The SCSI Interface

This lecture describes the behaviour of a basic SCSI interface.
After this lecture you should be able to list some advantages and disadvantages of the SCSI peripheral interface and give the values of the signals on the SCSI bus during the various bus phases.

Introduction

In a previous lecture we studied the "Centronics" parallel printer interface. This interface is limited to driving one device per interface, it is unidirectional and it is limited to one type of peripheral – a printer.

The Small Computer Systems Interface (SCSI) is a more flexible parallel interface. This interface can control up to 8 devices from the same interface, can transfer data both to- and from- the peripheral and can control many different types of devices. Both the physical and logical aspects of the SCSI interface have been standardized in enough detail that similar types of devices from different manufacturers will work with the same device driver software. For example, different manufacturers' SCSI disk drives all have the same connectors, accept the same commands, issue the same status messages, etc.

The SCSI standard defines conceptual models and commands for ten classes of devices including disk drives, tape drives, printers, scanners, and others.

The SCSI interface makes life easier for the consumer and system integrator by making it possible to use peripherals from many different manufacturers and by eliminating the need to re-write device driver software for each new model. However, the additional parts and NRE costs of the SCSI controller (and possibly the need to add a host interface to the computer) can add significantly to the cost.

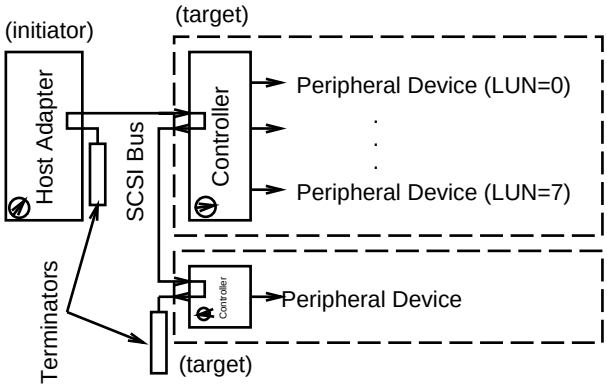
Enhancements to the original SCSI standard have increased the bus width to 32 bits and added additional features to increase the transfer rate. All of these additions are backwards compatible. In this lecture we will study a basic SCSI interface and briefly mention some of the extensions at the end.

SCSI Bus Operation

Initiators, Targets, LUNs and SCSI IDs

The SCSI bus connects a *host adapter* in a computer with one or more *peripheral controllers* attached to peripherals such as disk drives. The host adapter is also called an *initiator* because it initiates an I/O procedure and the controller is called a *target* because it is the target of the host's commands.

Each device on the SCSI bus is assigned an address (or "SCSI ID") between 0 and 7 by means of a switch. In addition, each controller can support up to eight logical units (LUN - Logical Unit Number) as shown in the diagram below:



Physical Characteristics

Each device on the SCSI bus has *two* 50-pin connectors which allow the devices to be daisy-chained into a parallel bus.

Since the signals on the SCSI bus switch at several MHz and the total cable length can be several metres, the bus must be treated as a transmission line and terminated at both ends to minimize reflections and the resulting signal distortion. An unterminated SCSI bus will usually not work reliably.

The signals on the SCSI bus are at TTL levels (0 to 5 volts). Two of the signals (BSY* and SEL*) must be open-collector outputs because they can be driven

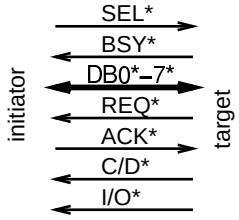
simultaneously by multiple devices. The other signals can be either tri-state or open-collector outputs.

The terminators consist of a 330 ohm resistor to +5V and a 220 ohm resistor to ground.

Exercise: What is the impedance of these terminators? What would be a good choice for the impedance of the cables?

Control Signals

The basic pins on the SCSI bus and their functions are described below. Note that all the signals are active-low.



- DB0* to DB7* - these 8 parallel data bits are used to transfer commands, data and status during various phases of a SCSI operation. During the SELECTION bus phase the data bits are also used by the initiator to select a particular target.

- SEL[ect]* - a signal driven by the initiator during the SELECTION bus phase to select a particular target. For example, if DB7 is asserted when SEL* is asserted then the target with SCSI ID 7 becomes the active target.

- B[u]SY* - a signal asserted by the selected target to indicate that it is using the bus.

- CONTROL*/DATA (C*/D) - a signal driven by the target to indicate whether data or control (command or status) information is to be transferred between the initiator and target.

- INPUT*/OUTPUT (I*/O) - also driven by target, and defines whether the *initiator* is to do an input or output operation to the data bus. The table below summarizes the meanings of the four combinations of C*/D and I*/O.

C*/D	I*/O	Meaning
CONTROL	OUTPUT	command from initiator
DATA	INPUT	data from target
DATA	OUTPUT	data to target
CONTROL	INPUT	status byte from the target

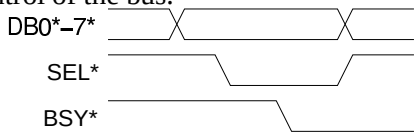
- REQ* and ACK* - two signals used for handshaking for the four types of information transfer over the data bus. REQ* is driven by the target and ACK* is driven by the initiator.

Bus Phases

The basic operation of the SCSI bus proceeds through four bus states:

SELECTION PHASE In this phase the initiator selects a particular target. The initiator waits until the bus is free (as indicated by BSY* not being asserted), asserts one of the bits on the data bus and SEL*. The addressed target device then recognizes it is being selected and takes control of the bus by asserting BSY*.

