	Host nibble mode handshake signal. Set low to indicate host is ready for nibble. Set high to indicate nibble has been received.
nSELECTIN	1284Active	Out	Set high when host is in a 1284 transfer mode.
nINIT	nINIT	Out	Not used for reverse data transfer
nACK	PtrClk	In	Set low to indicate valid nibble data, set high in response to HostBusy going high.
BUSY	PtrBusy	In	Used for Data bit 3, then 7
PE	AckDataReq	In	Used for Data bit 2, then 6
SELECT	Xflag	In	Used for Data bit 1, then 5
nERROR	nDataAvail	In	Used for Data bit 0, then 4
DATA[8:1]	Not Used		

Nibble mode, like the Compatible mode, requires that the software drive the protocol by setting and reading the lines on the parallel port. Nibble mode is the most software intensive mode for reverse channel data communication. For this reason, there is a severe limitation of approximately 50K bytes per second for this type of data transfer. The major advantage of this combination of modes is the ability to operate on all PCs that have a parallel port. The performance limitations incurred by the Nibble mode operation does not have much visible effect on peripherals that have low reverse channel requirements, such as printers, but can be nearly intolerable when used for other devices.

Byte Mode

With later implementations of the parallel port interface, some manufacturers, led by IBM on the PS/2 parallel port, added the capability to disable the drivers used for driving the data lines, and allowed the data port to become an input read data port. This enables a peripheral to send an entire byte of data to the PC in one data transfer cycle by using the 8 data lines, rather than the two cycles required using the Nibble mode. This ability enables a Byte mode for reverse channel data transfer that can be used to provide data rates into the PC approaching that of the Compatibility mode, from the PC.

Table 3 -- Byte Mode Signals

SPP Signal	Byte Mode Name	In/Out	Description Signal usage when in Byte Mode data transfer
nSTROBE	HostClk	Out	Pulsed low at the end of each Byte mode data transfer to indicate that the byte was received. Acknowledge signal.
nAUTOFEED	HostBusy	Out	Set low to indicate host is ready for byte. Set high to indicate byte has been received. Handshake signal.
nSELECTIN	1284Active	Out	Set high when host is in a 1284 transfer mode.
nINIT	nINIT	Out	Not used. Set high.
nACK	PtrClk	In	Set low to indicate valid data on the data lines, set high in response to HostBusy going high.

BUSY	PtrBusy	In	Forward channel Busy status.
PE	AckDataReq	In	Follows nDataAvail
SELECT	Xflag	In	Extensibility flag. Not used in Byte mode.
nERROR	nDataAvail	In	Set low by peripheral to indicate that reverse data is available.
DATA[8:]	DATA[8:1]	Bi-Di	Used to provide data from peripheral to host.

EPP Mode

The Enhanced Parallel Port protocol was originally developed by Intel, Xircom and Zenith Data Systems, as a means to provide a high performance parallel port link that would still be compatible with the standard parallel port. This protocol capability was implemented by Intel in the 386SL chipset (82360 I/O chip). This was prior to the establishment of the IEEE 1284 committee and the associated standards work.

The EPP protocol offered many advantages to parallel port peripheral manufacturers and was quickly adopted by many as an optional data transfer method. A loose association of around 80 interested manufacturers was formed to develop and promote the EPP protocol. This association became the EPP Committee and was instrumental in helping to get this protocol adopted as one of the IEEE 1284 advanced modes. Since EPP capable parallel ports were available prior to the release of the 1284 standard, there is a small deviation between the pre-1284 EPP ports and 1284 EPP protocol.

Table 4 -- EPP Signal Definitions

SPP Signal	EPP Signal Name	EPP Signal Description	
nSTROBE	nWRITE	Out	Active low. Indicates a write operation High for a read cycle.
nAUTOFEED	nDATASTB	Out	Active low. Indicates a Data_Read or Data_Write operation is in process.
nSELECTIN	nADDRSTB	Out	Active low. Indicates an Address_Read or Address_Write operation is in process.
nINIT	nRESET	Out	Active low. Peripheral reset.
nACK	nINTR	In	Peripheral interrupt. Used to generate an interrupt to the host.
BUSY	nWAIT	In	Handshake signal. When low it indicates that is OK to start a cycle (assert a strobe), when high it indicates that it is OK to end the cycle (de-assert a strobe).
D[8:1]	AD[8:1]	Bi-Di	Bi-directional address/data lines.
PE	user defined	In	Can be used differently by each peripheral
SELECT	user defined	In	Can be used differently by each peripheral.
nERROR	user defined	In	Can be used differently by each peripheral.

