

Apple IIGS

Media Control Toolset

ERS

Version 1.0a3

**Copyright 1989,90,91
Apple Computer, Inc.
All Rights Reserved**

**By
Dan Hitchens**

Introduction

The Media Controller Toolset is a collection of routines that provide a consistent interface for controlling multimedia devices.

History

The initial motivation for designing this toolset was to port over to HyperCard IIGS, the two multimedia toolkits (HyperCard Videodisc Toolkit and HyperCard Audio-CD Toolkit) that were designed for the Macintosh. Since they were both written in high-level languages and provided control of multimedia devices for HyperCard, porting them over without deviating from the initial commands and implementation technique (i.e. XCMDs) seemed in order.

Unfortunately, that approach covers only part of the total needs of developers and users. XCMD's only work with HyperCard applications; desk accessories and applications are left having to reinvent the wheel (so to speak), if they want to provide the same capability.

Another problem that occurs is that the Videodisc Toolkit and the Audio-CD Toolkit are totally independent pieces of code with different calling sequences, yet, they provide many capabilities in common. A better approach would be to combine the two toolkits into one.

This commonality of capabilities between the Audio-CD and Videodisc toolkits, suggested that all multimedia devices provide a useful common set of similar controlling features (i.e. play, stop, pause, scan forward, scan backwards, etc.). Using only this small common set, you can effectively command all kinds of media devices (e.g., laserdisc, VCR, camcorder, CD, audio tape recorder, slide projector, digitized sounds, midi sequences, etc.) This commonality of control features provides the basis from which a toolset with an integrated and consistent interface to multimedia devices could be designed.

The primary goal for this toolset, is to provide this standard consistent interface for multimedia devices. This consistent interface (a common toolset and associated drivers for each device) would not only allow developers to produce HyperCard (HyperCard IIGS) stacks with multimedia capability, but also application programs and desk accessories.

Another goal of this toolset is to provide HyperCard IIGS with the same capability and commands of the Laserdisc and Audio-CD toolkits that were designed for the Macintosh. The toolset should be designed so that the HyperCard IIGS hypertalk XCMD's can be implemented (with possible glue

CDEV and Media_Channels

The CDEV is a control panel device which allows the user to specify the configuration he wants, then writes the selected configuration to a file named "Media.Setup". The user will pick a device type (i.e. Pioneer 6000A, Sony 1500, Apple CD SC, etc.) and the port over which the device is connected (i.e. modem port, printer port, SCSI, etc.). This connection or pathway (driver through port) shall be called a **media_channel**. The toolset will have the capability to have more than one media_channel open at one time, requiring the user to also choose which media_channel number (*as of this writing, say a maximum of 8 channels: 1,2,3,4,5,6,7,8*) he wants to have all subsequent calls to that device routed through. Once the user has set his configuration (this is very similar to how printer drivers and ports are chosen for the system), the file "Media.Setup" will be updated with the information.

An example of the media_channels in the above diagram would be:

- 1 -Pioneer 4200 driver connected to printer port.
- 2 -Sony 1200 driver connected to modem port.
- 3 -Apple CD SC driver connected to a SCSI port.

Media.Setup file

This file, as described above, contains the current media configuration, as specified by the user by means of the CDEV. The Media.Setup file resides in the Media.Control folder of the Drivers directory of the System directory. If the CDEV or Toolset doesn't find a Media.Setup file there, it will look for one on a local volume to support network booting.

On network booted systems, If a diskless user boots strictly from the file server, CDEV and toolset will look in the servers volume for the Media.Setup file. If the user has a disk drive and has the minimal system disk to boot from, then the CDEV and toolset will look on the minimal system disk volume for the Media.Setup file.

The structure of the Media.Setup file is as follows:

1 WORD	;file signature bytes
33 BYTE Space	;Device connected to Media Channel 1
33 BYTE Space	;Device connected to Media Channel 2
33 BYTE Space	;Device connected to Media Channel 3
33 BYTE Space	;Device connected to Media Channel 4
33 BYTE Space	;Device connected to Media Channel 5
33 BYTE Space	;Device connected to Media Channel 6
33 BYTE Space	;Device connected to Media Channel 7
33 BYTE Space	;Device connected to Media Channel 8

33 BYTE Space	;Port connected to Media Channel 1
33 BYTE Space	;Port connected to Media Channel 2
33 BYTE Space	;Port connected to Media Channel 3
33 BYTE Space	;Port connected to Media Channel 4
33 BYTE Space	;Port connected to Media Channel 5
33 BYTE Space	;Port connected to Media Channel 6
33 BYTE Space	;Port connected to Media Channel 7
33 BYTE Space	;Port connected to Media Channel 8

NOTE: The device and port connections are P-strings. If a connection has been setup by the CDEV, then the 33 byte space will hold the device or ports ASCII P-string name. If no connection was made then a null P-string (zero in the first byte) will reside in the 33 byte reserved space.

Toolset

The toolset is the consistent interface. Programs will make calls through the toolset to control the media devices. The toolset actually handles few of the commands, most calls will be handed off to the selected device driver where the real work is performed. The application (or hypercard) calls the toolset, which in turn will make appropriate calls to the device driver, which in turn will make appropriate GS/OS calls to the device. This technique is analagous to the way printing is handled (application programs call the print manager, which in turn hands off commands to the printer driver, which in turn sends commands through the port driver to the actual device.) The toolset provides the consistent interface to the user while each of the device drivers worry about the specific characteristics of the device and how to perform the toolset commands.

Device Drivers

The device drivers as described above is where most of the actual work is performed. The toolset will be called and the device driver will be dispatched to perform the task. Each device driver is tailored specifically for the device that it is to control. The device driver knows and understands the particular command sequences that control the device and communicates them through the port driver.

We are planning to intially write the following device drivers:

-Pioneer 4200	(through Modem or Printer port)
-Apple CDSC	(SCSI device)
-Pioneer 2000	(through game port)

Port Drivers

Port drivers are GS/OS device drivers and calls to the device are made through standard GS/OS calls.

File Locations

The media control device drivers, Media.Setup file, and the CD-Remote resource files are located on the bootup volume at `**/SYSTEM/DRIVERS/MEDIA.CONTROL/`.

CD-Remote Resource File Format

The CD-Remote resource files are a database that is designed for compatibility with the CD-Remote desk accessory data base on the Macintosh. The information is stored differently on the GS, however, the data content is the same. Calls in the toolset are provided to retrieve and store information and can be accessed by GS desk accessories, applications, and Hypercard GS XCMD's.

