SCSI Manager Reference

(Not Recommended)

Carbon > Resource Management

2006-07-12

Contents

SCSI Manager Reference (Not Recommended)

Framework:	CoreServices/CoreServices.h
Declared in	SCSI.h

Overview

Carbon supports only the `SCSIAction` function in the SCSI Manager, although this function is no longer recommended. For Mac OS X, the I/O Kit should be used to support more complex SCSI devices.

Important: The SCSI Manager is deprecated in Mac OS X v10.2 and later. You should use I/O Kit to support SCSI devices instead. For more information, see *SCSI Architecture Model Device Interface Guide*.

Callbacks

SCSICallbackProcPtr

Defines a pointer to a completion routine.

```
typedef void (*SCSICallbackProcPtr) (
    void * scsiPB
);
```

If you name your function `MySCSICallbackProc`, you would declare it like this:

```
void MySCSICallbackProc (
    void * scsiPB
);
```

Availability
Available in Mac OS X v10.0 and later.
Not available to 64-bit applications.

Declared In
`SCSI.h`

Data Types

CDB

You use the command descriptor block record to pass SCSI commands to the `SCSIAction` function.

```
union CDB {
    BytePtr cdbPtr;
    UInt8 cdbBytes[16];
};
typedef union CDB CDB;
typedef CDB * CDBPtr;
```

Fields

`cdbPtr`

A pointer to a buffer containing a CDB.

`cdbBytes`

A buffer in which you can place a CDB.

Discussion

The SCSI commands can be stored within this structure, or you can provide a pointer to them. You set the `scsiCDBIsPointer` flag in the SCSI parameter block if this record contains a pointer.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

`SCSI.h`

DeviceIdent

You use the device identification record to specify a target device by its bus, SCSI ID, and logical unit number (LUN).

```
struct DeviceIdent {
    UInt8 diReserved;
    UInt8 bus;
    UInt8 targetID;
    UInt8 LUN;
};
typedef struct DeviceIdent DeviceIdent;
```

Fields

`diReserved`

Reserved.

`bus`

The bus number of the SIM/HBA for the target device.

`targetID`

The SCSI ID number of the target device.

`LUN`
: The target LUN, or 0 if the device does not support logical units.

Availability
Available in Mac OS X v10.0 and later.
Not available to 64-bit applications.

Declared In
`SCSI.h`

DeviceIdentATA

```
struct DeviceIdentATA {
    UInt8 diReserved;
    UInt8 busNum;
    UInt8 devNum;
    UInt8 diReserved2;
};
typedef struct DeviceIdentATA DeviceIdentATA;
```

Availability
Available in Mac OS X v10.1 and later.
Not available to 64-bit applications.

Declared In
`SCSI.h`

SCSI_PB

You use the SCSI Manager parameter block to pass information to the `SCSIAction` function.

```
struct SCSI_PB {
    SCSIHdr * qLink;
    short scsiReserved1;
    UInt16 scsiPBLength;
    UInt8 scsiFunctionCode;
    UInt8 scsiReserved2;
    volatile OSErr scsiResult;
    DeviceIdent scsiDevice;
    SCSICallbackUPP scsiCompletion;
    UInt32 scsiFlags;
    BytePtr scsiDriverStorage;
    Ptr scsiXPTprivate;
    long scsiReserved3;
};
typedef struct SCSI_PB SCSI_PB;
```

Fields

`qLink`
: A pointer to the next entry in the request queue. This field is used internally by the SCSI Manager and must be set to 0 when the parameter block is initialized. The SCSI Manager functions always set this field to 0 before returning, so you do not need to set it to 0 again before reusing a parameter block.

`scsiReserved1`
: Reserved.

`scsiPBLength`
: The size of the parameter block, in bytes, including the parameter block header.

`scsiFunctionCode`
: A function selector code that specifies the service being requested.

`scsiReserved2`
: Reserved.

`scsiResult`
: The result code returned by the XPT or SIM when the function completes. The value `scsiRequestInProgress` indicates that the request is still in progress or queued.

`scsiDevice`
: A 4-byte value that uniquely identifies the target device for a request. The `DeviceIdent` data type designates the bus number, target SCSI ID, and logical unit number (LUN).

`scsiCompletion`
: A pointer to a completion routine.

`scsiFlags`
: Flags indicating the transfer direction and any special handling required for this request.

`scsiDriverStorage`
: A pointer to the device driver's private storage. This field is not affected or used by the SCSI Manager.

`scsiXPTprivate`
: Private field for use in XPT.

`scsiReserved3`
: Reserved.

Availability
Available in Mac OS X v10.0 and later.
Not available to 64-bit applications.

Declared In
`SCSI.h`

SCSICallbackUPP

Defines a universal procedure pointer (UPP) to a completion routine.

```
typedef SCSICallbackProcPtr SCSICallbackUPP;
```

Availability
Available in Mac OS X v10.0 and later.
Not available to 64-bit applications.

Declared In
`SCSI.h`

SCSI_IO

Defines the SCSI I/O parameter block.

```
struct SCSI_IO {
    SCSIHdr * qLink;
    short scsiReserved1;
    UInt16 scsiPBLength;
    UInt8 scsiFunctionCode;
    UInt8 scsiReserved2;
    volatile OSErr scsiResult;
    DeviceIdent scsiDevice;
    SCSICallbackUPP scsiCompletion;
    UInt32 scsiFlags;
    BytePtr scsiDriverStorage;
    Ptr scsiXPTprivate;
    long scsiReserved3;
    UInt16 scsiResultFlags;
    UInt16 scsiReserved3pt5;
    BytePtr scsiDataPtr;
    UInt32 scsiDataLength;
    BytePtr scsiSensePtr;
    UInt8 scsiSenseLength;
    UInt8 scsiCDBLength;
    UInt16 scsiSGListCount;
    UInt32 scsiReserved4;
    UInt8 scsiSCSIstatus;
    SInt8 scsiSenseResidual;
    UInt16 scsiReserved5;
    long scsiDataResidual;
    CDB scsiCDB;
    long scsiTimeout;
    BytePtr scsiReserved5pt5;
    UInt16 scsiReserved5pt6;
    UInt16 scsiIOFlags;
    UInt8 scsiTagAction;
    UInt8 scsiReserved6;
    UInt16 scsiReserved7;
    UInt16 scsiSelectTimeout;
    UInt8 scsiDataType;
    UInt8 scsiTransferType;
    UInt32 scsiReserved8;
    UInt32 scsiReserved9;
    UInt16 scsiHandshake[8];
    UInt32 scsiReserved10;
    UInt32 scsiReserved11;
    SCSI_IO * scsiCommandLink;
    UInt8 scsiSIMpublics[8];
    UInt8 scsiAppleReserved6[8];
    UInt16 scsiCurrentPhase;
    short scsiSelector;
    OSErr scsiOldCallResult;
    UInt8 scsiSCSImessage;
    UInt8 XPTprivateFlags;
    UInt8 XPTextras[12];
};
typedef struct SCSI_IO SCSI_IO;
typedef SCSI_IO SCSIExecIOPB;
```

Fields

`qLink`
: A pointer to the next entry in the request queue. This field is used internally by the SCSI Manager and must be set to 0 when the parameter block is initialized. The SCSI Manager functions always set this field to 0 before returning, so you do not need to set it to 0 again before reusing a parameter block.

