# Sun 2 SCSI Interface Architectural Specification


## *ABSTRACT*

The Sun 2 SCSI Interface is an interface to the ANSI standard Small Computer Systems Interface (SCSI) bus. It is intended to be used for communication between the CPU and disks, tape drives, and other peripheral devices.


Implements the SCSI standard at the single master level, single-ended cable, with parity.

Compatible with all known SCSI controllers.

Software is mostly independent of the manufacturer of the controller or of the peripheral device to be used.

Different peripherals may be mixed on the same SCSI bus.

Transfer rates up to 1.2 Mbytes/second.

Data transfer via Multibus DMA (20-bit address, 16-bit data).

Command and status transfer via programmed I/O.

Completion signified by interrupt.

Can wait indefinitely for bus availability without data loss.

### SCSI Compatibility

An SCSI subsystem consists of a *Host Adapter,* a 50-pin cable (the SCSI bus), and one or more *Controllers.* The Host Adapter connects to the CPU's I/O bus (e.g. - the Multibus). The cable, which may be up to 3 meters long, connects the Host Adapter to the Controllers. The Controllers are usually in the same package as the peripheral devices (e.g. - disk drives) which they serve. The interface described here is a Host Adapter.

The SCSI standard defines several features which are optional. It is not necessary for the Host Adapter to implement any of the optional features in order for it to work with all SCSI Controllers. If a controller supports features which the Host Adapter does not implement, it is supposed to not use those features which the Host Adapter does not implement.

The most-commonly-mentioned optional feature is disconnect/reconnect. The Sun 2 Host Adapter does not implement this feature. The main implication of this is that there can only be one outstanding operation being performed at any one time on the SCSI bus. With disconnect/reconnect, transfers may be initiated on several different devices at the same time, and the first one which is ready for data may then use the bus. Without disconnect/reconnect, a transfer is started on a device, and the SCI bus is then dedicated to that device until the transfer has been completed.

While this feature is nice to have, it takes quite a few chips to implement, mostly because there are not any good DMA controller chips available. The consensus was that most applications for the SCSI bus will not need to have overlapped activity of multiple controllers, and that it is better to save money on this interface than to give it full capability. Note that the system will still work with any SCSI controller, but multiple controllers will not be able to take advantage of each other's latency time. When SCSI interface chips and good DMA chips become available, it will be possible to redesign the Host Adapter to support disconnect/reconnect, with only minimal impact on the software drivers.

**Software Simplification**

A major goal of the SCSI standard is to obviate the need to write a new driver for each new peripheral device to be added to the system. To this end, the SCSI spec defines both a hardware bus protocol and a command format. Apart from differences in the storage capacity of different disks, the software needed to interface to one disk should be the same as the software for interfacing to any other disk. The spec even attempts to handle differences in storage capacities in a simple fashion. Disk controllers from different manufacturers should not require different software either, unless an advanced feature (such as data search) of one controller is used, and another controller doesn't have that feature.

So far as is practical, the commands formats for different kinds of devices have been designed to be very similar. Thus the command to read a tape looks much like the command to read a disk. The mechanism whereby the transactions are set up and the completion status is returned is identical from device to device.

**Sun 2 Host Adapter Hardware Features**

Referring to figure 2, the CPU controls the Host Adapter through 7 registers. These registers are mapped into Multibus memory space. The base address for accessing the registers is switch-selectable to any 16 Kbyte boundary.

The data register is used for transferring the data bytes to and from the SCSI bus. It is a 16-bit register on the Multibus side, so that Multibus bandwidth may be effectively utilized. Since the SCSI data bus is only 8 bits, byte packing and unpacking must be done in order to accomplish 16-bit transfers over the Multibus. This is handled automatically. It is possible to perform all transfers 8 bits at a time, without doing any packing or unpacking.

Normally, data transfers are done with DMA. There is a 20-bit DMA address register, which may be written from the Multibus as a lower section of 16 bits and an upper section of 4 bits. It is not possible to read back the contents of the DMA address register. However, there is a 16-bit DMA count register, which is both readable and writeable. The DMA count register is incremented at the same time as the DMA address register, so it is possible to determine how many bytes have been transferred with DMA. Both registers count bytes, and the amount to increment (1 or 2) depends on whether byte packing is enabled (16-bit Multibus transfers) or disabled (8-bit Multibus transfers). The DMA count register does not select or enforce a requested transfer count. The control over how many bytes to transfer for a transaction is done by the controller, as a result of a count field that is part of a command packet. This is not a design decision on our part; it is the way that SCSI is defined.

