Parallel port

From OSDev Wiki

The parallel port uses a sub-d 25 connector to provide a 8-bit data bus. It is commonly used by printers. There are 3 kinds of parallel ports: Standard Parallel Port (SPP), Enhanced Parallel Port (EPP) and Extended Capabilities Parallel Port (ECP). iirc they are all part of IEEE Standard 1284, or is it just the second two?

Contents

- 1 Pin types
- 2 Registers & Common Address
- 3 Parallel Port Software Interface
  - 3.1 Data Register
  - 3.2 Status Register
  - 3.3 Control Register
- 4 Standard Parallel Port Mode
  - 4.1 Centronics Handshaking

Pin types

Most parallel ports come in either 36 or 25 pin varieties. 25 being the most common.

Registers & Common Address

Common base addresses are:

- LPT1: 0x378 (or occasionally 0x3BC) (IRQ 7)
- LPT2: 0x278 (IRQ 6)
- LPT3: 0x3BC (IRQ 5)

The BIOS Data Area should contain the IO base addresses of any existing parallel ports

Parallel Port Software Interface

This page or section is a stub. You can help the wiki by *accurately* contributing (http://wiki.osdev.org/index.php?title=Parallel_port&action=edit) to it.

Each parallel port has three IO port registers, Data, Status and Control. Their addresses are relative to the base address of the parallel port in question.

Data Register

Address = Base Address + 0

Any byte writen to this register is put on pins 2 through 9 of the port. Any read from this register reflects the state of the port.

Status Register

Address = Base Address + 1

Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
Reserved	Reserved	IRQ	ERROR	SELECT_IN	PAPER_OUT	ACK	BUSY

The ERROR, ACK and BUSY signals are active low when reading from the IO port.

Control Register

Address = Base Address + 2

Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
STROBE	AUTO_LF	INITIALISE	SELECT	IRQACK	BIDI	Unused	Unused

The INITIALISE signal is active low when writing to the IO port.

The STROBE signal is for handshaking and alerts the printer to data being ready at the data port.

AUTO_LF is the Automatic Line-Feed signal. If this is set and the printer receives a Carriage-Return character (0x0D), the printer will automatically perform a Line-Feed (character 0x0A) as well.

INITIALISE, sometimes called PRIME, alerts the device that data that a data conversation is about to start. This signal may result in a printer performing a reset and any buffers being flushed.

Standard Parallel Port Mode

This is the most basic of the parallel modes. The other modes are EPP and ECP. All systems should support this mode and it may well be the default at boot time. Some BIOSes also support setting the default mode of the parallel port. In this mode, the Data register and the Control register are Write-Only and the Status register is Read-Only.

In Standard (or Compatibility) mode, data is sent using a mechanism called Centronics Handshaking, described below.

This mode requires communuication handshaking to be performed by software which limits the maximum data throughput of the port. This means that a standard mode has a maximum transfer rate of around 1000 bytes per second, depending on the timings of the computer and receiving device. The

more advanced types (or modes) of parallel ports, EPP and ECP, reduce this by providing hardware-based handshaking. Relieving software of this requirement reduces CPU load and increases the port's maximum potential speed.

For a line printer, this method should be enough to get things going, simply sending characters using this method to the parallel port while the printer is online should get the print head moving, assuming no buffers are in the way to store the values.

Depending on the connected device, you may have to raise the INITIALISE signal before data transmission to ready the device, and possibly again after in order to flush any buffers.

Centronics Handshaking

In Standard (or Compatibility) mode, data is sent to the connected device by writing the byte to the data port, then pulsing the STROBE signal. This pulse informs the device that data is ready to be read. The device will respond by raising its BUSY signal and then reading the data and performing some processing on it. Once this processing is complete, the device will lower the Busy signal and may raise a brief ACK signal to indicate that it has finished.

```
// Sends a byte to the printer
void Parallel_SendByte( unsigned char pData )
{
	unsigned char lControl;

	// Wait for the printer to be receptive
	while ( ! Ports_In8( 0x379 ) & 0x80 )
	{
		Timer_Delay( 10 );
	}

	// Now put the data onto the data lines
	Ports_Out8( 0x378, pData );

	// Now pulse the strobe line to tell the printer to read the datc
	lControl = Ports_In8( 0x37A);
	Ports_Out8( 0x37A, lControl | 1 );
	Timer_Delay( 10 );
	Ports_Out8( 0x37A, lControl );


	// Now wait for the printer to finish processing
	while ( ! Ports_In8( 0x379 ) & 0x80 )
	{
		Timer_Delay( 10 );
	}
}
```

`Timer_Delay()` pauses processing for the specified number of milliseconds. `Ports_In8()` and `Ports_Out8()` read and write a byte of data to/from the IO port.

Categories: Stubs | Common Devices