`scsiReserved1`
: Reserved.

`scsiPBLength`
: The size of the parameter block, in bytes, including the parameter block header.

`scsiFunctionCode`
: A function selector code that specifies the service being requested.

`scsiReserved2`
: Reserved.

`scsiResult`
: The result code returned by the XPT or SIM when the function completes. The value `scsiRequestInProgress` indicates that the request is still in progress or queued.

`scsiDevice`
: A 4-byte value that uniquely identifies the target device for a request. The `DeviceIdent` data type designates the bus number, target SCSI ID, and logical unit number (LUN).

`scsiCompletion`
: A pointer to a completion routine.

`scsiFlags`
: Flags indicating the transfer direction and any special handling required for this request.

`scsiDriverStorage`
: A pointer to the device driver's private storage. This field is not affected or used by the SCSI Manager.

`scsiXPTprivate`
: Private field for use in XPT.

`scsiReserved3`
: Reserved.

`scsiResultFlags`
: Output flags that modify the `scsiResult` field.

`scsiReserved3pt5`
: Reserved.

`scsiDataPtr`
: A pointer to a data buffer or scatter/gather list. You specify the data type using the `scsiDataType` field.

`scsiDataLength`
: The amount of data to be transferred, in bytes.

`scsiSensePtr`
: A pointer to the autosense data buffer. If autosense is enabled (the `scsiDisableAutosense` flag is not set), the SCSI Manager returns `REQUEST SENSE` information in this buffer.

`scsiSenseLength`
: The size of the autosense data buffer, in bytes.

`scsiCDBLength`
: The length of the SCSI command descriptor block, in bytes.

`scsiSGListCount`
: The number of elements in the scatter/gather list.

`scsiReserved4`
: Reserved.

`scsiSCSIstatus`
: The status returned by the SCSI device.

`scsiSenseResidual`
: The automatic `REQUEST SENSE` residual length (that is, the number of bytes that were expected but not transferred). This number is negative if extra bytes had to be transferred to force the target off of the bus.

`scsiReserved5`
: Reserved for output.

`scsiDataResidual`
: The data transfer residual length (that is, the number of bytes that were expected but not transferred). This number is negative if extra bytes had to be transferred to force the target off the bus.

`scsiCDB`
: This field can contain either the actual CDB or a pointer to the CDB. You set the `scsiCDBIsPointer` flag if this field contains a pointer.

`scsiTimeout`
: The length of time the SIM should allow before reporting a timeout of the SCSI bus. The time value is represented in Time Manager format (positive values for milliseconds, negative values for microseconds). The timer is started when the I/O request is sent to the target. If the request does not complete within the specified time, the SIM attempts to issue an `ABORT` message, either by reselecting the device or by asserting the attention (/ATN) signal. A value of 0 specifies the default timeout for the SIM. The default timeout for the SCSI Manager 4.3 SIM is infinite (that is, no timeout).

`scsiReserved5pt5`
: Reserved.

`scsiReserved5pt6`
: Reserved.

`scsiIOFlags`
: Additional I/O flags describing the data transfer.

`scsiTagAction`
: Reserved.

`scsiReserved6`
: Reserved for input.

`scsiReserved7`
: Reserved for input.

`scsiSelectTimeout`
: An optional SELECT timeout value, in milliseconds. The default is 250 ms, as specified by SCSI-2. The accuracy of this period is dependent on the HBA.A value of 0 specifies the default timeout. Some SIMs ignore this parameter and always use a value of 250 ms.

`scsiDataType`
: The data type pointed to by the `scsiDataPtr` field.

`scsiTransferType`
: The type of transfer mode to use during the data phase.

`scsiReserved8`
: Reserved for input.

`scsiReserved9`
: Reserved for input.

`scsiHandshake`
: Handshaking instructions for blind transfers, consisting of an array of word values, terminated by 0. The SIM polls for data ready after transferring the amount of data specified in each successive `scsiHandshake` entry. When it encounters a 0 value, the SIM starts over at the beginning of the list. Handshaking always starts from the beginning of the list every time a device transitions to data phase.

`scsiReserved10`
: Reserved for input.

`scsiReserved11`
: Reserved for input.

`scsiCommandLink`
: A pointer to a linked parameter block. This field provides support for SCSI linked commands. This optional feature ensures that a set of commands sent to a device are executed in sequential order without interference from other applications. You create a list of commands using this pointer to link additional parameter blocks. Each parameter block except the last should have the `scsiCDBLinked` flag set in the `scsiFlags` field. A `CHECK CONDITION` status from the device will abort linked command execution. Linked commands may not be supported by all SIMs.

`scsiSIMpublics`
: An additional input field available for use by SIM developers.

`scsiCurrentPhase`
: The current SCSI bus phase reported by the SIM after handling an original SCSI Manager function. This field is used only by the XPT and SIM during original SCSI Manager emulation.

`scsiSelector`
: The function selector code that was passed to the `_SCSIDispatch` trap during original SCSI Manager emulation. The SIM uses this field to determine which original SCSI Manager function to perform.

`scsiOldCallResult`
: The result code from an emulated original SCSI Manager function. The SIM returns results to all original SCSI Manager functions in this field, except for the `SCSIComplete` result, which it returns in `scsiResult`.

`scsiSCSImessage`
: The message byte returned by an emulated `SCSIComplete` function. This field is only used by the XPT and SIM during original SCSI Manager emulation.

`XPTprivateFlags`
: Reserved.

`XPTextras`
: Reserved.

Availability
Available in Mac OS X v10.0 and later.
Not available to 64-bit applications.

Declared In
`SCSI.h`

SCSIBusInquiryPB

Defines a SCSI bus inquiry parameter block.

```
struct SCSIBusInquiryPB {
    SCSIHdr * qLink;
    short scsiReserved1;
    UInt16 scsiPBLength;
    UInt8 scsiFunctionCode;
    UInt8 scsiReserved2;
    volatile OSErr scsiResult;
    DeviceIdent scsiDevice;
    SCSICallbackUPP scsiCompletion;
    UInt32 scsiFlags;
    BytePtr scsiDriverStorage;
    Ptr scsiXPTprivate;
    long scsiReserved3;
    UInt16 scsiEngineCount;
    UInt16 scsiMaxTransferType;
    UInt32 scsiDataTypes;
    UInt16 scsiIOpbSize;
    UInt16 scsiMaxIOpbSize;
    UInt32 scsiFeatureFlags;
    UInt8 scsiVersionNumber;
    UInt8 scsiHBAInquiry;
    UInt8 scsiTargetModeFlags;
    UInt8 scsiScanFlags;
    UInt32 scsiSIMPrivatesPtr;
    UInt32 scsiSIMPrivatesSize;
    UInt32 scsiAsyncFlags;
    UInt8 scsiHiBusID;
    UInt8 scsiInitiatorID;
    UInt16 scsiBIReserved0;
    UInt32 scsiBIReserved1;
    UInt32 scsiFlagsSupported;
    UInt16 scsiIOFlagsSupported;
    UInt16 scsiWeirdStuff;
    UInt16 scsiMaxTarget;
    UInt16 scsiMaxLUN;
    char scsiSIMVendor[16];
    char scsiHBAVendor[16];
    char scsiControllerFamily[16];
    char scsiControllerType[16];
    char scsiXPTversion[4];
    char scsiSIMversion[4];
    char scsiHBAversion[4];
    UInt8 scsiHBAslotType;
    UInt8 scsiHBAslotNumber;
    UInt16 scsiSIMsRsrcID;
    UInt16 scsiBIReserved3;
    UInt16 scsiAdditionalLength;
};
typedef struct SCSIBusInquiryPB SCSIBusInquiryPB;
```

Fields

`qLink`

A pointer to the next entry in the request queue. This field is used internally by the SCSI Manager and must be set to 0 when the parameter block is initialized. The SCSI Manager functions always set this field to 0 before returning, so you do not need to set it to 0 again before reusing a parameter block.

