Parallel I/O Ports

This lecture covers the design of parallel I/O ports. Parallel ports are used to interface the CPU to I/O devices. Two common parallel port standards, the "Centronics" parallel port and the SCSI interface, are also described.
After this lecture you should be able to design simple input and output I/O ports using address decoders, registers and tri-state buffers.

I/O Ports

All useful microcomputer systems have input/output (I/O) devices. These I/O devices move data between the outside world and the computer. The interface between the CPU and these I/O devices is through registers that appear as memory locations in the address or I/O address spaces of the processor. Through these registers the CPU can input (read) or output (write) a number of bits (typically one byte) at a time.

Typical examples of I/O port include output ports that drive LEDs, ports to scan a keypad, ports to control machinery, etc. More complex I/O interfaces such as floppy disk controllers or serial interface chips usually contain several I/O ports. Some ports are used to obtain status information about the interface through "status registers" and other ports can control the interface's operation through "control registers."

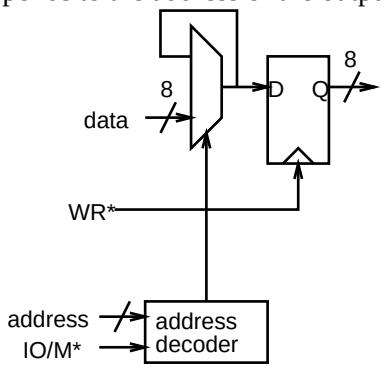
For example, each printer interface on the IBM PC has associated with it a status port that can be used to obtain certain status information (busy, on-line, out of paper, etc). The printer interface also has a control port that can be used to reset the printer and set the automatic line feed mode. In addition, there is an output port that is used to output the character to be printed.

Implementation of I/O Ports

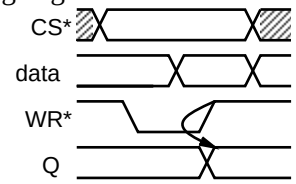
Output

Output ports are implemented using registers. The register's data inputs (D) are connected to the CPU data bus and the register's clock input is driven by a write strobe (e.g. MEMW*). In addition, an address decoder is used to make sure the register is only reloaded when the CPU is addressing the desired IO or memory address. The rising edge of the write strobe loads the data into the register output (Q) and this output stays fixed until the register is written again.

The following schematic shows how a register could be connected to operate as an output port. The CPU's write strobe (WR*) is used to clock the data into the register, but only if the address on the CPU bus corresponds to the address of the output port:



The following timing diagram shows the relationship between the signals. Note that the output is held after the rising edge of the write strobe (WR*):

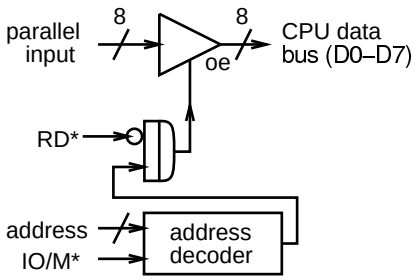


Input

Input ports can also be implemented with a minimum of hardware. A tri-state buffer is used to connect the external digital input to the CPU's data bus during a read cycle if the CPU is addressing the memory or IO address assigned to that input port. The read strobe (RD*) is used to enable the buffer so that it connects the external input to the CPU data bus.

The following schematic shows how a register could be connected to operate as a parallel input port. The CPU's read strobe (RD*) is used to enable the output of a tri-state buffer when RD* is active and

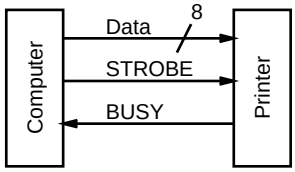
the address corresponds to the address of the input port:



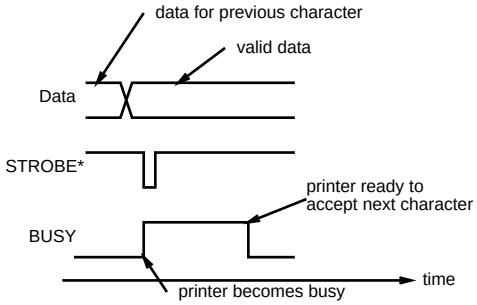
The value read by the CPU will be the value on the input at the time that the port is read.

"Centronics" Parallel Printer Port

This simple unidirectional (output only) interface is used to drive printers. There are 8 data lines and two data transfer control lines, STROBE* and BUSY. BUSY is an output from the printer that is high when the printer cannot accept data. STROBE* is a an output from the PC which is strobed (brought low and then high again) to transfer the data on the data lines to the printer. This interface uses TTL (L$=$ 0V H$\approx +5$V) signal levels.



To write a character to the printer the computer waits until busy is low, puts the character on data lines and brings STROBE* low and then high again.



There are additional handshaking lines to control various printer features (e.g. auto line feed) and to indicate various printer status conditions (e.g. out of paper).

There original IBM PC's parallel port was an output-only Centronics-compatible interface but in recent designs the port can also be configured as an input. The maximum speed depends on the CPU speed and software used but is typically 50 to 100 kbytes/s.

IEEE standard 1284 specifies a parallel port that is bidirectional and allows for higher-speed transfers by using hardware to take care of the handshaking operations.

Small Computer System Interface (SCSI)

This parallel interface allows for bidirectional data transfer and for up to 8 hosts ("initiators") and peripherals ("targets") to be connected together in a bus (parallel) topology. The SCSI interface is well defined and is available on many different computers. It is widely used to connect computers to disk and tape drives, CD-ROMs, scanners, high-speed printers, etc.

The SCSI interface includes a protocol for arbitrating access to the bus by initiators and for selecting a specific target. The actual data transfers over the SCSI bus use a similar request/acknowledge protocol with each byte transfer being acknowledged by the target before another byte is transferred.

Depending on the speed of the peripheral and the host interface the bus can transfer data at up to several megabytes per second. The SCSI devices attached to the bus are electrically connected in parallel with each device configured to respond to a particular bus ID number (ID).

The physical interface uses a 50-pin connectors with two connectors on each device so that they can be daisy-chained. Because of the high bus speeds, care has to be taken to properly terminate the bus in it's characteristic impedance to minimize reflections. Like the Centronics interface the SCSI bus also uses TTL-level signals but it needs open-collector or tri-state drivers.

Another advantage of the SCSI interface is that it defines a set of common commands for devices with similar characteristics. This allows the same software to drive different devices. For example, the same generic commands (rewind, skip forward, etc) can be used to control tape drives from different manufacturers.

Although a SCSI interface can be built using a

simple parallel interface and programmed i/o, this type of interface is too slow for most applications. SCSI interface chips are available that implement the interface between the CPU and the SCSI bus and transfer data using either DMA or on-board FIFOs.

Software Aspects

The value on the output port is set with MOV (if the port is memory-mapped) or OUT (if the port is mapped into the I/O space) instructions. Similarly, the value on an input port is read with a MOV or IN instruction.

It's often necessary to set or clear a particular bit on an output port or to test the value of a particular bit on an input port. This can be done with bit masks and the bit-wise logical operations AND and OR.

Other Parallel Buses

The IDE/ATAPI parallel bus is used mainly to interface a CPU to disk drivers. It is commonly seen on IBM-PC compatible computers.

The IEEE-488 standard (also known as the General Purpose Interface Bus (GPIB) and HPIB) is another bidirectional interface. Like the SCSI bus it allows multiple bus masters ("talkers") and slaves ("listeners"). It was developed by HP who named it HPIB (HP Interface Bus). The standard is called IEEE-488. This bus is used mostly for control of laboratory instruments.