The following is a sample description of the CD-Remote Resource File:

```
resource rPstring (1) {"1.0d1,Media Ctrl,1990"};
resource rPstring (2) {"Fleetwood Mac: Tango In The Night"};
resource rPstring (101) {"Big Love"};
resource rPstring (102) {"Seven Wonders"};
resource rPstring (103) {"Everywhere"};
resource rPstring (104) {"Caroline"};
resource rPstring (105) {"Tango In The Night"};
resource rPstring (106) {"Mystefied"};
resource rPstring (107) {"Little Lies"};
resource rPstring (108) {"Family Man"};
resource rPstring (109) {"Welcome To The Room...Sara"};
resource rPstring (110) {"Isn't It Midnight"};
resource rPstring (111) {"When I See You Again"};
resource rPstring (112) {"You and I, Part II"};

resource rCstring (1)
{"01,01,01,02,01,03,01,04,01,05,01,06,01,07,01,08,01,09,01,10,01,11,01,
12"};
```

LEGEND:

- rPstring (1) is a comma separated P-string with the following

entries:

- Version number
- Media Control signature "Media Ctrl"
- Year
- rPstring (2) is the disc's title.
- rPstring (100+track no.) is the track title (i.e. rPstring (105) is track title string for track 5.)
- rCString (1) is the disc's program string.

CD-Remote File Names

The CD-Remote file names are generated from the Unique-ID (32-bit long value) passed to the routine (MCSetDiscTitle, MCGetDiscTitle, MCSetTrackTitle, MCGetTrackTitle, MCSetProgram, MCGetProgram) as follows:

Unique-ID = \$xYYYYYYY

X (bits 31-28)= Disk type identifier bits

YYYYYYY (bits 27-0)=Unique number

The disk type identifier generates the prefix to the file name and is translated as follows:

\$0 = "CD" for Compact Discs

\$1 = "LD" for Laser Disc

\$2 = "VD" for Video Device (VCR, Camcorder, etc.)

\$3-\$f = "M3" - "MF" for Media Device (generic, applications can use as they wish.)

Example:

Unique-ID = \$00123456 would generate file: "CD.0123456"

Unique-ID = \$29876543 would generate file: "VD.9876543"

Unique-ID = \$789abcdef would generate file: "M7.89ABCDEF"

Toolset Dependencies

The media control toolset requires the following tools to be loaded and started:

- \$01 Tool Locator
- \$02 Memory Manager
- \$03 Miscellaneous Toolset
- \$0B Integer Math Toolset
- \$1E Resource Manager

Typical Representative Applications of Toolset

A True MultiMedia Controller NDA

The toolset's consistent interface to multimedia devices provides ideal support for a generic multimedia controller NDA. An NDA has been designed (see Media Controller NDA ERS) which has a simple user interface incorporating a basic set of buttons (play, stop, scan, step, etc.) and the ability to switch between media channels. Possible enhancements to the NDA might incorporate the VideoMix NDA and/or CD-Remote NDA to provide a truly multimedia controller.

MultiMedia Sequence Editor, Scheduler

The capabilities that the media controller toolset provides, and the consistent interface for controlling multimedia devices, allows for designing a multimedia sequencing editor. The multimedia sequencing editor would finally allow the user the capability to control a whole host of media devices from one application. This multimedia sequencing editor could be icon based and would allow the user to manipulate sequences of media on a timeline.

Toolset Housekeeping Routines

\$0126 MCBotInit

Initializes the media controller toolkit; called only by the Tool Locator.

WARNING:

An application must never make this call.

_MCBootInit

\$0226 MCStartUp

Starts up the media controller toolkit for use by an application.

IMPORTANT:

An application must make this call before it makes any other media controller toolkit calls.

pushword *userID* ;passed user ID

_MCStartUp

Possible Errors

\$2610WasStarted The tool was already started.

\$00xx GS/OS Errors are returned unaltered .

\$02xx Memory Manager errors are returned unaltered.

\$03xx Miscellaneous Toolset errors are returned unaltered.

\$0Bxx Integer Math Toolset errors are returned unaltered.

_MCReset

\$0626 MCStatus

Indicates whether the media controller toolkit is active.

pushword *returned* ;space for returned value
_MCStatus

returned

This routine returns a boolean; TRUE if active (\$0001), FALSE if inactive (\$0000).

Driver Specific Routines

\$0A26 MCLoadDriver

Loads the driver into memory. This call is typically only called by the media control tool and is not usually called by an application (it is only specified here for completeness.)

IMPORTANT:
Applications normally don't make this call

pushword *mcChannelNo* ;passed channel number
_MCLoadDriver

Possible Errors

\$2611BadChannel An invalid media channel no. was specified.

\$00xx GS/OS Errors are returned unaltered .

\$02xx Memory Manager errors are returned unaltered.

\$03xx Miscellaneous Toolset errors are returned unaltered.

\$0Bxx Integer Math Toolset errors are returned unaltered.

\$0B26 MCUnLoadDriver

UnLoads a driver from memory. This call is typically only called by the media control tool but can be called by an application to cause the currently loaded driver to unload (this call would normally be called if the application was directly modifying the Media.Setup file and wanted to force the currently loaded driver to unload so that the next media control tool call will load and startup the newly specified driver.)

IMPORTANT:
Applications normally don't make this call

pushword *mcChannelNo* ;passed channel number
_MCUnLoadDriver

NOTE:

Call MCDSHutDown before making this call to ensure the driver is shutdown.

Possible Errors

\$2605NotLoaded No Driver is currently loaded
\$2611BadChannel An invalid media channel no. was specified.

\$00xx GS/OS Errors are returned unaltered .
\$02xx Memory Manager errors are returned unaltered.
\$03xx Miscellaneous Toolset errors are returned unaltered.
\$0Bxx Integer Math Toolset errors are returned unaltered.

\$1426 MCDStartUp

Starts up the driver. This call is typically only called by the media control tool and is not normally called by an application (it is only specified here for completeness.)

IMPORTANT:

Applications normally don't make this call.

pushword	<i>mcChannelNo</i>	;passed channel number
PushLong	<i>portnameptr</i>	;P-string ptr to connected ports name
_MCDStartUp		

portnameptr

P-string pointer which points to the name of the port that you request the driver to connect to.

Possible Errors

\$2605NotLoaded	No Driver is currently loaded
\$260BInvalidPort	Invalid port specified
\$2611BadChannel	An invalid media channel no. was specified.