One of the most important features to note here is that the entire data transfer occurs within one ISA I/O cycle. The effect is that by using the EPP protocol for data transfer, a system can achieve transfer rates from 500K to 2M bytes per second. In this fashion, parallel port peripherals can operate at close to the same performance levels as an equivalent ISA plug-in card. The ability to get this level of performance from a parallel port attached device is one of the major features of the EPP protocol. With interlocked handshakes, the data transfer will happen at the speed of the slowest of



the interfaces, the host adapter or the peripheral device. This "speed adaptive" property is transparent to both the host and peripheral. All 1284 transfer modes are implemented with interlocking handshakes.

The simplest software view of EPP is that of an extension to the register definitions for the standard parallel port. As shown earlier, the SPP consists of three registers, offset from the port's base address: Data port, Status port, and Control port. The most common EPP implementations expand this to use ports not defined by the SPP. See table 5.

Table 5 EPP Register Definitions

Port Name	Offset	Mode	Read/Write	Description
SPP Data Port	+0	SPP/EPP	W	Standard SPP data port. No autostrobing.
SPP Status Port	+1	SPP/EPP	R	Reads the input status lines on the interface.
SPP Control Port	+2	SPP/EPP	W	Sets the state of the output control lines.
EPP Address Port	+3	EPP	R/W	Generates an interlocked address read or write cycle.
EPP Data Port	+4	EPP	R/W	Generates an interlocked data read or write cycle.
Not Defined	+5 to +7	EPP	N/A	Used differently by various implementations. May be used for 16 and 32 bit I/O.

By generating a single I/O write instruction to "base_address + 4", the EPP controller will generate the necessary handshake signals and strobes to transfer the data using an EPP Data_Write cycle. I/O instructions to the base addresses, ports 0 through 2, will cause behavior exactly as that as to a standard parallel port. This guarantees compatibility with standard parallel port peripherals and printers. Address cycles are generated when read or write I/O operations are generated to "base_address + 3".

Ports 5 through 7 are used differently by various hardware implementations. These may be used to implement 16 or 32-bit software interfaces, or used for configuration registers, or not used at all. For example, the Warp Nine Engineering's F/PortPlus card has only an 8-bit data interface, but can be accessed using 32-bit I/O, for EPP data operations. The ISA controller will intercept the 32-bit I/O and actually generate 4 fast 8-bit I/O cycles. The first cycle will be to the addressed I/O port using byte 0 (bits 0-7), the second cycle will be to port+1 using byte 1, then port+2 using byte 2 and finally port+3 using byte 3. These additional cycles are generated by hardware and are transparent to the software. The total time for these four cycles will be less than 4 independent 8 bit cycles. For example, the F/PortPlus card (from Warp Nine Engineering) maps 4 I/O ports (offset 4 to 7) to the internal EPP Data register. This enables the software to use 32-bit I/O operations for EPP data transfer. Address cycles are still limited to 8-bit I/O.

The ability to transfer data to or from the PC by the use of a single instruction is what enables EPP mode parallel ports to transfer data at ISA bus speeds. Rather than having the software implement an I/O intensive software loop, a block of data can be transferred with a single REP_IO instruction. Depending upon the host adapter port implementation and the capability of the peripheral, an EPP port can transfer data from 500K bytes to nearly 2M bytes per second. This data transfer rate is more than enough to enable network adapters, CD ROM, tape backup and other peripherals to operate at nearly ISA bus performance levels.



The EPP protocol and current implementations provide a high degree of coupling between the peripheral driver and the peripheral. What this means is the software driver is always able to determine and control the state of communication to the peripheral at any given time. Intermixing of read and write operations as well as block transfers are readily done. This type of coupling is ideal for many register-oriented or real-time controlled peripherals such as network adapters, data acquisition, portable hard drives, and other devices.

