About the SCSI Manager Support for third-party drives

Along with three new routines designed to support the message phases of the SCSI standard, Apple has modified the way the System deals with data transfers across the SCSI bus and the way external boot drives should be formatted.

Transfer Modes

The **SCSI Manager** on the Mac Plus has both a polled and a blind transfer mode. The polled mode is slower, but surer, since it checks the DRQ signal on the SCSI chip before each data transfer and will wait as long as it has to for each byte. The blind mode is half-again as fast as the polled mode but relies on timing constraints to get the data back and forth between the Mac and the external drive. Data has to pass back and forth every two microseconds after the start of a data transfer or the **SCSI Manager** will either read invalid data or write data when the drive isn't ready to receive it, and the data will be lost. If the drive has internal interrupts or some other slow-down that might cause it to miss the two microsecond deadline, a polled transfer mode is the only one you can use reliably.

The situation is a little different on the Mac SE and Mac II-family machines. Here, a hardware handshake was added to make sure the reads and writes stay synchronized. Even though the transfer routines are still called SCSIRBlind and SCSIWBlind they are not blind transfers in the same sense that they were on the Mac Plus. Timing is a factor again because of the synchronization check. In the case of the SE, a second byte has to follow its predecessor within 265 to 284 milliseconds. For a Mac II, the timing interval is about 16 microseconds. In either case, if the interval is exceeded, you'll wind up with a CPU bus error and you'll need to fall back on the polled transfer mode.

Partition Map

Because some systems may need to support a UNIX operating system as well as the Mac OS, a new partition map format should be used on external drives that boot the system through the SCSI bus. While the old format can still be used, Apple is encouraging developers to use this new one.

Byte (Size)	Name	Description
0 (2)	pmSig	Always 0x0504D
2 (2)	pmSigPad	Reserved
4 (4)	pmMapBlkCnt	Total blocks in map
8 (4)	pmPyPartStart	First partition block
C (4)	pmPartBlkCnt	Total partition blocks
10 (32)	pmPartName	Partition name
30 (32)	pmPartType	Partition type
50 (4)	pmLgDataStart	Data area's 1st logical block
54 (4)	pmDataCnt	Total data area blocks
5C (4)	pkmLgBootStart	Boot code's 1st logical block
60 (4)	pklmBootSize	Boot code size in bytes
64 (4)	pmBootLoad	Boot code load address
68 (4)	pmBootLoad2	More boot load data
6C (4)	pmBootEntry	Boot code entry point
70 (4)	pmBootEntry2	More boot code entry data
74 (4)	pmBootCksum	Boot code checksum
78 (16)	pmProcessor	Processor type
88 (128)	arg	Boot-specific arguments

When using the new partition map format, bytes 0, 4, 8, C, 10 and 30 are the ones currently used. The **Start Manager** will expect to see these fields and they should be defined as indicated above. The rest of the fields may or may not be used but they've been defined as indicated to give you a standard way to store driver and operating system information.

Regardless of the format you choose for the partition map, however, there is no change in the driver descriptor map.

Error Codes

There is an expanded list of error codes that pertain to bus-specific conditions:

scArbNBErr	(3)	Arbitration failure during SCSIGet; bus busy
scMgrBusyErr	(7)	SCSI Manager already occupied when SCSIGet was called

scSequenceErr	(8)	Operation out of sequence
scBusTOErr	(9)	Bus timeout before data ready on blind read or write
scComplPhaseErr	(10)	Bus not in Status phase; SCSIComplete call failed

In order, scArbNBErr is returned when arbitration fails because the bus is already busy when your program calls the SCSIGet routine. On the other hand, if it is the **SCSI Manager** (rather than the bus), that's busy when you call SCSIGet, the error message is scMgrBusyErr.

Calling routines out of order produces a scSequenceErr result code.

The **SCSI Manager** produces a ScBusTOErr message if the data isn't ready within the timeout period of a SCSIRBlind or SCSIWBlind routine, and it returns scComplPhaseErr if you attempt a SCSIComplete routine when the bus is out of the Status phase. What the last error code tells you is that either the filler bytes were written or bytes were read and lost.

Things to Remember

If you are writing a program to partition a disk you're reminded of the following:
1) With the sole exception of physical block 0, every block on the disk has to belong to a partition. Unused blocks have a partition type of their own, called Apple_Free. The other types of partitions include Apple_MFS for flat file system; Apple_HFS for hierarchical file system; Apple_Unix_SVR2 for UNIX; Apple_partition_map for the special partition that holds the partition map; Apple_Driver for the partition that contains a device driver; Apple_PRODOS for a partition designated for an Apple IIGS; and Apple_Scratch for partitions that are empty.

2) There has to be a partition map for every partition, including the partition that holds the partition map.

3) Device drivers all get their own individual partitions instead of being in the associated operating system's partition.

4) Repartitioning involves two separate steps. Existing partitions are first marked as Apple_Free partitions (during which process you set the first eight blocks to 0 after first copying their contents somewhere else). After that, you combine the existing partitions with contiguous free partitions to create a single, larger, area.

5) You can shift existing partitions towards the end of the disk if you need more room for the partition map as a result of repartitioning. The partition map is a special case. Its partition is the only one that can be expanded without damaging or destroying its contents.