The timing diagram below shows how an initiator selects a target and how the target takes control of the bus.



Exercise: A tape drive set to SCSI ID 1 and a disk drive set to SCSI ID 5 are both connected to a SCSI bus. If the host controller puts the (logical) value 020H on the data bus and asserts SEL* which device will take control of the bus? What do you think would happen if this device was turned off?

COMMAND PHASE This target then reads a 6-byte "command descriptor block" from the initiator. This command specifies the type of operation (e.g. read, write, seek, return status, etc.), the LUN, information about the data to be transferred (e.g. sector number) and the amount of data to be transferred in the subsequent data phase. To obtain the command the target asserts OUT and CONTROL and uses REQ*/ACK* handshaking to transfer 6 bytes. (The handshake details are given in the next section).

DATA PHASE Based on the command received from the initiator the target then performs the requested operation (e.g. reads a sector) and/or transfers the desired data to/from the initiator. The target transfers the data by asserting DATA and either IN or OUT depending on the transfer direction.

STATUS PHASE When the data transfer is complete the target transfers a status byte to the initiator. It does this by transferring one byte while asserting CONTROL and IN. If this byte is zero then the command completed successfully, otherwise the initiator may issue a "request sense" command to find out what went wrong.

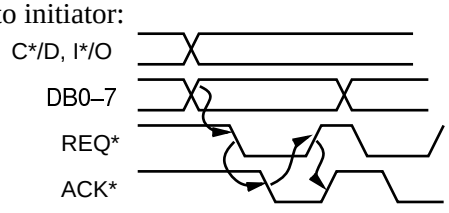
BUS FREE PHASE When the status byte has been transferred the target releases BSY* and this makes the bus available for the next select/command/data/status transfer sequence.

Information Transfer Handshaking

All information transfer (command, data or status) over the data bus between the initiator and target is controlled by the target. The target transfers one byte at a time using handshaking with the REQ* and ACK* lines.

The target asserts REQ* to start a transfer and the initiator responds with ACK* to acknowledge the transfer. On transfers from the target to the initiator the target asserts the data before asserting REQ* and waits until ACK* is asserted before sending the next byte. On transfers from the initiator to the target the target asserts REQ* and waits until ACK* is asserted before reading the data from the bus.

The following timing diagram shows an example of the handshaking during information transfer from target to initiator:



Exercise: Which device (initiator or target) determines the data transfer rate during the information transfer?

Command Parameter Blocks

The contents of SCSI command parameter blocks are usually described in the form of a table showing the meanings of the different bits. Table 2 shows an example of the command block for disk read operation.

The first byte is the operation code (08H). The MS 3 bits of the second byte contain the LUN (typically 0 if only one disk is attached to the controller). The next 20 bits give the starting logical block number to be transferred and the fifth 8 bits are the number of blocks to transfer. The bits in the last (control) byte are used for optional control information and if not used all of the bits may be set to zero.

Exercise: If each logical block holds 512 bytes, how large a disk drive could this command support?

Enhancements to SCSI Protocol

Commands longer than 6 bytes. The SCSI standard defines command parameter blocks of 6, 10, and 12 bytes. The larger commands allow longer block addresses and transfer counts to be specified in the commands.

DB-25 connectors. The original SCSI interface used a large 50-pin connector. A smaller and less expensive DB-25 connector is often used on PCs and low-cost SCSI peripherals.

Messages. The SCSI-II standard extends the SCSI handshaking by allowing variable-length "messages" to be exchanged between target and initiator. The messages are used to control optional parts of the SCSI protocol such a parity, disconnect/reconnect and many other optional features.

Messages are supported by two additional bus handshaking signals: MSG* and ATN*. MSG* is asserted by the target and allows it to transfer message bytes to/from the initiator. The ATN* signal allows the initiator to request that the target read a message from the initiator (the target always has control of the bus after the selection phase).

Disconnect/Re-Connect and Re-arbitration. In high-performance systems it's possible for the host to interleave accesses to various devices.

Bit / Byte	7	6	5	4	3	2	1	0
0	Operation Code (08h)							
1	Logical Unit Number		(MSB)					
2	Logical Block Address							
3								(LSB)
4	Transfer Length							
5	Control							

Table 1: Example of 6-Byte Command Parameter Block.

For example the host may issue a seek command to one disk drive, then disconnect from that drive and issue a command to another disk drive. It's also possible for devices to reconnect to the original host when data becomes available instead of having the host poll to check if data is ready.

Synchronous mode. To speed up the data transfer the target and initiator can negotiate to operate in a "synchronous" mode. This speeds up the handshaking by allowing a certain number of un-ACK*nowledged REQ* strobes before suspending the data transfer.

Wide mode. The SCSI II standard allows for transfers of 16 or 32 bits in parallel.

Reset. The RESET* signal on the SCSI bus allows a host adapter to reset all of the devices attached to the bus.

Parity Bits. The SCSI bus can use parity bit (DBP*) to attempt to catch errors. Use of parity is optional.

Multiple Initiators. It's possible to have multiple initiators on the same SCSI bus. The selection phase allows arbitration for control of the bus. If there is contention between initiators then the device with the highest SCSI ID gets control.

Non-Standard Commands

There are devices that use the electrical and low-level protocols of the SCSI bus but which use non-standard commands. One reason for this is that the device may not be a good match to any of the ten device models (disk, tape, etc) defined in the SCSI standard. In this case the manufacturer may simply have decided to use the SCSI interface as a high-performance peripheral bus. In other cases the manufacturer uses proprietary commands because it's easier than making their device conform to an existing standard.