ECP Mode

The Extended Capability Port, or ECP, protocol was proposed by Hewlett Packard and Microsoft as an advanced mode for communication with printer and scanner type peripherals. Like the EPP protocol, ECP provides for a high performance bi-directional communication path between the host adapter and the peripheral.

Unlike EPP, when the ECP protocol was proposed, a standard register implementation was also proposed. This can be found in the Microsoft document "The IEEE 1284 Extended Capabilities Port Protocol and ISA Interface Standard" available from Microsoft Corp. This document defines features that are implementation specific which the IEEE 1284 standard could not address. These features include Run_Length_Encoding (RLE) data compression for the host adapters, FIFOs for both the forward and reverse channels, and DMA as well as programmed I/O for the host register interface.

The RLE feature enables real time data compression that can achieve compression ratios up to 64:1. This is particularly useful for printers and scanners that are transferring large raster images that have large strings of identical data. In order for the RLE mode to be enabled both the host and the peripheral must support it.

Channel addressing is, conceptually, a little different than the EPP addressing. Channel addressing is intended to be used to address multiple logical devices within a single physical device. Think of this in terms of a new multi-function device such as FAX/Printer/Modem. Within one physical package, having a single parallel port attached, there is a printer, fax and modem. Each of these separate functions can be thought of as separate logical devices within the same package. Using the ECP channel addressing to access each of these devices, you could receive data from the modem data device while the printer data channel is busy processing a print image. With the compatibility mode protocol, if the printer gets busy then no more communication can occur until the printer data channel is free. With ECP, the software driver simply addresses another channel and communication can continue.

Table 6 -- ECP Mode Signals

SPP Signal	ECP Mode Name	In/Out	Description -- Signal usage when in ECP Mode data transfer
nSTROBE	HostClk	Out	Used with PeriphAck to transfer data or address information in forward direction.
nAUTOFEED	HostAck	Out	Provides Command/Data status in the forward direction. Used PeriphClk to transfer data in the reverse direction.
nSELECTIN	1284Active	Out	Set high when host is in a 1284 transfer mode.
nINIT	nReverseRequest	Out	Driven low to put the channel in reverse direction.
nACK	PeriphClk	In	Used with HostAck to transfer data in the reverse direction.
BUSY	PeriphAck	In	Used with HostClk to transfer data or address information in the forward direction. Provides Command/Data status in the reverse direction.



PE	nAckReverse	In	Driven low to acknowledge nReverseRequest.
SELECT	Xflag	In	Extensibility flag.
nERROR	nPeriphRequest	In	Set low by peripheral to indicate that reverse data is available.
Data[8:1]	Data[8:1]	Bi-Di	Used to provide data between the peripheral and host.

The Microsoft specification, "The IEEE 1284 Extended Capabilities Port Protocol and ISA Interface Standard", defines a common register interface for ISA based 1284 adapters with ECP. This specification also defines a number of operational modes that the adapter can operate under. Table 7 identifies these modes.

Table 7 -- ECR Register Modes

Mode	Description
000	SPP mode
001	Bi-directional mode (Byte mode)
010	Fast Centronics
011	ECP Parallel Port mode
100	EPP Parallel Port mode ^(note 1)
101	(reserved)
110	Test mode
111	Configuration mode

This mode is implemented by the SMC FDC37C665/666 controller and is not defined by the ECP specification. Most 1284 I/O controllers implement the EPP mode in a similar fashion.

The register model for an ECP port is similar to that of the standard parallel port, but takes advantage of a significant design oddity of the ISA bus architecture. In the IBM compatible PC architecture, only the first 1024 I/O ports, or addresses, are used. This is the basic I/O space of 0x000h to 0x3ffh. In order to address this range of addresses, only 10 address bits are needed (AD0-9). In order to save cost, older PC's only drove and decoded these address bits on the ISA bus and therefore limited the available I/O space to these 1024 registers. Newer PC's, actually drive and decode more address bits, enabling a larger available I/O space. This creates, in effect, multiple "pages" of the first 1K I/O ports. A software driver can access these pages by adding 1024 (0x400h hex notation) to the base address being accessed. Therefore, addressing addresses 0x378h and 0x778h can access 2 registers on two separate pages, but be guaranteed not to interfere with any other installed ISA device. The advantage is that by using this "aliasing" effect, new cards can have "hidden" registers, thus expanding the available number of registers available, and still maintain compatibility with older ISA cards that only decode 10 address bits.