`scsiReserved1`

Reserved.

`scsiPBLength`
: The size of the parameter block, in bytes, including the parameter block header.

`scsiFunctionCode`
: A function selector code that specifies the service being requested.

`scsiReserved2`
: Reserved.

`scsiResult`
: The result code returned by the XPT or SIM when the function completes. The value `scsiRequestInProgress` indicates that the request is still in progress or queued.

`scsiDevice`
: A 4-byte value that uniquely identifies the target device for a request. The `DeviceIdent` data type designates the bus number, target SCSI ID, and logical unit number (LUN).

`scsiCompletion`
: A pointer to a completion routine.

`scsiFlags`
: Flags indicating the transfer direction and any special handling required for this request.

`scsiDriverStorage`
: A pointer to the device driver's private storage. This field is not affected or used by the SCSI Manager.

`scsiXPTprivate`
: Private field for use in XPT.

`scsiReserved3`
: Reserved.

`scsiEngineCount`
: The number of engines on the HBA. This value is 0 for a built-in SCSI bus.

`scsiMaxTransferType`
: The number of data transfer types available on the HBA.

`scsiDataTypes`
: A bit mask describing the data types supported by the SIM/HBA. Bits 3 through 15 and bit 31 are reserved by Apple Computer, Inc. Bits 16 through 30 are available for use by SIM developers.

`scsiIOpbSize`
: The minimum size of a SCSI I/O parameter block for this SIM.

`scsiMaxIOpbSize`
: The minimum size of a SCSI I/O parameter block for all currently registered SIMs. That is, the largest registered `scsiIOpbSize`.

`scsiFeatureFlags`
: These flags describe various physical characteristics of the SCSI bus.

`scsiVersionNumber`
: The version number of the SIM/HBA.

`scsiHBAInquiry`
: Flags describing the capabilities of the bus.

`scsiTargetModeFlags`
: Reserved.

`scsiScanFlags`
: Reserved.

`scsiSIMPrivatesPtr`
: A pointer to the SIM's private storage.

`scsiSIMPrivatesSize`
: The size of the SIM's private storage, in bytes.

`scsiAsyncFlags`
: Reserved.

`scsiHiBusID`
: The highest bus number currently registered with the XPT. If no buses are registered, this field contains 0xFF (the ID of the XPT).

`scsiInitiatorID`
: The SCSI ID of the HBA. This value is 7 for a built-in SCSI bus.

`scsiBIReserved0`

`scsiBIReserved1`

`scsiFlagsSupported`
: A bit mask that defines which `scsiFlags` bits are supported.

`scsiIOFlagsSupported`
: A bit mask that defines which `scsiIOFlags` bits are supported.

`scsiWeirdStuff`
: Flags that identify unusual aspects of a SIM's operation.

`scsiMaxTarget`
: The highest SCSI ID value supported by the HBA.

`scsiMaxLUN`
: The highest logical unit number supported by the HBA.

`scsiSIMVendor`
: An ASCII text string that identifies the SIM vendor. This field returns `'Apple Computer'` for a built-in SCSI bus.

`scsiHBAVendor`
: An ASCII text string that identifies the HBA vendor. This field returns `'Apple Computer'` for a built-in SCSI bus.

`scsiControllerFamily`
: An optional ASCII text string that identifies the family of parts to which the SCSI controller chip belongs. This information is provided at the discretion of the HBA vendor.

`scsiControllerType`
: An optional ASCII text string that identifies the specific type of SCSI controller chip. This information is provided at the discretion of the HBA vendor.

`scsiXPTversion`
: An ASCII text string that identifies the version number of the XPT. You should use the other fields of this parameter block to check for specific features, rather than relying on this value.

`scsiSIMversion`
: An ASCII text string that identifies the version number of the SIM. You should use the other fields of this parameter block to check for specific features, rather than relying on this value.

`scsiHBAversion`
: An ASCII text string that identifies the version number of the HBA. You should use the other fields of this parameter block to check for specific features, rather than relying on this value.

`scsiHBAslotType`
: The slot type, if any, used by this HBA.

`scsiHBAslotNumber`
: The slot number for the SIM. Device drivers should copy this value into the `dCtlSlot` field of the device control entry. This value is 0 for a built-in SCSI bus.

`scsiSIMsRsrcID`
: The ID for the SIM. Device drivers should copy this value into the `dCtlSlotID` field of the device control entry. This value is 0 for a built-in SCSI bus.

`scsiAdditionalLength`
: The additional size of this parameter block, in bytes. If this structure includes extra fields to return additional information, this field contains the number of additional bytes.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

`SCSI.h`

SCSIAbortCommandPB

Defines a SCSI abort command parameter block.

```
struct SCSIAbortCommandPB {
    SCSIHdr * qLink;
    short scsiReserved1;
    UInt16 scsiPBLength;
    UInt8 scsiFunctionCode;
    UInt8 scsiReserved2;
    volatile OSErr scsiResult;
    DeviceIdent scsiDevice;
    SCSICallbackUPP scsiCompletion;
    UInt32 scsiFlags;
    BytePtr scsiDriverStorage;
    Ptr scsiXPTprivate;
    long scsiReserved3;
    SCSI_IO * scsiIOptr;
};
typedef struct SCSIAbortCommandPB SCSIAbortCommandPB;
```

Fields

`qLink`
: A pointer to the next entry in the request queue. This field is used internally by the SCSI Manager and must be set to 0 when the parameter block is initialized. The SCSI Manager functions always set this field to 0 before returning, so you do not need to set it to 0 again before reusing a parameter block.

`scsiReserved1`
: Reserved.

`scsiPBLength`
: The size of the parameter block, in bytes, including the parameter block header.

`scsiFunctionCode`
: A function selector code that specifies the service being requested.

`scsiReserved2`
: Reserved.

`scsiResult`
: The result code returned by the XPT or SIM when the function completes. The value `scsiRequestInProgress` indicates that the request is still in progress or queued.

`scsiDevice`
: A 4-byte value that uniquely identifies the target device for a request. The `DeviceIdent` data type designates the bus number, target SCSI ID, and logical unit number (LUN).

`scsiCompletion`
: A pointer to a completion routine.

