Interfacing the Extended Capabilities Port

Table of Contents

Introduction to the Extended Capabilities Port

The Extended Capabilities Mode was designed by Hewlett Packard and Microsoft to be implemented as the *Extended Capabilities Port Protocol and ISA Interface Standard*. This protocol uses additional hardware to generate hand shaking signals etc just like the EPP mode, thus runs at very much the same speed than the EPP mode. This mode, however may work better under Windows as it can use DMA channels to move it's data about. It also uses a FIFO buffer for the sending and/or receiving of data.

Another feature of ECP is a real time data compression. It uses Run Length Encoding (RLE) to achieve data compression ratio's up to 64:1. This comes is useful with devices such as Scanners and Printers where a good part of the data is long strings which are repetitive.

The Extended Capabilities Port supports a method of channel addressing. This is not intended to be used to daisy chain devices up but rather to address multiple devices within one device. Such an example is many fax machines on the market today which may contain a Parallel Port to interface it to your computer. The fax machine can be split up into separate devices such as the scanner, modem/Fax and printer, where each part can be addresses separately, even if the other devices cannot accept data due to full buffers.

ECP Hardware Properties

While Extended Capabilities Printer Ports use exactly the same D25 connector as your SPP, ECP assigns different tasks to each of the pins, just like EPP. This means that there is also a different handshake method when using a ECP interface.

The ECP is backwards compatible to the SPP and EPP. When operating in SPP mode, the individual lines operate in exactly the same fashion than the SPP and thus are labelled Strobe, Auto Linefeed, Init, Busy etc. When operating in EPP mode, the pins function according to the method described in the EPP protocol and have a different method of hand shaking. When the port is operating in ECP mode, then the following labels are assigned to each pin.

Pin	SPP Signal	ECP Signal	Direct'n	Function
1	Strobe	HostCLK	Out	A low on this line indicates, that there is valid data at the host. When this pin is de-asserted, the +ve clock edge should be used to shift the data into the device.
2-9	Data 0-7	Data 0-7	In/Out	Data Bus. Bi-directional
10	Ack	PeriphCLK	In	A low on this line indicates, that there is valid data at the Device. When this pin is de-asserted, the +ve clock edge should be used to shift the data into the Host.
11	Busy	PeriphAck	In	When in reverse direction a HIGH indicates Data, while a LOW indicates a Command Cycle.\nIn forward direction, functions as PeriphAck.
12	Paper Out End	nAckReverse	In	When Low, Device acknowledges Reverse Request.
13	Select	X-Flag	In	Extensibility Flag
14	Auto Linefeed	Host Ack	Out	When in forward direction a HIGH indicates Data, while a LOW indicates a Command Cycle.\nIn reverse direction, functions as HostAck.
15	Error Fault	PeriphRequest	In	A LOW set by the device indicates reverse data is available.
16	Initialise	NReverse Request	Out	A LOW indicates data is in reverse direction.
17	Select Printer	1284 Active	Out	A HIGH indicates Host is in 1284 Transfer Mode. Taken low to terminate.
18-25	Ground	Ground	GND	Ground.

Table 1. Pin Assignments For Extended Capabilities Parallel Port Connector.

The HostAck and PeriphAck lines indicate whether the signals on the data line are data or a command. If these lines are high then data is placed on the data lines (Pins 2-7). If a command cycle is taking place then the appropriate line will be low, i.e. if the host is sending a command, then HostAck will be low or if the device/peripheral is sending a command the PeriphAck line will be low.

A command cycle can be one of two things, either a RLE count or an address. This is determined by the bit 7 (MSB) of the data lines, i.e. pin 9. If bit 7 is a 0, then the rest of the data (bits 0-6) is a run length count which is used with the data compression scheme. However if bit 7 is a 1, then the data present on bits 0 to 6 is a channel address. With one bit missing this can only be a value from 0 to 127(DEC).

The ECP Handshake

The ECP handshake is different to the SPP handshake. The most obvious difference is that ECP has the ability at anytime to transmit data in any direction, thus additional signalling is required. Below is the ECP handshake for both the Forward and Reverse Directions.