The ECP register model takes advantage of this and defines 6 registers that only require 3 actual I/O ports. Table 8 identifies these registers. Note that the register definition may be dependent upon the current mode of operation. The ECR register is the most important aspect of this register configuration. This is the register that is used to set the current operational mode. Additionally, this register can be used by software to determine if an ECP-capable port is installed in the PC. Detection software can try to access any ECR registers by adding 0x402h to the base address of the LPT ports identified in the BIOS LPT port table.

Table 8 -- ECP Register Description

Offset	Name	Read/Write	ECP Mode	Function
--------	------	------------	----------	----------



000	Data	R/W	000-001	Data Register
000	ecpAfifo	R/W	011	ECP Address FIFO
001	dsr	R/W	all	Status Register
002	dcr	R/W	all	Control Register
400	cFifo	R/W	010	Parallel Port Data FIFO
400	ecpDfifo	R/W	011	ECP Data FIFO
400	tfifo	R/W	110	Test FIFO
400	cnfgA	R	111	Configuration Register A
401	cnfgB	R/W	111	Configuration Register B
402	ecr	R/W	all	Extended Control Register

This paper will not attempt to describe all the functions of the ECP registers. For information regarding the register usage and bit definitions, please refer to the Microsoft document or the I/O controller data sheet.

It should be noted that if the port is in either standard parallel port mode or bi-directional mode, then the first 3 registers behave exactly as a standard parallel port. This way, compatibility with older devices and device drivers is maintained.

Usage of this port is similar to that of the EPP port. An operational mode is written to the ecr register, and then data transfer is accomplished by writing or reading from the appropriate I/O port. All of the handshaking is automatically generated by the interface controller. The major difference is that the ECP port is meant to be driven by DMA rather than explicit I/O operations. Again, this is a loosely coupled interface that is targeted primarily at peripherals that interchange large blocks of data.

All parallel ports can implement a bi-directional link by using the Compatible and Nibble modes for data transfer. Byte mode can be utilized by about 25% of the installed base of parallel ports. All three of these modes utilize software only to transfer the data. The driver has to write the data, check the handshake lines (i.e.: BUSY), assert the appropriate control signals (i.e.: STROBE) and then go on to the next byte. This is very software intensive and limits the effective data transfer rate to 50 to 100 Kbytes per second.

In addition to the previous 3 modes, EPP and ECP are being implemented on the latest I/O controllers by most of the Super I/O chip manufacturers. These modes use hardware to assist in the data transfer. For example, in EPP mode, a byte of data can be transferred to the peripheral by a simple OUT instruction. The I/O controller handles all the handshaking and data transfer to the peripheral.

IEEE 1284 Related Standard's Activities

The IEEE 1284-1994 standard was approved in March of 1994. At that time the 1284 committee was disbanded and is no longer active. The current 1284 "dot" committees are involved in developing NEW standards that are related to the 1284 standard, but will not replace or change the released 1284 standard.



Currently there are, or have been, four working groups developing new standards that create additional support for the 1284 interface. These are new standards and not revisions of the current IEEE 1284-1994 released standard. The exception to this is the P1284R committee.

IEEE P1284R

"Revision and update for IEEE Standard 1284-1994"

Every five years an IEEE standard needs to be reviewed. The outcome of this process is either: 1) Reaffirmation as a standard without changes; 2) Reaffirmation with revisions, editorial or functional; or 3) Discontinued. The current IEEE 1284-1994 is going through option 2 and being updated for editorial issues and content.

IEEE Std. 1284.1-1997

"Standard for Information Technology for Transport Independent Printer/Scanner Interface (TIP/SI)"

This committee is developing a standard for the control and management of printer and scanner related jobs. This work is based upon the NPAP protocol developed by the Network Printing Alliance (NPA).