`scsiFlags`
: Flags indicating the transfer direction and any special handling required for this request.

`scsiDriverStorage`
: A pointer to the device driver's private storage. This field is not affected or used by the SCSI Manager.

`scsiXPTprivate`
: Private field for use in XPT.

`scsiReserved3`
: Reserved.

`scsiIOptr`
: A pointer to the parameter block to be canceled.

Availability
Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
`SCSI.h`

SCSITerminateIOPB

Defines a SCSI terminate I/O parameter block.

```
struct SCSITerminateIOPB {
    SCSIHdr * qLink;
    short scsiReserved1;
    UInt16 scsiPBLength;
    UInt8 scsiFunctionCode;
    UInt8 scsiReserved2;
    volatile OSErr scsiResult;
    DeviceIdent scsiDevice;
    SCSICallbackUPP scsiCompletion;
    UInt32 scsiFlags;
    BytePtr scsiDriverStorage;
    Ptr scsiXPTprivate;
    long scsiReserved3;
    SCSI_IO * scsiIOptr;
};
typedef struct SCSITerminateIOPB SCSITerminateIOPB;
```

Fields

`qLink`
: A pointer to the next entry in the request queue. This field is used internally by the SCSI Manager and must be set to 0 when the parameter block is initialized. The SCSI Manager functions always set this field to 0 before returning, so you do not need to set it to 0 again before reusing a parameter block.

`scsiReserved1`
: Reserved.

`scsiPBLength`
: The size of the parameter block, in bytes, including the parameter block header.

`scsiFunctionCode`
: A function selector code that specifies the service being requested.

`scsiReserved2`
: Reserved.

`scsiResult`
: The result code returned by the XPT or SIM when the function completes. The value `scsiRequestInProgress` indicates that the request is still in progress or queued.

`scsiDevice`
: A 4-byte value that uniquely identifies the target device for a request. The `DeviceIdent` data type designates the bus number, target SCSI ID, and logical unit number (LUN).

`scsiCompletion`
: A pointer to a completion routine.

`scsiFlags`
: Flags indicating the transfer direction and any special handling required for this request.

`scsiDriverStorage`
: A pointer to the device driver's private storage. This field is not affected or used by the SCSI Manager.

`scsiXPTprivate`
: Private field for use in XPT.

`scsiReserved3`
: Reserved.

`scsiIOptr`
: A pointer to the parameter block to be canceled.

Availability
Available in Mac OS X v10.0 and later.
Not available to 64-bit applications.

Declared In
`SCSI.h`

SCSIGetVirtualIDInfoPB

Defines a SCSI virtual ID information parameter block.

```
struct SCSIGetVirtualIDInfoPB {
    SCSIHdr * qLink;
    short scsiReserved1;
    UInt16 scsiPBLength;
    UInt8 scsiFunctionCode;
    UInt8 scsiReserved2;
    volatile OSErr scsiResult;
    DeviceIdent scsiDevice;
    SCSICallbackUPP scsiCompletion;
    UInt32 scsiFlags;
    Ptr scsiDriverStorage;
    Ptr scsiXPTprivate;
    long scsiReserved3;
    UInt16 scsiOldCallID;
    Boolean scsiExists;
    SInt8 filler;
};
typedef struct SCSIGetVirtualIDInfoPB SCSIGetVirtualIDInfoPB;
```

Fields

`qLink`
A pointer to the next entry in the request queue. This field is used internally by the SCSI Manager and must be set to 0 when the parameter block is initialized. The SCSI Manager functions always set this field to 0 before returning, so you do not need to set it to 0 again before reusing a parameter block.

`scsiReserved1`
Reserved.

`scsiPBLength`
The size of the parameter block, in bytes, including the parameter block header.

`scsiFunctionCode`
A function selector code that specifies the service being requested.

`scsiReserved2`
Reserved.

`scsiResult`
The result code returned by the XPT or SIM when the function completes. The value `scsiRequestInProgress` indicates that the request is still in progress or queued.

`scsiDevice`
A 4-byte value that uniquely identifies the target device for a request. The `DeviceIdent` data type designates the bus number, target SCSI ID, and logical unit number (LUN).

`scsiCompletion`
A pointer to a completion routine.

`scsiFlags`
: Flags indicating the transfer direction and any special handling required for this request.

`scsiDriverStorage`
: A pointer to the device driver's private storage. This field is not affected or used by the SCSI Manager.

`scsiXPTprivate`
: Private field for use in XPT.

`scsiReserved3`
: Reserved.

`scsiOldCallID`
: The virtual SCSI ID of the device you are searching for.

`scsiExists`
: The XPT returns true in this field if the `scsiDevice` field contains a valid device identification record.

Availability

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In

`SCSI.h`

SCSILoadDriverPB

Defines a SCSI load driver parameter block.

```
struct SCSILoadDriverPB {
    SCSIHdr * qLink;
    short scsiReserved1;
    UInt16 scsiPBLength;
    UInt8 scsiFunctionCode;
    UInt8 scsiReserved2;
    volatile OSErr scsiResult;
    DeviceIdent scsiDevice;
    SCSICallbackUPP scsiCompletion;
    UInt32 scsiFlags;
    Ptr scsiDriverStorage;
    Ptr scsiXPTprivate;
    long scsiReserved3;
    short scsiLoadedRefNum;
    Boolean scsiDiskLoadFailed;
    SInt8 filler;
};
typedef struct SCSILoadDriverPB SCSILoadDriverPB;
```

Fields

`qLink`
: A pointer to the next entry in the request queue. This field is used internally by the SCSI Manager and must be set to 0 when the parameter block is initialized. The SCSI Manager functions always set this field to 0 before returning, so you do not need to set it to 0 again before reusing a parameter block.

`scsiReserved1`
: Reserved.

`scsiPBLength`
: The size of the parameter block, in bytes, including the parameter block header.

`scsiFunctionCode`
: A function selector code that specifies the service being requested.

`scsiReserved2`
: Reserved.

`scsiResult`
: The result code returned by the XPT or SIM when the function completes. The value `scsiRequestInProgress` indicates that the request is still in progress or queued.

`scsiDevice`
: A 4-byte value that uniquely identifies the target device for a request. The `DeviceIdent` data type designates the bus number, target SCSI ID, and logical unit number (LUN).

`scsiCompletion`
: A pointer to a completion routine.

`scsiFlags`
: Flags indicating the transfer direction and any special handling required for this request.

`scsiDriverStorage`
: A pointer to the device driver's private storage. This field is not affected or used by the SCSI Manager.

`scsiXPTprivate`
: Private field for use in XPT.

`scsiReserved3`
: Reserved.

`scsiLoadedRefNum`
: If the driver is successfully loaded, this field contains the driver reference number returned by the SIM.

`scsiDiskLoadFailed`
: If this field is set to true, the SIM should attempt to load its own driver regardless of whether there is one on the device. If this field is set to false, the SIM has the option of loading a driver from the device or using one of its own.