\$00xx GS/OS Errors are returned unaltered .
\$02xx Memory Manager errors are returned unaltered.
\$03xx Miscellaneous Toolset errors are returned unaltered.
\$0Bxx Integer Math Toolset errors are returned unaltered.
\$1Exx Resource Manager errors are returned unaltered.

\$1526 MCDShutDown

Shuts down the device driver. This call is typically only called by the media control tool and is not normally called by an application (it is only specified here for completeness.)

IMPORTANT:

Applications normally don't make this call.

pushword	<i>mcChannelNo</i>	;passed channel number
_MCDShutDown		

Possible Errors

\$2605NotLoaded No Driver is currently loaded
\$2611BadChannel An invalid media channel no. was specified.

\$02xx Memory Manager errors are returned unaltered.

Generic Controller Routines

\$0D26 MCBinToTime

Converts a binary value to its equivalent BCD time value.

```
pushlong  result           ;space for result
pushlong  mcBinVal        ;binary value to convert
_MCBinToTime
```

mcBinVal

Binary value to convert

result

Times are returned as BCD digits as follows:

Bits 31-24	BCD Hours (\$00 to \$99)
Bits 23-16	BCD Minutes (\$00 to \$59)
Bits 15-8	BCD Seconds (\$00 to \$59)
Bits 7-0	BCD partial seconds (\$00 to \$74)

Example:

MCBinToTime(\$12D687) would return the BCD value of \$01234567.

Possible Errors

\$0Bxx Integer Math Toolset errors are returned unabltered.

\$1B26 MCControl

Issues a control command to the media device.

```
pushword mcChannelNo ;channel number
pushword ctlcommand ;control command
_MCControl
```

<u><i>ctlcommand</i></u>	<u>Action</u>
mcCInit	Initilize and reset the device
mcCEject	Spin down and eject the disk
mcCVideoOn	Turn video on
mcCVideoOff	Turn video off
mcCDisplayOn	Turn video position display on
mcCDisplayOff	Turn video position display off
mcCBlankVideo	Blank video for next MCSearchTo command
mcCDefaultCom	Set communication as per control panel settings
mcC8Data1Stop	Set 8 data and 1 stop bits
mcC7Data1Stop	Set 7 data and 1 stop bits
mcC6Data1Stop	Set 6 data and 1 stop bits
mcC5Data1Stop	Set 5 data and 1 stop bits
mcC8Data2Stop	Set 8 data and 2 stop bits
mcC7Data2Stop	Set 7 data and 2 stop bits
mcC6Data2Stop	Set 6 data and 2 stop bits
mcC5Data2Stop	Set 5 data and 2 stop bits
mcCBaudDflt	Set baud rate as per control panel setting
mcCBaud50	Set the baud rate to 50
mcCBaud75	Set the baud rate to 75
mcCBaud110	Set the baud rate to 110
mcCBaud134	Set the baud rate to 134.5
mcCBaud150	Set the baud rate to 150
mcCBaud300	Set the baud rate to 300
mcCBaud600	Set the baud rate to 600
mcCBaud1200	Set the baud rate to 1200
mcCBaud1800	Set the baud rate to 1800
mcCBaud2400	Set the baud rate to 2400
mcCBaud3600	Set the baud rate to 3600
mcCBaud4800	Set the baud rate to 4800
mcCBaud7200	Set the baud rate to 7200
mcCBaud9600	Set the baud rate to 9600
mcCBaud19200	Set the baud rate to 19200

mcCModem	Communicate with the device through the modem port
mcCPrinter	Communicate with the device through the printer port
mcCLockDev	Lock device from user physically manipulating it.
mcCUnLockDev	Unlock device from user physically manipulating it.
mcCIgnoreDS	Set so as to ignore disk switched errors (don't report them.)
mcCReportDS	Set to report disk switched errors.

Possible Errors

\$2605NotLoaded	No Driver is currently loaded
\$2607DevRtnError	Device returned error (unable to perform)
\$2608UnRecStatus	Unrecognizable status from device
\$2609BadSelector	Invalid selector value specified
\$260A	FunnyData funny data receive (retry again)
\$260BInvalidPort	Invalid port specified
\$2611BadChannel	An invalid media channel no. was specified.

\$00xx GS/OS Errors are returned unaltered .
 \$02xx Memory Manager errors are returned unaltered.
 \$03xx Miscellaneous Toolset errors are returned unaltered.
 \$0Bxx Integer Math Toolset errors are returned unaltered.

\$2826 MCGetDiscID

This routine returns a unique ID for the currently running disc.

```
pushlong  result           ;space for result
pushword  mcChannelNo      ;passed channel number
_MCGetDiscID
```

The returned value is a unique number for the given disc. For a CD, this number is the total number of blocks (or disc's serial number if available). It's very unlikely that two arbitrary CDs are the same length down to the 75th of a second.

NOTE:

The unique ID returned is a BCD value as follows:

Bits 31-27	BCD zero (\$0)
Bits 27-24	BCD Hours (\$0 to \$9)
Bits 23-16	BCD Minutes (\$00 to \$59)
Bits 15-8	BCD Seconds (\$00 to \$59)
Bits 7-0	BCD partial seconds (\$00 to \$74)

This unique ID is normally used for such routines as: MCGetDiscTitle, MCSetDiscTitle, MCGetTrackTitle, MCSetTrackTitle, MCGetProgram, and MCSetProgram.

Cautionary Note:

On some devices, this call may take some time to complete due to the necessity of the device to search to different locations on the disc to calculate the unique ID.

\$1226 MCGetDiscTitle

Returns the title of the disc by reading the *CD Remote file* database.

pushlong	<i>mcDiscID</i>	;passed disc ID
pushlong	<i>PStrPtr</i>	;passed resultant P-string pointer
<i>_MCGetDiscTitle</i>		

mcDiscID

Unique disc or media Identifier

PStrPtr

Passed pointer to buffer where you want the resultant P-String to be stored.

Note:

The file not found error is returned if no entries have been entered for the particular disc ID (this can be used to determine if any previous entries have been made.)

\$2726 MCGetDiscTOC

This routine returns track information that can be used to generate a table of contents. You pass *mcTrackNo* the track number you want and this routine returns its starting absolute time address (BCD HH:MM:SS:FF).

```
pushlong  result           ;space for result
pushword  mcChannelNo      ;passed channel number
pushword  mcTrackNo        ;passed track number
_MCGetDiscTOC
```

result

Resultant start of track time value.