ECP Forward Data Cycle

ECP Forward Data Cycle
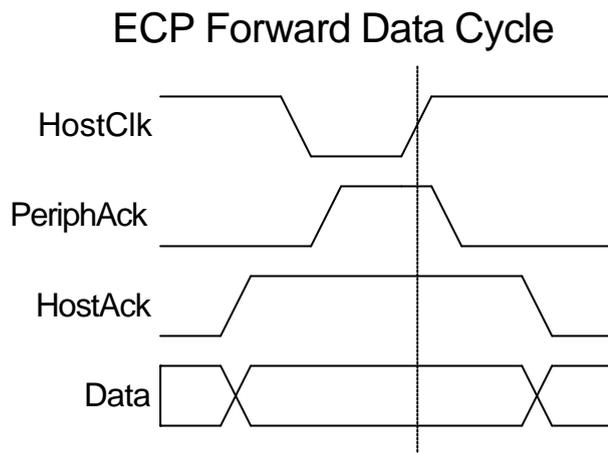
HostClk

PeriphAck

HostAck

Data

Figure 1. Enhanced Capabilities Port Forward Data Cycle.

1. **Data is placed on Data lines by Host.**

2. Host then indicates a Data Cycle will proceed by asserting HostAck.

3. **Host indicates valid data by asserting HostClk low.**

4. Peripheral sends its acknowledgment of valid data by asserting PeriphAck.

5. **Host de-asserts HostClk high. +ve edge used to shift data into the Peripheral.**

6. Peripheral sends it's acknowledgment of the byte via de-asserting PeriphAck.

ECP Forward Command Cycle

ECP Forward Command Cycle
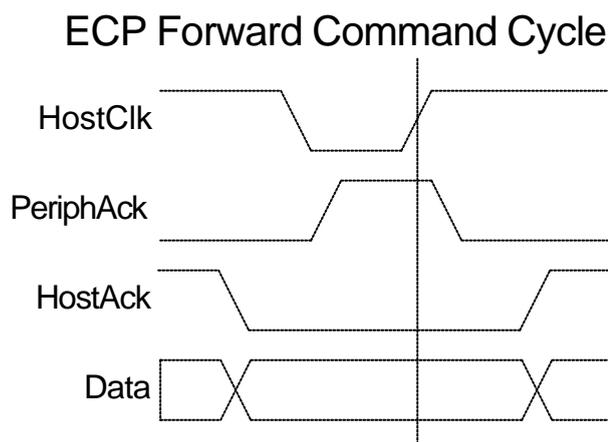
HostClk

PeriphAck

HostAck

Data

Figure 2. Enhanced Capabilities Port Forward Command Cycle.

1. **Data is placed on Data lines by Host.**

2. Host then indicates a Command cycle will proceed by de-asserting HostAck.

3. **Host indicates valid data by asserting HostClk low.**

4. Peripheral sends its acknowledgment of valid data by asserting PeriphAck.

5. **Host de-asserts HostClk high. +ve edge used to shift data into the Peripheral.**

6. Peripheral sends it's acknowledgment of the byte via de-asserting PeriphAck.

ECP Reverse Data Cycle
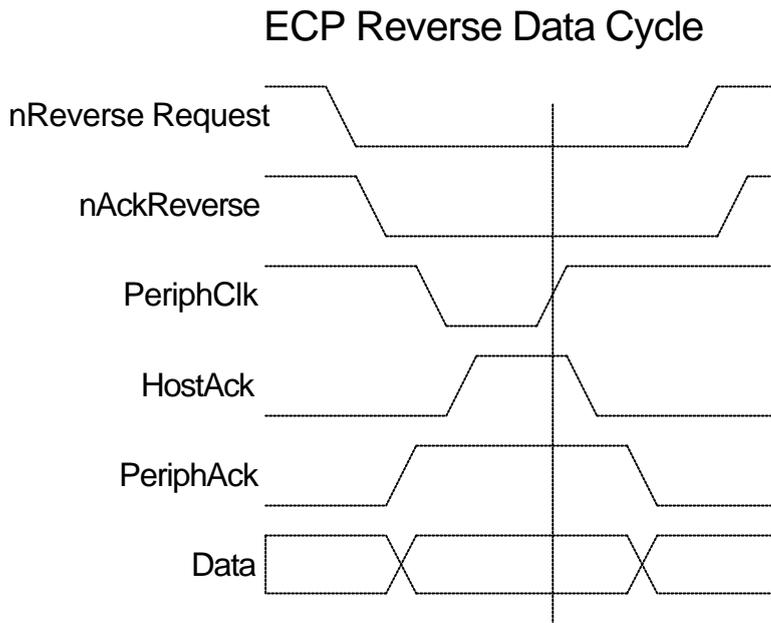
ECP Reverse Data Cycle

Figure 3. Enhanced Capabilities Port Reverse Data Cycle.