IEEE P1284.2

"Standard for Test, Measurement and Conformance to IEEE Std. 1284"

This committee is working to develop a standard test suite that can be implemented to test and verify peripherals, host adapters and cable assemblies for conformance to the 1284 standard.

IEEE P1284.3

"Standard for Interface and Protocol Extensions to IEEE Std. 1284 Compliant Peripheral and Host Adapter Ports"

This committee is developing the necessary application and driver software interfaces required to implement drivers for 1284 compliant peripherals. This committee is developing a standard for sharing the parallel port in a daisy chained or switched (multiplexed) architecture. In addition, this standard will define a data link layer for the 1284 parallel port that will address port contention and data transfer.

IEEE P1284.4

"Standard for Data Delivery and Logical Channels for IEEE Std. 1284 Interfaces"

This committee is developing a Transport Protocol standard that may be used across an IEEE Std. 1284-1994 parallel port interface, or any other suitable physical interface such as USB or IEEE 1394. This standard will provide a protocol to create and maintain multiple conversations across a single parallel port link. Think of this as using the 1284 parallel port to communicate with a multifunction device having printer, scanner and fax capabilities. With this protocol multiple applications will have access to the individual functions without the knowledge of the other applications. The starting point for this standard is the Hewlett-Packard standard for Multiple Logical Channels (MLC). It should be noted that the goal of this committee is not to make a standard of the HP MLC specification, but to use the MLC spec as a starting point from which to meet the objectives standard. There is no guarantee, at



this time, that the final IEEE standard will be backwards compatible with the current implementation of HP's MLC.

Advances in High Speed Parallel Port Performance

The 1284 standard provided new data transfer protocols that could be implemented in an enhanced register model. One of the key advantages of this was that the register model could remain backward compatible with the existing model but add enhanced features for better data transfer. This was achieved by implementing hardware state machines that would offload the data transfer handshaking from the host driver. The Enhanced Parallel Port (EPP) and Extended Capabilities Port (ECP) are these advanced modes. These modes enable data transfer rates of over 1MB/S with reduced host utilization on today's ISA ports, and 3-5MB/S on future PCI implementations.

Printing, reading or writing from tape, and other peripheral communication can consume a significant percentage of the parallel port bandwidth when active. In reality though, these activities do not occur very often and use very little of the overall available bandwidth. This means that a significant PC resource is being wasted. To use this additional potential the IEEE 1284.3 committee developed a protocol to allow sharing of the parallel port by multiple peripherals. The Daisy Chain protocol is the result of this effort.

Devices that adhere to this protocol (DC devices) have two connectors on them. One connector connects to the host port of the PC or the Pass Through port of another DC device. The other connector is the pass through port. This port replicates, or passes through the signals from the host port when the peripheral is not communicating with the host. The Daisy Chain protocol allows four peripherals to share a single parallel port and still attach an older 'legacy' peripheral. In this way, like the 1284 standard, backwards compatibility with the existing installed base is preserved.

The wire protocol that is used for 1284.3 is "out of band" to the 1284 defined protocols. This allows all DC devices to utilize the full capabilities of the 1284 standard. The 1284.3 protocol provides the means to address individual devices on the parallel port chain and to select a particular device for use by the host. When a DC device is selected the 'downstream' devices and the legacy device do not see any activity on the parallel port. To these devices the parallel port is idle.

One advantage of standardizing on the switching protocol is to enable the use of a common software driver on the host PC. The host driver, possibly provided by the Operating System, can then provide the switching services for the host application. The 1284.3 Service Provider Interface defines the types of services that are required by client applications. These services will be provided by the host.

This 1284.3 SPI is responsible for determining how many devices are attached to the parallel port and discovering what they are. This information is obtained by the 1284 device ID method for LPT plug and play. To the client application it appears that the DC device is the only device attached to the parallel port. The 1284.3 SPI driver manages the switching and selection of the peripheral.



Once you have a software layer to provide the basic port enumeration services it is a natural extension to include the data transfer functions. The 1284.3 Data Link interface defines these services. With a full software interface to the parallel port we are able to achieve independence from the actual hardware interface. This makes it easier to migrate to newer and faster hardware without effecting the client applications.