Times are specified in BCD digits as follows:

Bits 31-24	BCD Hours (\$00 to \$99)
Bits 23-16	BCD Minutes (\$00 to \$59)
Bits 15-8	BCD Seconds (\$00 to \$59)
Bits 7-0	BCD partial seconds (\$00 to \$74)

mcTrackNo

The track number that you want its starting time value.

\$0926 MCGetErrorMsg

This routine returns a P-string text message describing the media control toolset error number passed.

```
pushword  mcErrorNo           ;passed media control toolset error
pushlong  PStrPtr             ;passed P-string pointer to buffer
_MCGetErrorMsg
```

mcErrorNo

This is the media control toolset error number that you want a text description of.

PStrPtr

This is a P-string pointer that points to the buffer area you want the resultant string to be copied to (you should reserve room for a maximum P-string size of 256 bytes).

Note:

If no P-string text message exists for the error number passed, this routine will return the passed error number as an error code indicating that no message was found.

\$1626 MCGetFeatures

Returns the features of a device (Because of the need for possible future expansion, we elected to use enumerated values for selecting the particular features information using *mcFeatSel* instead of just returning a bit-field.)

pushlong	<i>returned</i>	;space for result status
pushword	<i>mcChannelNo</i>	;passed channel number
pushword	<i>mcFeatSel</i>	;passed feature selector value
_MCGetFeatures		

<i>mcFeatSel</i>	<i>returned</i>
------------------	-----------------

mcFTypes

0=	Does nothing
1=	Does InChapters
2=	Does InFrames
3=	Does InFrames & InChapters
4=	Does InTimes
5=	Does InTimes & InChapters
6=	Does InTimes & InFrames
7=	Does InTimes, InChapters, & InFrames

mcFStep Maximum fps speed value (normal 255)
(0=step not supported by device)

mcFRecord Device supports *McRecord* (0=No, 1=yes)

mcFVideo Device supports toggling video (0=No,
1=yes)

mcFEject Device supports ejecting medium (0=No,
1=yes)

mcFLock Device supports user lock (locking user
from physically operating the device,
i.e. eject disk) (0=No, 1=Yes)

mcFVDisplay Device supports video display of
location (0=No, 1=Yes)

mcFVOverlay	No. of lines of overlay characters device supports (0= doesn't support overlay)
mcFVOChars	No. of characters per line supported by overlay (0= doesn't support overlay)
mcFVolume	Device supports volume control (0=No, 1=Yes)

\$2D26 MCGetName

Returns the name and version of the device driver connected to the media channel specified.

```
pushword  mcChannelNo      ;channel number
pushlong  PStrPtr          ;pointer to returned P-string buffer
_MCGetName
```

PStrPtr

This is a passed pointer to the area you want the resultant P-string to be stored.

The value returned is a P-string which is made up as follows:

- The ASCII characters "MCToolkit" followed by
- The device drivers short name followed by its version number (if no device has been connected to this channel, then "NoPlayer" is returned.
- Followed by the connect port name.

EXAMPLES:

```
"MCToolkit Pioneer4200 1.1 MODEM"
"MCToolkit Pioneer2000 1.2d3 GAME PORT"
"MCToolkit NoPlayer"
```

Note:

If an error occurs, the resultant P-string will be undefined (ie. no string will be passed back.)

\$2926 MCGetNoTracks

This routine returns the number of tracks/chapters for the currently running media.

```
pushword result           ;space for result
pushword mcChannelNo      ;passed channel number
_MCGetNoTracks
```

result

The result is a binary value indicating the number of tracks/chapters on the currently installed media.

\$2426 MCGetPosition

Returns the current location at which the device is positioned.

```
pushlong   returned           ;space for results
pushword   mcChannelNo       ;passed channel number
pushword   mcUnitType        ;passed unit type requested for results
_MCGetPosition
```

returned

The current location of the device (as specified by *mcUnitType*.)

mcUnitType

mclnChapters	Specified as Chapters (laserdisc) or tracks (cd) (integer)
mclnFrames	Specified as Frames (video) (integer)
mclnTime	Specified as Time (hours,minutes, seconds, blocks) (BCD)

\$1026 MCGetProgram

This routine returns a comma separated GS/OS-string list. There are two entries for each track/chapter. (You can find the number of tracks/chapters by calling *MCGetTimes*.) The first item is the track/chapter number in the sequence specified in the *CD Remote file*. The second item is 1 if the track/frame should be played, and 0 if the track/frame should not be played. A normal program sequence would look as follows:
01,01,02,01,03,01,04,01...(and so on)

```
pushlong  mcDiscID           ;passed disc ID
pushlong  resultPtr         ;passed GS/OS-string result pointer
_MCGetProgram
```

mcDiscID

Unique disc or media Identifier

resultPtr

Passed GS/OS resultant string pointer.

\$1D26 MCGetSpeeds

Gets a list of available speeds a player can play.

```
pushword  mcChannelNo      ;channel number
pushlong  PStrPtr          ;passed pointer to returned P-string
                                ;buffer.
_MCGetSpeeds
```

PStrPtr

This routine returns an ASCII P-string of the available speeds a player can perform. The speeds are specified in frames per second, and are separated by commas. The calling routine passes a pointer to a P-string buffer area where the result will be stored.

\$1A26 MCGetStatus

Returns the status of a device (Because of the need for possible future expansion, we elected to use enumerated values for selecting the particular status information using *mcStatusSel* instead of just returning a bit-field.)

pushword	<i>returned</i>	;space for result status
pushword	<i>mcChannelNo</i>	;passed channel number
pushword	<i>mcStatusSel</i>	;passed status selector value

_MCGetStatus

mcStatusSel *returned*

mcSDeviceType

mcSLaserDisc	Its a laserdisc player
mcSCDAudio	Its a CD audio player
mcSLaserCD	Its a laserdisc & CD (combination) player.
mcSVCR	Its a VCR
mcSCamCorder	Its a camcorder
mcSVMonitor	Its a video monitor

mcSPlayStatus

mcSPlaying	The device is playing
mcSStill	The device is still
mcSParked	The device is parked or ejected.
mcSUnknown	Unable to determine

mcSDoorStatus

mcSDoorOpen	Door is open
mcSDoorClosed	Door is closed
mcSUnknown	Unable to determine

mcSDiscType

mcS_CLV	A CLV disc is inserted
mcS_CAV	A CAV disc is inserted
mcS_CDV	A CDV disc is inserted
mcS_CD	A CD is inserted
mcSUnknown	Unable to determine

mcSDiscSize

mcSDisc3inch	Playing a 3-inch disc
mcSDisc5inch	Playing a 5-inch disc
mcSDisc8inch	Playing an 8-inch disc

mcSDisc12inch	Playing a 12-inch disc
mcSUnknown	Unable to determine

mcSDiscSide

mcSSideOne	Playing side one
mcSSideTwo	Playing side two
mcSUnknown	Unable to determine