1. **Host sets nReverseRequest Low to request a reverse channel.**

2. Peripheral acknowledges reverse channel request via asserting nAckReverse low.

3. **Data is placed on data lines by Peripheral.**

4. Data cycle is then selected by Peripheral via PeriphAck going high.

5. **Valid data is indicated by the Peripheral setting PeriphClk low.**

6. Host sends its acknowledgment of valid data via HostAck going high.

7. **Device/Peripheral sets PeriphClk high. +ve edge used to shift data into the Host.**

8. Host sends it's acknowledgment of the byte by de-asserting HostAck low.

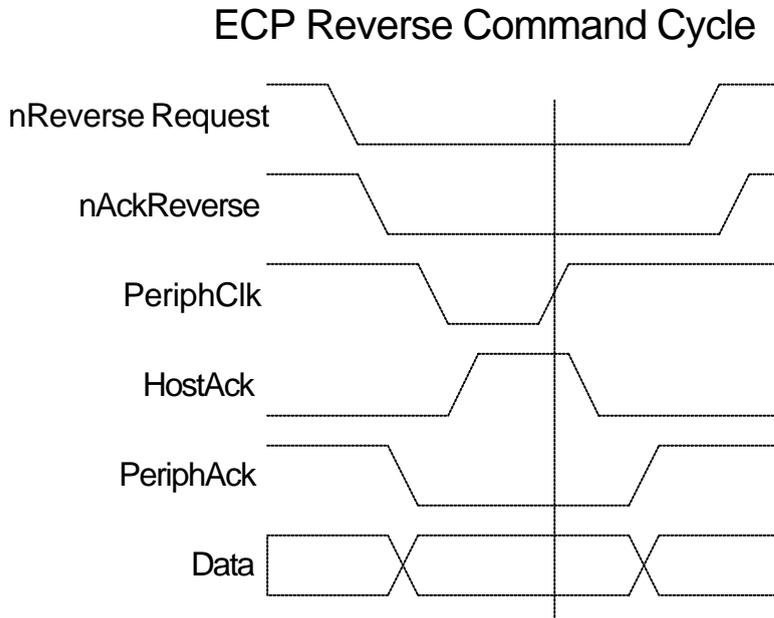## ECP Reverse Command Cycle

ECP Reverse Command Cycle

Figure 4. Enhanced Capabilities Port Reverse Command Cycle.

1. **Host sets nReverseRequest Low to request a reverse channel.**

2. Peripheral acknowledges reverse channel request via asserting nAckReverse low.

3. **Data is placed on data lines by Peripheral.**

4. Command cycle is then selected by Peripheral via PeriphAck going low.

5. **Valid data is indicated by the Peripheral setting PeriphClk low.**

6. Host sends its acknowledgment of valid data via HostAck going high.

7. **Device/Peripheral sets PeriphClk high. +ve edge used to shift data into the Host.**

8. Host sends it's acknowledgment of the byte by de-asserting HostAck low.

EPP Handshake vs SPP Handshake

If we look back at the SPP handshake you will realise it only has 5 steps,

1. *Write the byte to the Data Port*

2. *Check to see is the printer is busy. If the printer is busy, it will not accept any data, thus any data which is written will be lost.*

3. *Take the Strobe (Pin 1) low. This tells the printer that there is the correct data on the data lines. (Pins 2-9)*

4. *Put the strobe high again after waiting approximately 5 microseconds after putting the strobe low. (Step 3)*

5. *Check for Ack from Peripheral.*

and that the ECP handshake has many more steps. This would suggest that ECP would be slower that SPP. However this is not the case as all of these steps above are controlled by the hardware on your I/O control. If this handshake was to be implemented via software control then it would be a lot slower that it's SPP counterpart.

RLE - Run Length Encoding

As briefly discussed earlier, the ECP Protocol includes a Simple Compression Scheme called Run Length Encoding. It can support a maximum compression ratio of 64:1 and works by sending repetitive single bytes as a run count and one copy of the byte. The run count determines how many times the following byte is to be repeated.

For example, if a string of 25 'A's were to be sent, then a run count byte equal to 24 would be sent first, followed by the byte 'A'. The receiving peripheral on receipt of the Run Length Count, would expand (Repeat) the next byte a number of times determined via the run count.

The Run Length Byte has to be distinguished from other bytes in the Data Path. It is sent as a Command to the ECP's Address FIFO Port. Bytes sent to this register can be of two things, a Run Length Count or an Address. These are distinguished by the MSB, Bit 7. If Bit 7 is Set (1), then the other 7 bits, bits 0 to 6 is a channel address. If Bit 7 is Reset (0), then the lower 7 bits is a run length count. By using the MSB, this limits channel Addresses and Run Length Counts to 7 Bits (0 - 127).