The 1284.3 Data Link defines a packet protocol that enables the delivery of packets of data to other side of the interface. This creates a connection-less, peer to peer relationship between the host and the peripheral. The header provides information on the data length and the transport protocol being used. This ensures that the data is only delivered to the appropriate client. The data length is used to enable DMA for the reception of data.

At this point the 1284/1284.3 protocol stack provides a software interface that is independent of the actual hardware implementation. With the appropriate implementation of the data link the parallel port could be used with many different transport protocols. Although this is true, there was no transport layer that could take advantage of the features of the 1284 interface. In particular the channel capabilities of the EPP and ECP modes.

The IEEE 1284.1 committee had defined a printer management standard that could operate over networks or direct attached interfaces. They requested that the IEEE develop a transport protocol that would take advantage of the 1284 standard parallel port capabilities but also be suitable for other physical interfaces. One of the main requirements was that the transport layer be able to maintain multiple conversations, or sessions, between the PC and the peripheral. In addition to this the protocol should ensure that one conversation cannot block another. An example of blocking is what you see in older Centronics interfaces. If the paper should run out of the printer or you open the tray to add more paper you can no longer communicate with the printer. With 1284.4 these functions should be independent and not effect one another.

The IEEE 1284.4 committee was established to develop and define this transport protocol. After analysis of various existing protocols, such as PPP and the MFP IS16550, it was determined that the Multiple Logical Channels (MLC) protocol developed by Hewlett-Packard and Genoa Technology met the general requirements for this standard. QualityLogic™ (formerly Genoa Technology) and HP offered this protocol as the basis for the 1284.4 standard. While MLC was the basis for 1284.4, the final standard is not backward compatible with MLC.

The 1284.4 protocol allows a device to carry on multiple, concurrent exchanges of data and/or control information with another device across a single point-to-point link. Blocking of one data exchange has no effect on the others. While the protocol was required to operate over the 1284 interface it may also operate over other interfaces such as USB and IEEE 1394.

1284.4 defines a packet protocol that enables this communication. Within the packet is the identity of the source and destination endpoints, client data, as well as control and credit information. Credit is the flow control method that is used to ensure that channel blocking does not occur.

With credit the target of a data transfer gives the source of the data an indication of how much data it is guaranteed to accept. In this way the source can never send more data than the target can accept. Upon initialization both sides negotiate for how much data needs to be transferred and in what packet sizes.



The source and target identities define a connection between the sides. A source may communicate with multiple target endpoints. In this way one client can communicate with multiple services on the target. A scanner is one example of how this may operate. The scanner application on the host may need to communicate with two functions on the scanner peripheral. The first function may be the scanner control and status monitor. The application will then establish a conversation between itself and this function on the scanner. At some point a page to be scanned may be inserted. Now the scanner can request the establishment of a connection between the scan engine and the host application. Now we have two independent conversations established between the PC and the peripheral.

This model can be extended to many different conversations. Conversations may occur between devices on different physical interfaces. The advantage of 1284.4 is that it can port to other PC related interfaces by the implementation of the appropriate software interface. This means that a client application on the PC can communicate with devices on multiple physical interfaces without any changes.

In conclusion, the parallel port is being enhanced by new hardware and software support. With backward compatibility and a huge installed base this interface will provide a convenient and powerful port for future peripherals. This presentation will provide an introduction to the protocol stack and issues involved in developing parallel port peripherals.