\$2626 MCGetTimes

This routine returns information about the disc.

```
pushlong  result           ;space for result
pushword  mcChannelNo      ;passed channel number
pushword  mcTimesSel       ;passed info selector value
_MCGetTimes
```

mcTimeSel

```
mcElapsedTrack ;Elapsed time on current track/chapter
mcRemainTrack  ;Remaining time on curr. track/chapter
mcElapsedDisc  ;Elapsed time on disc
mcRemainDisc   ;Remaining time on disc
mcTotalDisc    ;Total run time on disc

mcTotalFrames ;Returns total no. of frames on disc
mcTracks      ;Returns binary start and ending track
               numbers (Bits 31-16=ending, Bits
               15-0=starting track no.)
```

NOTE:

Times are returned as BCD digits as follows:

```
Bits 31-24    BCD Hours ($00 to $99)
Bits 23-16    BCD Minutes ($00 to $59)
Bits 15-8     BCD Seconds ($00 to $59)
Bits 7-0      BCD partial seconds ($00 to $74)
```

\$OE26 MCGetTrackTitle

This routine returns the title of a track by reading the database maintained by the *CD Remote file*. The returned string is empty if the disc isn't in the database, or if an error occurred.

pushlong	<i>mcDiscID</i>	;passed disc ID
pushword	<i>mcTrackNo</i>	;passed track number
pushlong	<i>PStrPtr</i>	;passed P-string result pointer
_MCGetTrackTitle		

mcDiscID

Unique disc or media Identifier

mcTrackNo

The track you want the title of

PStrPtr

Passed pointer to where you want the resultant P-String to be stored.

\$2026 MCJog

Advance *mcNJog* units forward, or backwards repeated *mcJogRepeat* times (advance forward if *mcJogRepeat* is positive, backwards if *mcJogRepeat* is negative).

pushword	<i>mcChannelNo</i>	;passed channel number
pushword	<i>mcUnitType</i>	;passed unit type
pushlong	<i>mcNJog</i>	;passed no. of units to jog
pushword	<i>mcJogRepeat</i>	;passed times to repeat
_MCJog		

mcUnitType

mclnChapters	Specified as Chapters (laserdisc) or tracks (cd) (long integer)
mclnFrames	Specified as Frames (video) (long integer)
mclnTime	Specified as Time (hours,minutes, seconds, blocks) (BCD)

mcNJog

Number of units to jog (for mclnFrames, number of frames; for mclnTimes, BCD time value; for mclnChapters, no. of chapters.)

mcJogRepeat

Number of times to repeat the jog sequence (+=forward direction, -=reverse direction.)

\$1826 MCPause

This routine will put the device in pause mode if you are playing. You can resume play by issuing an *MCPlay* command.

```
pushword  mcChannelNo      ;passed channel number  
_MCPause
```

\$1726 MCPlay

Start the device moving forward at normal play speed.

```
pushword mcChannelNo ;passed channel number  
_MCPlay
```

\$2A26 MRecord

This routine puts the device into record mode (if the device has record capability otherwise, it does nothing and returns with an error.)

```
pushword  mcChannelNo      ;passed channel number  
_MRecord
```

\$2526 MCSetAudio

Controls the audio output of the device.

```
pushword mcChannelNo ;passed channel number
pushword mcAudioCtl ;passed audio control value
_MCSetAudio
```

mcAudioCtl

```
AudioOff ;Audio Off
AudioRight ;Audio Right channel only
AudioLinR ;Audio left in right only
AudioMinR ;Audio mixed in right only
AudioRinL ;Audio right in left only
AudioRinLR ;Audio right in left and right
AudioReverse ;Audio right in left, left in right
AudioRinLMR ;Audio right in left, mixed in right
AudioLeft ;Audio left channel only
AudioStereo ;Audio Both Channels (Stereo)
AudioLinLR ;Audio left in left and right
AudioLinLMR ;Audio left in left, mixed in right
AudioMinL ;Audio mixed in left only
AudioMinLRinR ;Audio mixed in left, right in right
AudioMinLLinR ;Audio mixed in left, left in right
AudioMonaural ;Audio mixed in left and right(monaural)
```

\$2E26 MCSetVolume

Sets the left and right volume levels on the device.

```
pushword  mcChannelNo      ;passed channel number
pushword  mcLeftVol       ;passed left volume level
pushword  mcRightVol     ;passed right volume level
_MCSetAudio
```

mcLeftVol

This is the left volume level. It ranges from \$0000 for muted volume to \$ffff for full volume.

mcRightVol

This is the right volume level. It ranges from \$0000 for muted volume to \$ffff for full volume.

\$1C26 MScan

Causes the device to scan forward or reverse. This command is device dependent

```
pushword mcChannelNo ;channel number
pushword mcDirection ;scan direction
_MCScan
```

<u><i>mcDirection</i></u>	<u>Action</u>
positive value	scan forward
negative value	scan backwards

\$2226 MCSearchDone

Returns status indicating whether a previous MCSearchTo command has completed.

```
pushword returned           ;space for result  
pushword mcChannelNo       ;passed channel number  
_MCSearchDone
```

returned

This is a boolean return value. It Returns 1 if search point has been reached, zero if not reached.

NOTE:

This routine will return true only once for each MCSearchTo command.

\$2126 MCSearchTo

Starts a search to the location specified by *searchloc*.

pushword	<i>mcChannelNo</i>	;passed channel number
pushword	<i>mcUnitType</i>	;passed search location unit type
pushlong	<i>searchloc</i>	;passed search location
_MCSearchTo		

mcUnitType

mclnChapters	Specified as Chapters (laserdisc) or tracks (cd) (integer)
mclnFrames	Specified as Frames (video) (integer)
mclnTime	Specified as Time (hours,minutes, seconds, blocks) (BCD)

searchloc

The location you want to search to specified in *mcUnitType* units.

NOTE:

After performing an MCSearchTo command, an MCSearchWait or a series of MCSearchDone commands until returning true must be performed to insure the search has completed before issuing another command.