Availability
Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
`SCSI.h`

SCSIDriverPB

Defines a SCSI driver identification parameter block.

```
struct SCSIDriverPB {
    SCSIHdr * qLink;
    short scsiReserved1;
    UInt16 scsiPBLength;
    UInt8 scsiFunctionCode;
    UInt8 scsiReserved2;
    volatile OSErr scsiResult;
    DeviceIdent scsiDevice;
    SCSICallbackUPP scsiCompletion;
    UInt32 scsiFlags;
    Ptr scsiDriverStorage;
    Ptr scsiXPTprivate;
    long scsiReserved3;
    short scsiDriver;
    UInt16 scsiDriverFlags;
    DeviceIdent scsiNextDevice;
};
typedef struct SCSIDriverPB SCSIDriverPB;
```

Fields

`qLink`

A pointer to the next entry in the request queue. This field is used internally by the SCSI Manager and must be set to 0 when the parameter block is initialized. The SCSI Manager functions always set this field to 0 before returning, so you do not need to set it to 0 again before reusing a parameter block.

`scsiReserved1`

Reserved.

`scsiPBLength`

The size of the parameter block, in bytes, including the parameter block header.

`scsiFunctionCode`

A function selector code that specifies the service being requested.

`scsiReserved2`

Reserved.

`scsiResult`

The result code returned by the XPT or SIM when the function completes. The value `scsiRequestInProgress` indicates that the request is still in progress or queued.

`scsiDevice`

A 4-byte value that uniquely identifies the target device for a request. The `DeviceIdent` data type designates the bus number, target SCSI ID, and logical unit number (LUN).

`scsiCompletion`

A pointer to a completion routine.

`scsiFlags`

Flags indicating the transfer direction and any special handling required for this request.

`scsiDriverStorage`

A pointer to the device driver's private storage. This field is not affected or used by the SCSI Manager.

`scsiXPTprivate`

`scsiReserved3`

Reserved.

`scsiDriver`

The driver reference number of the device driver associated with this device identification record.

`scsiDriverFlags`
: Driver information flags. These flags are not interpreted by the XPT but can be used to provide information about the driver to other clients.

`scsiNextDevice`
: The device identification record of the next device in the driver registration list.

Availability
Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared In
`SCSI.h`

Constants

SCSI Flags

Used in the `scsiFlags` field of the `SCSI_PB` structure.

```
enum {
    scsiDirectionMask = 0xC0000000,
    scsiDirectionNone = 0xC0000000,
    scsiDirectionReserved = 0x00000000,
    scsiDirectionOut = 0x80000000,
    scsiDirectionIn = 0x40000000,
    scsiDisableAutosense = 0x20000000,
    scsiFlagReservedA = 0x10000000,
    scsiFlagReserved0 = 0x08000000,
    scsiCDBLinked = 0x04000000,
    scsiQEnable = 0x02000000,
    scsiCDBIsPointer = 0x01000000,
    scsiFlagReserved1 = 0x00800000,
    scsiInitiateSyncData = 0x00400000,
    scsiDisableSyncData = 0x00200000,
    scsiSIMQHead = 0x00100000,
    scsiSIMQFreeze = 0x00080000,
    scsiSIMQNoFreeze = 0x00040000,
    scsiDoDisconnect = 0x00020000,
    scsiDontDisconnect = 0x00010000,
    scsiDataReadyForDMA = 0x00008000,
    scsiFlagReserved3 = 0x00004000,
    scsiDataPhysical = 0x00002000,
    scsiSensePhysical = 0x00001000,
    scsiFlagReserved5 = 0x00000800,
    scsiFlagReserved6 = 0x00000400,
    scsiFlagReserved7 = 0x00000200,
    scsiFlagReserved8 = 0x00000100
};
```

Constants

`scsiDirectionMask`

A bit field that specifies transfer direction, using these constants: `scsiDirectionIn`, `scsiDirectionOut`, and `scsiDirectionNone`

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiDirectionNone`

No data phase expected.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiDirectionOut`

Data out.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiDirectionIn`
Data in.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiDisableAutosense`
Disable the automatic REQUEST SENSE feature.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiCDBLinked`
The parameter block contains a linked CDB. This option may not be supported by all SIMs.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiQEnable`
Enable target queue actions. This option may not be supported by all SIMs.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiCDBIsPointer`
Set if the `scsiCDB` field of a SCSI I/O parameter block contains a pointer. If clear, the `scsiCDB` field contains the actual CDB. In either case, the `scsiCDBLength` field contains the number of bytes in the SCSI command descriptor block.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiInitiateSyncData`
Set if the SIM should attempt to initiate a synchronous data transfer by sending the SDTR message. If successful, the device normally remains in the synchronous transfer mode until it is reset or until you specify asynchronous mode by setting the `scsiDisableSyncData` flag. Because SDTR negotiation occurs every time this flag is set, you should set it only when negotiation is actually needed.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiDisableSyncData`
Disable synchronous data transfer. The SIM sends an SDTR message with a REQ/ACK offset of 0 to indicate asynchronous data transfer mode. You should set this flag only when negotiation is actually needed.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiSIMQHead`
: Place the parameter block at the head of the SIM queue. This can be used to insert error handling at the head of a frozen queue.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiSIMQFreeze`
: Freeze the SIM queue after completing this transaction.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiSIMQNoFreeze`
: Disable SIM queue freezing for this transaction.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiDoDisconnect`
: Explicitly allow device to disconnect.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiDontDisconnect`
: Explicitly prohibit device disconnection. If this flag and the `scsiDoDisconnect` flag are both 0, the SIM determines whether to allow or prohibit disconnection, based on performance criteria.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiDataReadyForDMA`
: Data buffer is locked and non-cacheable.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiDataPhysical`
: Data buffer address is physical.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiSensePhysical`
: Autosense data pointer is physical.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

SCSIAction function selector codes

Used in the `scsiFunctionCode` field of the parameter block passed to the `SCSIAction` function.

```
enum {
    SCSINop = 0x00,
    SCSIExecIO = 0x01,
    SCSIBusInquiry = 0x03,
    SCSIReleaseQ = 0x04,
    SCSIAbortCommand = 0x10,
    SCSIResetBus = 0x11,
    SCSIResetDevice = 0x12,
    SCSITerminateIO = 0x13
};
enum {
    SCSIGetVirtualIDInfo = 128,
    SCSILoadDriver = 130,
    SCSIOldCall = 132,
    SCSICreateRefNumXref = 133,
    SCSILookupRefNumXref = 134,
    SCSIRemoveRefNumXref = 135,
    SCSIRegisterWithNewXPT = 136
};
```

Constants

`SCSINop`

No operation.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`SCSIExecIO`

Execute a SCSI I/O transaction.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`SCSIBusInquiry`

Bus inquiry.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`SCSIReleaseQ`

Release a frozen SIM queue.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`SCSIAbortCommand`
: Abort a SCSI command.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`SCSIResetBus`
: Reset the SCSI bus.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`SCSIResetDevice`
: Reset a SCSI device.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`SCSITerminateIO`
: Terminate I/O transaction.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`SCSIGetVirtualIDInfo`
: Return `DeviceIdent` of a virtual SCSI ID.

`SCSILoadDriver`
: Load a driver from a SCSI device.

`SCSIOldCall`
: SIM support function for original SCSI Manager emulation.

`SCSICreateRefNumXref`
: Register a device driver.