It is possible for an odd-length data transfer to occur while the Host Adapter is performing 16-bit Multibus transfers. If the data is going from the controller to the host memory, the left-over byte will not be DMA'ed into memory. It will be left in the data register, and may be extracted with programmed I/O. Sufficient information is available for the CPU to easily determine that this has happened. In any case, this should happen only rarely.

The SCSI command/status register is used to send command packets to controllers and to receive status packets that are returned. These packets are transferred one byte at a time with programmed I/O. Since these packets are quite short (on the order of 6 bytes), it was felt to be not worth the expense of separate DMA channels for command and status. The SCSI message protocol may also handled with the command/status register.

The upper half of the Interface Control register provides the ability to control and to sense the state of the SCSI Bus Control lines. In normal operation, the CPU needs to deal with the control lines only when sending commands and receiving status, and then in a very simple fashion. For testing purposes, however, it is possible to explicitly control and sense all of the lines in a detailed fashion. Since disconnect/reconnect is not supported, the SCSI Attention line is not used.

The lower half of Interface Control register is used to enable/disable various functions of the Host Adapter, such as parity, interrupt generation, and 8-bit versus 16-bit data transfers. It

is also used to sense things like interrupt request pending, parity errors, and the presence of a left-over data byte, as mentioned previously.

The Interface Control register register is readable/writeable as one 16-bit register.

**Error Handling**

If a parity error occurs while inputting data, a bit is set in the Interface Control register. At the end of the transaction, the CPU may sense this bit. The bit will stay set until explicitly cleared. Note that parity here refers to parity on the SCSI data bus, and not to parity on system memory. If the parity error occurs while outputting data, it is up to the controller to sense that condition. The controller will then notify the CPU of the parity error when it returns completion status for the transaction.

If a DMA transfer over the Multibus is started, and transfer acknowledge is not received within about 13 microseconds, a bus error condition will occur on the SCSI board. This will result in the release of the Multibus. A bit will be set in the Interface Control register and an interrupt will be signalled if interrupts are enabled. Further SCSI DMA requests will be inhibited to prevent more invalid accesses. The only way to clear this condition is to reset the board, which the CPU may do by writing to a special SCSI register address.

**Interrupts**

The SCSI board interrupts with non-vectored Multibus interrupts. A switch on the board selects the interrupt level as either 3,4,5,or 6. A bit in the Interface control register enables or disables all interrupts from the SCSI board. An interrupt request bit shows the state of the internal interrupt request line, even if interrupts are disabled.

During an SCSI transaction, an interrupt request normally occurs as a result of the controller sending back status at the end of a transaction or if the controller sends a message anytime. However, if DMA requests are disabled, an interrupt request will also occur when the controller is ready for data. Of course, the interrupt enable bit may still be used to prevent the interrupt from actually occurring.

The bus error interrupt described in the previous section is similarly under the control of the interrupt enable bit. The two types of interrupts may be distinguished by respective interrupt request and bus error bits in the Interface Control register.

**Data Rate**

The SCSI hardware protocol allows data rates up to 1.2 MBytes/second. Each byte is transferred with an explicit request/acknowledge handshake. Controllers are designed with local memory, so transfers do not depend on the host being able to sustain a particular data rate. The Sun 2 Host Adapter implements the handshaking in hardware, and subject to Multibus availability, can transfer at the maximum rate. There is a synchronous SCSI protocol defined for higher data rates. The Sun 2 host adapter does not implement this protocol, nor do any controllers known to us at this time.

**Specifications**

Full SCSI compliance at the electrical level: 48 mA single-ended drive capability, standard 50-conductor flat cable, 220/330 ohm termination.

Single-master without disconnect/reconnect.

Parity software selectable.

20-bit address, 16-bit data Multibus DMA for the data portion of the SCSI transfer.

Programmed I/O for command, status, and message transfers.

Interrupts for status, messages, and Multibus errors.

Packaged on a single Multibus card.

**Power**

4.8 mA max at 5 V. (for entire board, which includes a UART, a real-time clock, and battery-backup RAM.)

**Chip Count**

59 chips, 79 16-pin equivalents, including functions mentioned in the power section.

**Cost**

$133.00 for SCSI interface alone. $187.50 for SCSI plus other functions mentioned above.

**Cost-cutting Alternatives**

There are several expensive chips in this design which could be replaced with several cheaper components. This would probable effect a cost savings of about $20-$30, at the expense of a denser board. The Multibus version of this board isn't terribly dense, so this should work reasonably well. However, this design is essentially duplicated on the Sun 2 single board, which is virtually wall-to-wall with chips. There are important advantages to keeping the two designs as similar as possible, which argues against the cost saving moves.