\$2326 MCSearchWait

Waits until the previous MCSearchTo command has been completed.

```
pushword mcChannelNo ;passed channel number  
_MCSearchWait
```

NOTE:

MCSearchWait should only be called after MCSearchTo, otherwise it will wait until it returns with a timeout error.

\$1926 MCSendRawData

Sends raw data to the device (this command is provided to support direct communication to the device that will allow applications access to features the device may provide that aren't supported by this toolset. Use MCWaitRawData to receive data directly from the device.)

```
pushword  mcChannelNo      ;channel number
pushlong  mcNativePtr     ;passed GS/OS string pointer to data
                                     ;to be sent to the device

_MCSendRawData
```

mcNativePtr

Pointer to GS/OS string data to be sent to the device.

\$1326 MCSetDiscTitle

This routine sets the disc title in the *CD Remote file* to be the title P-string given.

```
pushlong  mcDiscID           ;passed disc ID
pushlong  titleptr           ;passed title P-string ptr
_MCSetDiscTitle
```

mcDiscID

Unique disc or media Identifier.

titleptr

Pointer to a P-string which contains the title you want.

Note:

If there isn't a currently created file in the CD Remote database for the specified unique disc ID, this call will create a file using the passed unique disc ID and set the disc title to the string pointed to by the passed parameter *titleptr*.

\$1126 MCSetProgram

This routine sets the program of the *CD Remote file* (using the passed unique disc ID) which is a comma separated GS/OS-string list. There are two entries for each track/chapter. The first item is the track/chapter number in the sequence specified in the *CD Remote file*. The second item is 1 if the track/frame should be played, and 0 if the track/frame should not be played. A normal play sequence would be 01,01,02,01,03,01,04,01...

```
pushlong    mcDiscID           ;passed unique disc ID
pushlong    mcProgPtr          ;passed GS/OS-string pointer
_MCSetProgram
```

mcDiscID

Unique disc or media Identifier

mcProgPtr

Passed GS/OS pointer to the program string.

Note:

If there isn't a currently created file in the CD Remote database for the specified unique disc ID, this call will create a file using the passed unique disc ID and set the program to the string pointed to by the passed parameter *mcProgPtr*.

\$OF26 MCSetTrackTitle

This routine sets the track title in the *CD Remote file* to be the title P-string given.

```
pushlong    mcDiscID           ;passed disc ID
pushword    TrackNum           ;passed track number
pushlong    TitlePtr           ;passed title P-string pointer
_MCSetTrackTitle
```

mcDiscID

Unique disc or media Identifier

TrackNum

Track number you want set title of.

titleptr

Pointer to a P-string which contains the title that is to be set .

Note:

If there isn't a currently created file in the CD Remote database for the specified unique disc ID, this call will create a file using the passed unique disc ID and set the track title to the string pointed to by the passed parameter *TitlePtr*.

\$1E26 MCSpeed

Place the device at the specified frames per second:

```
pushword mcChannelNo ;passed channel number  
pushword mcFPS ;passed frames per second  
_MCSpeed
```

mcFPS

Frames per second frame rate

NOTE:

This command applies only to the next MCPlay command executed. After each MCPlay command the speed value is reset to normal (30fps).

\$2B26 MCStop

Stops the device. After an *MCStop* command, some devices will resume play at the current location and some will resume at the start of the media.

```
pushword mcChannelNo ;passed channel number  
_MCStop
```

\$1F26 MCStopAt

Set the location at which the device will stop playing (normally set before MCPlay call to insure device will stop correctly.)

pushword	<i>mcChannelNo</i>	;passed channel number
pushword	<i>mcUnitType</i>	;passed unit type
pushlong	<i>mcStopLoc</i>	;passed stop location
_MCStopAt		

mcUnitType

mclnChapters	Specified as Chapters (laserdisc) or tracks (cd) (integer)
mclnFrames	Specified as Frames (video) (integer)
mclnTime	Specified as Time (hours,minutes, seconds, blocks) (BCD)

mcStopLoc

The stop location specified in mcUnitType terms.

\$0C26 MCTimeToBin

Converts a BCD time value from hours, minutes, seconds, and frames to its binary equivalent.

```
pushlong  result           ;space for result
pushlong  mcTimeValue     ;Time value to convert
_MCTimeToBin
```

mcTimeValue

Times are specified in BCD digits as follows:

Bits 31-24	BCD Hours (\$00 to \$99)
Bits 23-16	BCD Minutes (\$00 to \$59)
Bits 15-8	BCD Seconds (\$00 to \$59)
Bits 7-0	BCD partial seconds (\$00 to \$74)

result

The result is a long binary value.

Example:

MCTimeToBin(\$01234567) would return the binary value of 1234567
(or \$12D687.)

\$2C26 MCWaitRawData

Receives raw data from the device until a terminal character has been received or until *tickwait* system ticks have passed.

pushword	<i>mcChannelNo</i>	;passed channel number
pushlong	<i>resultptr</i>	;passed GS/OS string result pointer
pushword	<i>tickwait</i>	;passed no. of ticks before timeout
		;error
pushword	<i>term_mask</i>	;terminal character and mask
	<i>_MCWaitRawData</i>	

result

This is a passed GS/OS resultant pointer.

tickwait

The number of system ticks to occur before the routine terminates with a timeout error. This is used to prevent system hangs waiting for characters from the device that may never occur.

term_mask

Bits 7-0 mask character.

Bits 15-8 terminal character.

The routine determines when to finish by masking the received characters with the passed mask and comparing it against the terminal character. When they are equal the routine will complete and the GS/OS string size will reflect the number of characters transferred including the terminal character.

If the passed mask character is zero and the terminal character is non-zero, then this routine will receive the number of characters specified by the GS/OS string size word before completing (this allows for block transfers without looking for a terminal character).

If mask character and the terminal character are zero then the routine will return one character in the GS/OS string if one is immediately available otherwise it will return and the return size will be zero after tickwait ticks have elapsed (normally for this case, set tickwait to one) (this technique allows for polling for one character at a time.)