ECP Software Registers

The table below shows the registers of the Extended Capabilities Port. The first 3 registers are exactly the same than with the Standard Parallel Port registers. Note should be taken, however, of the Enable bi-directional Port bit (bit 5 of the Control Port.) This bit reflects the direction that the ECP port is currently in, and will effect the FIFO Full and FIFO Empty bits of the ECR Register, which will be explained later.

Address	Port Name	Read/Write
Base + 0	Data Port (SPP)	Write
	ECP Address FIFO (ECP MODE)	Read/Write
Base + 1	Status Port (All Modes)	Read/Write
Base + 2	Control Port (All Modes)	Read/Write
Base + 400h	Data FIFO (Parallel Port FIFO Mode)	Read/Write
	Data FIFO (ECP Mode)	Read/Write
	Test FIFO (Test Mode)	Read/Write
	Configuration Register A (Configuration Mode)	Read/Write
Base + 401h	Configuration Register B (Configuration Mode)	Read/Write
Base + 402h	Extended Control Register (Used by all modes)	Read/Write

Table 2 : ECP Registers

ECP's Extended Control Register (ECR)

The most important register with a Extended Capabilities Parallel Port is the Extended Control Register (ECR) thus we will target it's operation first. This register sets up the mode in which the ECP will run, plus gives status of the ECP's FIFO among other things. You will find the contents of this register below, in more detail.

Bit		Function
7:5	*Selects Current Mode of Operation*	
	000	Standard Mode
	001	Byte Mode
	010	Parallel Port FIFO Mode
	011	ECP FIFO Mode
	100	EPP Mode
	101	Reserved
	110	FIFO Test Mode
	111	Configuration Mode
4	ECP Interrupt Bit	
3	DMA Enable Bit	
2	ECP Service Bit	
1	FIFO Full	
0	FIFO Empty	

Table 3 ECR - Extended Control Register

The three MSB of the Extended Control Register selects the mode of operation. There are 7 possible modes of operation, but not all ports will support all modes. The EPP mode is one such example, not being available on some ports. On the next page is a table of the Modes of Operation.

Modes of Operation

Standard Mode	Selecting this mode will cause the ECP port to behave as a Standard Parallel Port, without Bi-directional functionality.
Byte Mode / PS/2 Mode	Behaves as a SPP in Bi-directional (Reverse) mode.
Parallel Port FIFO Mode	In this mode, any data written to the Data FIFO will be sent to the peripheral using the SPP handshake. The hardware will generate the handshaking required. Useful with non-ECP devices such as Printers. You can have some of the features of ECP like FIFO buffers and hardware generation of handshaking but with the existing SPP handshake instead of the ECP handshake.
ECP FIFO Mode	Standard Mode for ECP use. This mode uses the ECP Handshake, already described.
EPP Mode / *Reserved*	On some chipsets, this mode will enable EPP to be used. While on others, this mode is still reserved.
Reserved	Currently Reserved.
FIFO Test Mode	While in this mode, any data written to the Test FIFO Register will be placed into the FIFO and any data read from the Test FIFO register will be read from the FIFO buffer. The FIFO Full/Empty Status Bits will reflect their true value, thus FIFO depth, among other things can be determined in this mode.
Configuration Mode	In this mode, the two configuration registers, cnfgA & cnfgB become available at their designated Register Addresses.

As outlined above, when the port is set to operate in Standard Mode, it will behave just like a Standard Parallel Port (SPP) with no bi-directional data transfer. If you require bi-directional transfer, then set the mode to Byte Mode. The Parallel Port FIFO mode and ECP FIFO mode both use hardware to generate the necessary handshaking signals. The only difference between each mode is that The Parallel Port FIFO Mode uses SPP handshaking, thus can be used with your SPP printer. ECP FIFO mode uses ECP handshaking.

The FIFO test mode can be used to test the capacity of the FIFO Buffers as well as to make sure they function correctly. When in FIFO test mode, any byte which is written to the TEST FIFO (Base + 400h) is placed into the FIFO buffer and any byte which is read from this register is taken from the FIFO Buffer. You can use this along with the FIFO Full and FIFO Empty bits of the Extended Control Register to determine the capacity of the FIFO Buffer. This should normally be about 16 Bytes deep.

The other Bits of the ECR also play an important role in the operation of the ECP Port. The ECP Interrupt Bit, (Bit 4) enables the use of Interrupts, while the DMA Enable Bit (Bit 3) enables the use of Direct Memory Access. The ECP Service Bit (Bit 2) shows if an interrupt request has been initiated. If so, this bit will be set. Resetting this bit is different with different chips. Some require you to Reset the Bit, e.g. Write a Zero to it. Others will reset once the Register has been read.