Glossary

- **bidirectional operation:** When the peripheral and host communicate using both forward and reverse data channels. As defined in this standard, Nibble and Byte Modes provide reverse channel communication and are used in conjunction with Compatibility Mode to provide bidirectional operation. Extended Capabilities Port (ECP) and Enhanced Parallel Port (EPP) Modes support bidirectional communication.
- **Centronics:** The popular name for the parallel printer port used as the parallel interface for most printers and supported by most "MS-DOS compatible" PCs. The name is derived from the printer manufacturer that introduced this interface, Centronics Data Computer Corporation. Despite a basic similarity, many variations of this interface have been implemented in different peripherals and hosts. This specification describes the more prevalent variations of the "Centronics" interface and defines a family of signaling methods that are backward compatible with the typical "Centronics" interface.
- **Centronics connector:** The popular name for the 36-pin ribbon contact type connector commonly used for the parallel port on printers. This connector is referred to as the "IEEE 1284-B" connector in this standard.
- **Command set:** A field in the Device ID message identifying the type of data expected by the peripheral. For example, a printer might use this field to report which page description language(s) it supports.
- **Compatible device:** A device that supports any of a specified range of popular variants of the "Centronics" interface. Compatible devices will interoperate with compliant devices in Compatibility Mode only.
- **Compliant device:** A device that supports either the Level 1 or Level 2 electrical interface, plus Compatible and Nibble Mode

operation, as well as the negotiation phases necessary to transition between these two modes.

- **Device driver:** A program that runs on the host and manages the sending and receiving of information from the peripheral. The driver utilizes the link level interface defined in this standard to communicate data between the application program and the peripheral personality.
- **Device ID:** A structured, variable length ASCII message identifying the manufacturer, command set, and model of the peripheral. Additional information may also be included. The Device ID is intended to assist the host in selecting the device and/or peripheral driver appropriate to the peripheral.
- **Forward channel:** Data path from the host to the peripheral.
- **Host:** A device, typically a personal computer, that will control the communications with attached peripherals.
- **IEEE 1284-A connector:** A plug or receptacle 25-pin subminiature D-shell connector, as specified in 8.2 and shown in Figure 24 of IEEE Std 1284-2000. This is the type of connector used on the MS-DOS compatible PC printer port adapter.
- **IEEE 1284-B connector:** A plug or receptacle 36-pin ribbon type connector, as specified in 8.2 and shown in Figure 25 of IEEE Std 1284-2000. This type of connector is also known as a "Centronics Connector."
- **IEEE 1284-C connector:** A miniature plug or receptacle 36-pin ribbon type connector, as specified in 8.2 and shown in Figure 26 of IEEE Std 1284-2000.
- **Level 1 device:** A device that supports the Level 1 electrical interface.
- **Level 2 device:** A device that supports the Level 2 electrical interface.
- **Link:** The physical connection and electronic hardware by which data is transferred between the host and the peripheral. This standard is concerned only with the link layer; the only information transfer defined or required as part of this standard is that necessary to control and synchronize the communication of peripheral data. The defined interface is transparent to the peripheral data communicated at data or higher layers.
- **n:** When preceding a capitalized signal name, denotes a signal having negative true logic (e.g., nAck).
- **PC parallel port:** The parallel printer port used as the parallel interface for most printers and supported by most (personal computers) PCs. This interface has been variously defined by different PC and peripheral manufacturers. This standard describes the more prevalent variations of the interface and defines a family of signaling methods that are backward compatible with the typical PC parallel port.
- **Peripheral:** A device, attached to a host via a communication link.
- **Peripheral personality:** The characteristics of a language processor or operating environment that a peripheral runs to interpret commands and data being sent.
- **Reverse channel:** Data path from the peripheral to the host.

- **Solicited status:** Information generated by the peripheral in response to a command from the host.
- **Status:** A term used generally to describe data generated by the peripheral that reflects the current operating state of the peripheral.
- **Status lines:** Unidirectional signals from the peripheral to the host, defined in Compatibility Mode to handshake data and to report error conditions. In other IEEE 1284 interface modes defined in this standard (IEEE Std 1284-2000), these lines are used for control, data, and/or status.
- **Unidirectional operation:** When the peripheral and host communicate data in one direction only. Compatibility Mode is unidirectional in the forward direction; Byte and Nibble Modes are unidirectional in the reverse direction.

For more information about Infogile products and services, contact the Infogile Sales Information Center at business@infogile.com. To access information on internet, go to www.Infogile.com.

© 2006 Infogile Corporation. All rights reserved.

This case study is for informational purposes only. INFOGILE MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY. Infogile and the Infogile logo are either registered trademarks or trademarks of Infogile Corporation in the United States and/or other countries. The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