EQUATES

```
InChapters          equ 1          ;Selector value for Chapters
InFrames            equ 2          ;Selector value for Frames
InTimes             equ 3          ;Selector value for Times

;-----
;Control values for MCCControl
;-----
mcClnit             equ 1          ;initilize player
mcCEject            equ 2          ;eject disc
mcCVideoOn          equ 3          ;turn video on
mcCVideoOff         equ 4          ;turn video off
mcCDisplayOn        equ 5          ;turn video position display off
mcCDisplayOff       equ 6          ;turn vidoe position display on
mcCBlankVideo       equ 7          ;blank video for next MCSearchTo
mcCDefaultCom       equ 8          ;set default communications
mcCLockDev          equ 9          ;set the device to locked
mcCUnLockDev        equ 10        ;unlock the device

mcC8Data1Stop       equ 40        ;set 8-data 1-stop bit
mcC7Data1Stop       equ 41        ;set 7-data 1-stop bit
mcC6Data1Stop       equ 42        ;set 6-data 1-stop bit
mcC5Data1Stop       equ 43        ;set 5-data 1-stop bit
mcC8Data2Stop       equ 44        ;set 8-data 2-stop bit
mcC7Data2Stop       equ 45        ;set 7-data 2-stop bit
mcC6Data2Stop       equ 46        ;set 6-data 2-stop bit
mcC5Data2Stop       equ 47        ;set 5-data 2-stop bit

mcCBaudDflt         equ 50        ;set baud rate to control panel
                                ;setting
mcCBaud50           equ 51        ;set 50 baud
mcCBaud75           equ 52        ;set 75 baud
mcCBaud110          equ 53        ;set 110 baud
mcCBaud134          equ 54        ;set 134.5 baud
mcCBaud150          equ 55        ;set 150 baud
mcCBaud300          equ 56        ;set 300 baud
mcCBaud600          equ 57        ;set 600 baud
mcCBaud1200         equ 58        ;set 1200 baud
mcCBaud1800         equ 59        ;set 1800 baud
mcCBaud2400         equ 60        ;set 2400 baud
mcCBaud3600         equ 61        ;set 3600 baud
mcCBaud4800         equ 62        ;set 4800 baud
mcCBaud7200         equ 63        ;set 7200 baud
mcCBaud9600         equ 64        ;set 9600 baud
mcCBaud19200        equ 65        ;set 19200 baud

mcCModem            equ 100       ;set to modem port
mcCPrinter          equ 101       ;set to printer port

mcCIgnoreDS         equ 200       ;set to ignore disk switched
                                ;errors (don't report them.)
mcCReportDS         equ 201       ;set to report disk switched errors.
```

```

;-----
;Status values for MCGetFeatures
;-----
mcFTypes          equ 0          ;Does frames, times & chapters
mcFStep           equ 1          ;Max. step value
mcFRecord         equ 2          ;Does MRecord function
mcFVideo          equ 3          ;Does video
mcFEject          equ 4          ;Does eject function
mcFLock           equ 5          ;Does user key lock
mcFVDisplay       equ 6          ;Does video location display
mcFVOverlay       equ 7          ;No. of lines of char. video
                                ;overlay, zero if doesn't support
mcFVOChars        equ 8          ;Video overlay no. of chars. per
                                ;line, zero if none.
mcFVolume         equ 9          ;Does volume control?

;-----
;Status values for MCGetStatus
;-----
mcSUnknown        equ 0          ;player unable to determine this status
mcSDeviceType     equ $0000      ;device type selector
mcSLaserDisc      equ 1
mcSCDAudio        equ 2
mcSCDLaserCD      equ 3
mcSVCR            equ 4
mcSCamCorder      equ 5
mcSPlayStatus     equ $0001      ;play status selector value
mcSPlaying        equ 1
mcSStill           equ 2
mcSParked         equ 3
mcSDoorStatus     equ $0002      ;players door status
mcSDoorOpen       equ 1
mcSDoorClosed     equ 2
mcSDiscType       equ $0003      ;disc type selector value
mcS_CLV           equ 1
mcS_CAV           equ 2
mcS_CDV           equ 3
mcS_CD            equ 4
mcSDiscSize       equ $0004      ;disc size selector value
mcSDisc3inch      equ 3
mcSDisc5inch      equ 5
mcSDisc8inch      equ 8
mcSDisc12inch     equ 12
mcSDiscSide       equ $0005      ;disc side selector value
mcSSideOne        equ 1
mcSSideTwo        equ 2
mcSVolumeL        equ $0006      ;Current Left volume selector
mcSVolumeR        equ $0007      ;Current Right volume selector

;-----
; MCGetTimes selector values
;-----
mcElapsedTrack     equ 0          ;Elapsed time on current track/chapter
mcRemainTrack      equ 1          ;Remaining time on curr. track/chapter
mcElapsedDisc      equ 2          ;Elapsed time on disc
mcRemainDisc       equ 3          ;Remaining time on disc
mcTotalDisc        equ 4          ;Total run time on disc

```

```

mcTotalFrames      equ 5           ;Returns total no. of frames on disc
mcTracks           equ 6           ;Returns the first and last track numbers
mcDiscID           equ 7           ;Returns a disc identifier

```

```

;-----
;Audio values
;-----

```

```

AudioOff           equ 0           ;Audio Off
AudioRight         equ 1           ;Audio Right channel only
AudioLinR          equ 2           ;Audio left in right only
AudioMinR          equ 3           ;Audio mixed in right only
AudioRinL         equ 4           ;Audio right in left only
AudioRinLR        equ 5           ;Audio right in left and right
AudioReverse       equ 6           ;Audio right in left, left in right
AudioRinLMR       equ 7           ;Audio right in left, mixed in right
AudioLeft         equ 8           ;Audio left channel only
AudioStereo       equ 9           ;Audio Both Channels (Stereo)
AudioLinLR        equ 10          ;Audio left in left and right
AudioLinLMR       equ 11          ;Audio left in left, mixed in right
AudioMinL         equ 12          ;Audio mixed in left only
AudioMinLRinR     equ 13          ;Audio mixed in left, right in right
AudioMinLLinR     equ 14          ;Audio mixed in left, left in right
AudioMonaural     equ 15          ;Audio mixed in left and right (monaural)

```

```

*****

```

```

*
* Error Equates
*

```

```

*****

```

```

UnImp              equ $2601       ;Unimplemented for this device
NotApplic          equ UnImp
BadSpeed           equ $2602       ;Invalid speed specified
BadUnitType       equ $2603       ;Invalid unit type specified
TimeOutErr        equ $2604       ;Timed out during device read
NotLoaded         equ $2605       ;No Driver is currently loaded
BadAudio          equ $2606       ;Invalid Audio Value
DevRtnError       equ $2607       ;Device returned error (unable to
                                ;perform)
UnRecStatus       equ $2608       ;Unrecognizable status from device
BadSelector       equ $2609       ;Invalid selector value specified
FunnyData         equ $260a       ;funny data receive (retry again)
InvalidPort       equ $260b       ;invalid port specified
OnlyOnce          equ $260c       ;Scans only once
NoResMgr          equ $260d       ;Resource manager not active (must
                                ;be loaded and started.)
ItemNotThere      equ $260e       ;Item Not found in CD-Remote
                                ;database
WasShutDown       equ $260f       ;The tool was already shutdown.
WasStarted        equ $2610       ;The tool was already started.
BadChannel        equ $2611       ;An invalid media channel no. was
                                ;specified.
InvalidParam      equ $2612       ;An invalid parameter was
                                ;specified.
CallNotSupported  equ $2613       ;An invalid media control tool call
                                ;was attempted.