The FIFO Full (Bit 1) and FIFO Empty (Bit 0) show the status of the FIFO Buffer. These bits are direction dependent, thus note should be taken of the Control Register's Bit 5. If bit 0 (FIFO Empty) is set, then the FIFO buffer is completely empty. If Bit 1 is set then the FIFO buffer is Full. Thus, if neither bit 0 or 1 is set, then there is data in FIFO, but is not yet full. These bits can be used in FIFO Test Mode, to determine the capacity of the FIFO Buffer.

ECP's Configuration Register A (cnfgA)

Configuration Register A is one of two configuration registers which the ECP Port has. These Configuration Registers are only accessible when the ECP Port is in Configuration Mode. (See Extended Control Register) CnfgA can be accessed at Base + 400h.

Bit		Function
7	1	Interrupts are level triggered
	0	Interrupts are edge triggered (Pulses)
6:4	00h	Accepts Max. 16 Bit wide words
	01h	Accepts Max. 8 Bit wide words
	02h	Accepts Max. 32 Bit wide words
	03h:07h	Reserved for future expansion
3	Reserved	
2	*Host Recovery : Pipeline/Transmitter Byte included in FIFO?*	
	0	In forward direction, the 1 byte in the transmitter pipeline doesn't affect FIFO Full.
	1	In forward direction, the 1 byte in the transmitter pipeline is include as part of FIFO Full.
1:0	Host Recovery : Unsent byte(s) left in FIFO	
	00	Complete Pword
	01	1 Valid Byte
	10	2 Valid Bytes
	11	3 Valid Bytes

Table 4 ECR - Configuration Register A

Configuration Register A can be read to find out a little more about the ECP Port. The MSB, shows if the card generates level interrupts or edge triggered interrupts. This will depend upon the type of bus your card is using. Bits 4 to 6, show the buses width within the card. Some cards only have a 8 bit data path, while others may have a 32 or 16 bit width. To get maximum efficiency from your card, the software can read the status of these bits to determine the Maximum Word Size to output to the port.

The 3 LSB's are used for Host Recovery. In order to recover from an error, the software must know how many bytes were sent, by determining if there are any bytes left in the FIFO. Some implementations may include the byte sitting in the transmitter register, waiting to be sent as part of the FIFO's Full Status, while others may not. Bit 2 determines weather or not this is the case.

The other problem is that the Parallel Ports output is only 8 bits wide, and that you many be using 16 bit or 32 bit I/O Instructions. If this is the case, then part of your Port Word (Word you sent to port) may be sent. Therefore Bits 0 and 1 give an indication of the number of valid bytes still left in the FIFO, so that you can retransmit these.

ECP's Configuration Register B (cnfgB)

Configuration Register B, like Configuration Register A is only available when the ECP Port is in Configuration Mode. When in this mode, cnfgB resides at Base + 401h. On the next page you will find the make-up of the cnfgB Register.

Bit(s)		Function
7	1	Compress outgoing Data Using RLE
	0	Do Not compress Data
6	Interrupt Status - Shows the Current Status of the IRQ Pin	
5:3	*Selects or Displays Status of Interrupt Request Line.*	
	000	Interrupt Selected Via Jumper
	001	IRQ 7
	010	IRQ 9
	011	IRQ 10
	100	IRQ 11
	101	IRQ 14
	110	IRQ 15
	111	IRQ 5
2:0	*Selects or Displays Status of the DMA Channel the Printer Card Uses*	
	000	Uses a Jumpered 8 Bit DMA Channel
	001	DMA Channel 1
	010	DMA Channel 2
	011	DMA Channel 3
	100	Uses a Jumpered 16 Bit DMA Channel
	101	DMA Channel 5
	110	DMA Channel 6
	111	DMA Channel 7

Table 5 - Configuration B Register

The Configuration Register B (cnfgB) can be a combination of read/write access. Some ports may be software configurable, where you can set the IRQ and DMA resources from the register. Others may be set via BIOS or by using jumpers on the Card, thus are read only.

Bit 7 of the cnfgB Register selects whether to compress outgoing data using RLE (Run Length Encoding.) When Set, the host will compress the data before sending. When reset, data will be sent to the peripheral raw (Uncompressed). Bit 6 returns the status of the IRQ pin. This can be used to diagnose conflicts as it will not only reflect the status of the Parallel Ports IRQ, but and other device using this IRQ.

Bits 5 to 3 give status of about the Port's IRQ assignment. Likewise for bits 2 to 0 which give status of DMA Channel assignment. As mentioned above these fields may be read/write. The disappearing species of Parallel Cards which have Jumpers may simply show it's resources as "Jumpered" or it may show the correct Line Numbers. However these of course will be read only.