`SCSILookupRefNumXref`
: Find a driver reference number.

`SCSIRemoveRefNumXref`
: Deregister a device driver.

`SCSIRegisterWithNewXPT`
: XPT was replaced; SIM needs to reregister.

kBusTypeSCSI

Used in the `diReserved` field of the `DeviceIdent` structure to identify the type of device described by the structure.

```
enum {
    kBusTypeSCSI = 0,
    kBusTypeATA = 1,
    kBusTypePCMCIA = 2,
    kBusTypeMediaBay = 3
};
```

Constants

`kBusTypeSCSI`

`DeviceIdent` holds information about a SCSI device.

Available in Mac OS X v10.1 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`kBusTypeATA`

`DeviceIdent` holds information about an ATA device.

Available in Mac OS X v10.1 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`kBusTypePCMCIA`

Not recommended.

Available in Mac OS X v10.1 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`kBusTypeMediaBay`

Not recommended.

Available in Mac OS X v10.1 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

SCSI Result Flags

Used in `scsiResultFlags` field of the `SCSI_IO` structure.

```
enum {
    scsiSIMQFrozen = 0x0001,
    scsiAutosenseValid = 0x0002,
    scsiBusNotFree = 0x0004
};
```

Constants

`scsiSIMQFrozen`

The SIM queue for this LUN is frozen because of an error. You must call the `SCSIReleaseQ` function to release the queue and resume processing requests.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiAutosenseValid`

An automatic `REQUEST SENSE` was performed after this I/O because of a `CHECK CONDITION` status message from the device. The data contained in the `scsiSensePtr` buffer is valid.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiBusNotFree`

The SCSI Manager was unable to clear the bus after an error. You may need to call the `SCSIResetBus` function to restore operation.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

SCSI IO Flags

Used in the `scsiIOFlags` field of the `SCSI_IO` structure.

```
enum {
    scsiNoParityCheck = 0x0002,
    scsiDisableSelectWAtn = 0x0004,
    scsiSavePtrOnDisconnect = 0x0008,
    scsiNoBucketIn = 0x0010,
    scsiNoBucketOut = 0x0020,
    scsiDisableWide = 0x0040,
    scsiInitiateWide = 0x0080,
    scsiRenegotiateSense = 0x0100,
    scsiDisableDiscipline = 0x0200,
    scsiIOFlagReserved0080 = 0x0080,
    scsiIOFlagReserved8000 = 0x8000
};
```

Constants

`scsiNoParityCheck`

Disable parity error detection for this transaction.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiDisableSelectWAtn`

Do not send the IDENTIFY message for LUN selection. The LUN is still required in the `scsiDevice` field so that the request can be placed in the proper queue. The LUN field in the CDB is untouched. The purpose is to provide compatibility with older devices that do not support this aspect of the SCSI-2 specification.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiSavePtrOnDisconnect`
: Perform a SAVE DATA POINTER operation automatically in response to a DISCONNECT message from the target. The purpose of this flag is to provide compatibility with devices that do not properly implement this aspect of the SCSI-2 specification.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiNoBucketIn`
: Prohibit bit-bucketing during the data-in phase of the transaction. Bit-bucketing is the practice of throwing away excess data bytes when a target tries to supply more data than the initiator expects. For example, if the CDB requests more data than you specified in the `scsiDataLength` field, the SCSI Manager normally throws away the excess and returns the `scsiDataRunError` result code. If this flag is set, the SCSI Manager refuses any extra data, terminates the I/O request, and leaves the bus in the data-in phase. You must reset the bus to restore operation. This flag is intended only for debugging purposes.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiNoBucketOut`
: Prohibit bit-bucketing during the data-out phase of the transaction. If a target requests more data than you specified in the `scsiDataLength` field, the SCSI Manager normally sends an arbitrary number of meaningless bytes (0xEE) until the target releases the bus. If this flag is set, the SCSI Manager terminates the I/O request when the last byte is sent and leaves the bus in the data-out phase. You must reset the bus to restore operation. This flag is intended only for debugging purposes.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiDisableWide`
: Disable wide data transfer negotiation for this transaction if it had been previously enabled. This option may not be supported by all SIMs.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiInitiateWide`
: Attempt wide data transfer negotiation for this transaction if it is not already enabled. This option may not be supported by all SIMs.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiRenegotiateSense`

Attempt to renegotiate synchronous or wide transfers before issuing a REQUEST SENSE. This is necessary when the error was caused by problems operating in synchronous or wide transfer mode. It is optional because some devices flush sense data after performing negotiation.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

SCSI_IO Data Types

Used in the `scsiDataType` field of the `SCSI_IO` parameter block.

```
enum {
    scsiDataBuffer = 0,
    scsiDataTIB = 1,
    scsiDataSG = 2,
    scsiDataIOTable = 3
};
```

Constants

`scsiDataBuffer`

The `scsiDataPtr` field contains a pointer to a contiguous data buffer, and the `scsiDataLength` field contains the length of the buffer, in bytes.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiDataTIB`

The `scsiDataPtr` field contains a pointer to a transfer instruction block. This is used by the XPT during original SCSI Manager emulation, when communicating with a SIM that supports this.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiDataSG`

The `scsiDataPtr` field contains a pointer to a scatter/gather list. The `scsiDataLength` field contains the total number of bytes to be transferred, and the `scsiSGListCount` field contains the number of elements in the scatter/gather list.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

SCSIBusInquiryPB Data Types

Used in the `scsiDataTypes` field of the `SCSIBusInquiryPB` structure.

```
enum {
    scsiBusDataTIB = (1 << scsiDataTIB),
    scsiBusDataBuffer = (1 << scsiDataBuffer),
    scsiBusDataSG = (1 << scsiDataSG),
    scsiBusDataIOTable = (1 << scsiDataIOTable),
    scsiBusDataReserved = 0x80000000
};
```

Discussion

These types correspond to the `scsiDataType` field of the SCSI I/O parameter block.

SCSI Transfer Types

Used in the `scsiTransferType` field of the `SCSI_IO` structure.

```
enum {
    scsiTransferBlind = 0,
    scsiTransferPolled = 1
};
```

Constants

`scsiTransferBlind`

Use DMA, if available; otherwise, perform a blind transfer using the handshaking information contained in the `scsiHandshake` field.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiTransferPolled`

Use polled transfer mode. The `scsiHandshake` field is not required for this mode.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

SCSIBusInquiryPB Feature Flags

Used in the `featureFlags` field of the `SCSIBusInquiryPB` structure.

```
enum {
    scsiBusLVD = 0x00000400,
    scsiBusUltra3SCSI = 0x00000200,
    scsiBusUltra2SCSI = 0x00000100,
    scsiBusInternalExternalMask = 0x000000C0,
    scsiBusInternalExternalUnknown = 0x00000000,
    scsiBusInternalExternal = 0x000000C0,
    scsiBusInternal = 0x00000080,
    scsiBusExternal = 0x00000040,
    scsiBusCacheCoherentDMA = 0x00000020,
    scsiBusOldCallCapable = 0x00000010,
    scsiBusUltraSCSI = 0x00000008,
    scsiBusDifferential = 0x00000004,
    scsiBusFastSCSI = 0x00000002,
    scsiBusDMAavailable = 0x00000001
};
```

Constants

`scsiBusInternalExternalUnknown`

The internal/external state of the bus is unknown.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiBusInternalExternal`

The bus is both internal and external.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiBusInternal`

The bus is at least partly internal to the computer.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiBusExternal`

The bus extends outside of the computer.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiBusCacheCoherentDMA`

DMA is cache coherent.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiBusOldCallCapable`
: The SIM supports the original SCSI Manager interface.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiBusDifferential`
: The bus uses a differential SCSI interface.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiBusFastSCSI`
: The bus supports SCSI-2 fast data transfers.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiBusDMAavailables`
: DMA is available.

scsiBusMDP

Used in the `scsiHBAInquiry` field of the `SCSIBusInquiryPB` parameter block.

```
enum {
    scsiBusMDP = 0x80,
    scsiBusWide32 = 0x40,
    scsiBusWide16 = 0x20,
    scsiBusSDTR = 0x10,
    scsiBusLinkedCDB = 0x08,
    scsiBusTagQ = 0x02,
    scsiBusSoftReset = 0x01
};
```

Constants

`scsiBusMDP`
: Supports the `MODIFY DATA POINTER` message.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiBusWide32`
: Supports 32-bit wide transfers.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiBusWide16`
: Supports 16-bit wide transfers.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiBusSDTR`
: Supports synchronous transfers.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiBusLinkedCDB`
: Supports linked commands.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiBusTagQ`
: Supports tagged queuing.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

`scsiBusSoftReset`
: Supports soft reset.

  Available in Mac OS X v10.0 and later.

  Not available to 64-bit applications.

  Declared in `SCSI.h`.

scsiOddDisconnectUnsafeRead1

Used in the `scsiWeirdStuff` field of the `SCSIBusInquiryPB` parameter block.

```
enum {
    scsiOddDisconnectUnsafeRead1 = 0x0001,
    scsiOddDisconnectUnsafeWrite1 = 0x0002,
    scsiBusErrorsUnsafe = 0x0004,
    scsiRequiresHandshake = 0x0008,
    scsiTargetDrivenSDTRSafe = 0x0010,
    scsiOddCountForPhysicalUnsafe = 0x0020,
    scsiAbortCmdFixed = 0x0040,
    scsiMeshACKTimingFixed = 0x0080
};
```

Constants

`scsiOddDisconnectUnsafeRead1`

Indicates that a disconnect or other phase change on a odd byte boundary during a read operation will result in inaccurate residual counts or data loss. If your device can disconnect on odd bytes, use polled transfers instead of blind.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiOddDisconnectUnsafeWrite1`

Indicates that a disconnect or other phase change on a odd byte boundary during a write operation will result in inaccurate residual counts or data loss. If your device can disconnect on odd bytes, use polled transfers instead of blind.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiBusErrorsUnsafe`

Indicates that a delay of more than 16 microseconds or a phase change during a blind transfer on a non-handshaked boundary may cause a system crash. If you cannot predict where delays or disconnects will occur, use polled transfers.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiRequiresHandshake`

Indicates that a delay of more than 16 microseconds or a phase change during a blind transfer on a non-handshaked boundary may result in inaccurate residual counts or data loss. If you cannot predict where delays or disconnects will occur, use polled transfers.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiTargetDrivenSDTRSafe`

Indicates that the SIM supports target-initiated synchronous data transfer negotiation. If your device supports this feature and this bit is not set, you must set the `scsiDisableSelectWAtn` flag in the `scsiIOFlags` field.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

scsiMotherboardBus

Used in the `scsiHBAslotType` field of the `SCSIBusInquiryPB` parameter block.

```
enum {
    scsiMotherboardBus = 0x00,
    scsiNuBus = 0x01,
    scsiPDSBus = 0x03,
    scsiPCIBus = 0x04,
    scsiPCMCIABus = 0x05,
    scsiFireWireBridgeBus = 0x06,
    scsiUSBBus = 0x07
};
```

Constants

`scsiMotherboardBus`

A built-in SCSI bus.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiNuBus`

A NuBus slot.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiPDSBus`

A processor-direct slot.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiPCIBus`

A SIM on a PCI bus card.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiPCMCIABus`

A SIM on a PCMCIA card.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiFireWireBridgeBus`

A SIM connected through a FireWire bridge.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiUSBBus`

A SIM connected on a USB bus.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

kDataOutPhase

Used in the `scsiCurrentPhase` field of the `SCSI_IO` structure.

```
enum {
    kDataOutPhase = ,
    kDataInPhase = 1,
    kCommandPhase = 2,
    kStatusPhase = 3,
    kPhaseIllegal0 = 4,
    kPhaseIllegal1 = 5,
    kMessageOutPhase = 6,
    kMessageInPhase = 7,
    kBusFreePhase = 8,
    kArbitratePhase = 9,
    kSelectPhase = 10,
    kMessageInPhaseNACK = 11
};
```

scsiErrorBase

```
enum {
    scsiErrorBase = -7936
};
```

scsiExecutionErrors

```
enum {
    scsiExecutionErrors = scsiErrorBase,
    scsiNotExecutedErrors = scsiTooManyBuses,
    scsiParameterErrors = scsiPBLengthError
};
```

scsiVERSION

```
enum {
    scsiVERSION = 43
};
```

vendorUnique

```
enum {
    vendorUnique = 0xC0
};
```

scsiDeviceSensitive

Used in the `scsiDriverFlags` field of the `SCSIDriverPB` parameter block.

```
enum {
    scsiDeviceSensitive = 0x0001,
    scsiDeviceNoOldCallAccess = 0x0002
};
```

Constants

`scsiDeviceSensitive`

Only the device driver should access this device. SCSI utilities and other applications that bypass drivers should check this flag before accessing a device.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

`scsiDeviceNoOldCallAccess`

This driver or device does not accept original SCSI Manager requests.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `SCSI.h`.

Result Codes

The table below shows the result codes most commonly returned by the SCSI Manager.

Result Code	Value	Description
`scsiRequestInProgress`	1	Parameter block request is in progress Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scCommErr`	2	Communications error, operation timeout.
`scArbNBErr`	3	Bus busy, arbitration timeout.
`scBadParmsErr`	4	Unrecognized TIB instruction.
`scPhaseErr`	5	Phase error on the SCSI bus.
`scCompareErr`	6	Comparison error from `scComp` instruction.
`scMgrBusyErr`	7	SCSI Manager busy.
`scSequenceErr`	8	Attempted operation is out of sequence.
`scBusTOErr`	9	Bus timeout during blind transfer.
`scComplPhaseErr`	10	SCSI bus was not in status phase on entry to `SCSIComplete`.

Result Code	Value	Description
`scsiPluginInternalError`	-7848	Internal consistency check failed Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiFamilyInternalError`	-7849	Internal consistency check failed Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiCannotLoadPlugin`	-7849	No matching service category Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiBadConnType`	-7850	Bad connection type Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiTargetReserved`	-7853	Target already reserved Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiIOInProgress`	-7854	Can't close connection, I/O in progress Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiBadConnID`	-7856	Bad connection ID Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiInvalidMsgType`	-7858	Invalid message type Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiPartialPrepared`	-7859	Could not do full prepare mem for I/O Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiBadDataLength`	-7860	A zero data length in the parameter block. Available in Mac OS X v10.0 and later. Not available to 64-bit applications.

Result Code	Value	Description
`scsiCDBLengthInvalid`	-7863	The CDB length supplied is not supported by this SIM; typically this means it was too big Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiTransferTypeInvalid`	-7864	The `scsiTransferType` requested is not supported by this SIM Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiDataTypeInvalid`	-7865	SIM does not support the requested `scsiDataType` Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiIDInvalid`	-7866	The initiator ID is invalid Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiLUNInvalid`	-7867	The logical unit number is invalid Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiTIDInvalid`	-7868	The target ID is invalid Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiBusInvalid`	-7869	The bus ID is invalid Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiRequestInvalid`	-7870	The parameter block request is invalid Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiFunctionNotAvailable`	-7871	The requested function is not supported by this SIM Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiPBLengthError`	-7872	The parameter block length supplied was too small for this SIM Available in Mac OS X v10.0 and later. Not available to 64-bit applications.

Result Code	Value	Description
`scsiQLinkInvalid`	-7881	The `qLink` field was not 0 Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiNoSuchXref`	-7882	No driver has been cross-referenced with this device Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiDeviceConflict`	-7883	Attempt to register more than one driver to a device Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiNoHBA`	-7884	No HBA detected Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiDeviceNotThere`	-7885	SCSI device not installed or available Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiProvideFail`	-7886	Unable to provide the requested service Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiBusy`	-7887	SCSI subsystem is busy Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiTooManyBuses`	-7888	SIM registration failed because the XPT registry is full Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiCDBReceived`	-7910	The SCSI CDB was received Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiNoNexus`	-7911	Nexus is not established Available in Mac OS X v10.0 and later. Not available to 64-bit applications.

Result Code	Value	Description
`scsiTerminated`	-7912	Parameter block request terminated by the host Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiBDRsent`	-7913	A SCSI bus device reset (BDR) message was sent to the target Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiWrongDirection`	-7915	Data phase was in an unexpected direction Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiSequenceFailed`	-7916	Target bus phase sequence failure Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiUnexpectedBusFree`	-7917	Unexpected bus free phase Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiDataRunError`	-7918	Data overrun/underrun error Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiAutosenseFailed`	-7920	Automatic `REQUEST SENSE` command failed Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiParityError`	-7921	An uncorrectable parity error occurred Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiSCSIBusReset`	-7922	Execution of this parameter block was halted because of a SCSI bus reset Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiMessageRejectReceived`	-7923	`REJECT` message received Available in Mac OS X v10.0 and later. Not available to 64-bit applications.

Result Code	Value	Description
`scsiIdentifyMessageRejected`	-7924	The target issued a `REJECT` message in response to the `IDENTIFY` message; the LUN probably does not exist Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiCommandTimeout`	-7925	The timeout value for this parameter block was exceeded and the parameter block was aborted Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiSelectTimeout`	-7926	Target selection timeout Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiUnableToTerminate`	-7927	Unable to terminate I/O parameter block request Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiNonZeroStatus`	-7932	The target returned non-zero status upon completion of the request Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiUnableToAbort`	-7933	Unable to abort parameter block request Available in Mac OS X v10.0 and later. Not available to 64-bit applications.
`scsiRequestAborted`	-7934	Parameter block request aborted by the host Available in Mac OS X v10.0 and later. Not available to 64-bit applications.

Deprecated SCSI Manager Reference (Not Recommended) Functions

A function identified as deprecated has been superseded and may become unsupported in the future.

Deprecated in Mac OS X v10.2

DisposeSCSICallbackUPP

Disposes of a UPP to a completion routine. (Deprecated in Mac OS X v10.2. There is no replacement function. For details about communicating with SCSI devices in Mac OS X v10.2 and later, see *SCSI Architecture Model Device Interface Guide*.)

Not recommended

```
void DisposeSCSICallbackUPP (
   SCSICallbackUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.2.

Not available to 64-bit applications.

Declared In
`SCSI.h`

InvokeSCSICallbackUPP

Calls your completion routine. (Deprecated in Mac OS X v10.2. There is no replacement function. For details about communicating with SCSI devices in Mac OS X v10.2 and later, see *SCSI Architecture Model Device Interface Guide*.)

Not recommended

```
void InvokeSCSICallbackUPP (
   void *scsiPB,
   SCSICallbackUPP userUPP
);
```

Discussion
You should not have to call the `InvokeSCSICallbackUPP` function as the system calls your completion routine for you.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.2.

Not available to 64-bit applications.

Declared In
`SCSI.h`

NewSCSICallbackUPP

Creates a new universal procedure pointer (UPP) to a completion routine. (Deprecated in Mac OS X v10.2. There is no replacement function. For details about communicating with SCSI devices in Mac OS X v10.2 and later, see *SCSI Architecture Model Device Interface Guide*.)

Not recommended

```
SCSICallbackUPP NewSCSICallbackUPP (
   SCSICallbackProcPtr userRoutine
);
```

Return Value
See the description of the `SCSICallbackUPP` data type.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.2.

Not available to 64-bit applications.

Declared In
`SCSI.h`

SCSIAction

Initiates a SCSI transaction or request a service from the XPT or SIM. (Deprecated in Mac OS X v10.2. There is no replacement function. For details about communicating with SCSI devices in Mac OS X v10.2 and later, see *SCSI Architecture Model Device Interface Guide*.)

Not recommended

```
OSErr SCSIAction (
   SCSI_PB *parameterBlock
);
```

Parameters

`parameterBlock`
A pointer to a SCSI Manager parameter block.

Return Value
A result code. See "SCSI Manager Result Codes" (page 42).

Discussion
The `SCSIAction` function initiates the request specified by the `scsiFunctionCode` field of the parameter block. Certain types of requests are handled by the XPT, but most are handled by the SIM.

When called asynchronously, `SCSIAction` normally returns the `NoErr` result code, indicating that the request was queued successfully. The result of the SCSI transaction is returned in the `scsiResult` field upon completion. If the `SCSIAction` function returns an error code, the request was not queued and the completion routine will not be called.

When the completion routine is called, it receives the A5 world that existed when the `SCSIAction` request was received. If A5 was invalid when the request was made, it is also invalid in the completion routine.

Your completion routine should use the following function prototype:

```
pascal void (*CallbackProc) (void * scsiPB);
```

There is no implied ordering of asynchronous requests made to different devices. An earlier request may be started later, and a later request may complete earlier. However, a series of requests to the same device is issued to that device in the order received, except when the `scsiSIMQHead` flag is set in the `scsiFlags` field of the parameter block.

When called synchronously, the `SCSIAction` function returns the actual result of the operation. It also places this result in the `scsiResult` field.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.2.
Not available to 64-bit applications.

Declared In
`SCSI.h`

Document Revision History

This table describes the changes to *SCSI Manager Reference*.

Date	Notes
2006-07-12	Made minor formatting changes.
2006-07-24	Deprecated entire document.
2003-01-02	Updated formatting.
	Updated Carbon status of `SCSIAction` function.
	Fixed typographical errors.

Index