```


Possible Errors

\$2601Unimp	Unimplemented for this device
\$2602BadSpeed	Invalid speed specified
\$2603BadUnitType	Invalid unit type specified
\$2604TimeOutErr	Timed out during device read
\$2605NotLoaded	No Driver is currently loaded
\$2606BadAudio	Invalid Audio Value
\$2607DevRtnError	Device returned error (unable to perform)
\$2608UnRecStatus	Unrecognizable status from device
\$2609BadSelector	Invalid selector value specified
\$260A	FunnyData funny data receive (retry again)
\$260BInvalidPort	Invalid port specified
\$260COnlyOnce	Scans only once
\$260D	NoResMgr Resource manager not active
\$260E	ItemNotThere Item Not found in CD-Remote database
\$260F	WasShutDown The tool was already shutdown.
\$2610WasStarted	The tool was already started.
\$2611BadChannel	An invalid media channel no. was specified.
\$2612InvalidParam	An invalid parameter was specified.

\$00xx GS/OS Errors are returned unaltered .

\$02xx Memory Manager errors are returned unaltered.

\$03xx Miscellaneous Toolset errors are returned unaltered.

\$0Bxx Integer Math Toolset errors are returned unaltered.

\$1Exx Resource Manager errors are returned unaltered.

```
*****
*
* This is a sample of the typical header code for a media control driver. This sample shows the
* structure (DriverID, Count of calls supported, Entry Jump, Version Number, Drivers Name
* string, routine vector table) and some typical entry and exit code.
*
*****
```

```
*****
*
* Equates
*
```

```
*****
DriverID      equ $3883                ;Media Control Driver ID
VersionNo     equ $0100                ;Version no. 1.00
*****
```

```
*****
*
* Driver Header and Vectors Routine
*
```

```
*****
P4200Drvr    PROC
              longi on
              longa on
              DC.W  DriverID              ;Drivers ID Marker (Media Ctrl driver)
              DC.W  (EndVector-DriverVtrs)/4 ;count of calls supported

              jmp   EntryStuff            ;Go do entry setup and vector to routine
              DC.W  VersionNo             ;Drivers version number (version 1.0)
              str   'SAMPLEDRIVER'       ;Drivers name
*****
```

```
Export DriverVtrs
DriverVtrs
DC.L  MCGetName           ;0 Gets the drivers name and version
DC.L  MCDStartUp         ;1 Starts up the driver
DC.L  MCDShutDown       ;2 Shuts down the driver
DC.L  MCGetFeatures     ;3 Gets features
DC.L  MCPlay            ;4 Play the device
DC.L  MCPause           ;5 Pause device
DC.L  MCSendRawData     ;6 Send raw data to the device
DC.L  MCGetStatus       ;7 Return device status info
DC.L  MCCControl        ;8 Sets device control info
DC.L  MCScan            ;9 Scan forward or reverse
DC.L  MCGetSpeeds      ;10 Returns Valid Speeds
DC.L  MCSpeed           ;11 Sets Speed for MCPlay
DC.L  MCStopAt         ;12 Sets stopping point
DC.L  MCJogN           ;13 Jog device n units
DC.L  MCSearchTo       ;14 Search to location
DC.L  MCSearchDone     ;15 Status of if done
DC.L  MCSearchWait     ;16 Waits for search to finish
DC.L  MCGetPosition    ;17 Returns current position
DC.L  MCSetAudio       ;18 Controls audio
DC.L  MCGetTimes       ;19 Gets disc times
DC.L  MCGetDiscTOC    ;20 Gets discs table of contents
DC.L  MCGetDisclD     ;21 Gets discs ID
DC.L  MCGetNoTracks   ;22 Gets discs no. of tracks
DC.L  MCRecord        ;23 Puts device into record mode
DC.L  MCStop          ;24 Stops the device
DC.L  MCWaitRawData   ;25 Waits for raw data from device
```

```
EndVector
ENDP
```

```

;-----
;
; NAME:      EntryStuff
;
; PURPOSE:   To save registers and setup direct page on stack and vector
;            to called routine.
;
; PASSED:    (X-reg.)= offset into "GPortVtrs" to called routine
;
; RETURNED:  (Vectors directly to called routine, routine returns through
;            correct EXIT routine)
;-----
EntryStuff      PROC
                Definestack

                phd                      ;save direct page
                phb                      ;save bank register
;set bank register (B-reg.) to programs bank (K-register)
                phk
                plb
;reserve 10 bytes from stack for direct page
                tsc                      ;get stack value
                sec
                sbc #DirectStack         ;reserve bytes for d-page
                tcs
                inc a                    ;inc. by one for d-page start
                tcd                      ;set direct page address

                jmp (DriverVtrs,x)       ;vector to called routine
                ENDP

```

```

;-----
;-----
;
; Dynamic Exits (This is a sample exit routine.)
;
;
; Each dynamic exit   takes 2 RTL's and moves them some number of bytes up the
; stack.
;
;-----
;-----

```

DynamicExits PROC EXPORT

```

                EXPORT Exit2
Exit2
                tax
;Restore B-reg.
                lda DirectStack,s        ;get saved B-reg. into bits 7-4
                pha                      ;save onto stack
                plb                      ;get rid of 8-bits
                plb                      ;now restore the b-register
;Restore D-reg.
                lda DirectStack+2,s
                tcd

                lda DS3+5,s
                sta DS3+7,s
                lda DS3+3,s

```

```

        sta DS3+5,s
        lda DS3+1,s
        sta DS3+3,s
        pla
        bra Exit0x

;
;This clears up the stack and restores registers
;
        EXPORT Exit0
Exit0
        tax                                ;save A-reg in X-reg
;Restore B-reg.
        lda DirectStack,s                ;get saved B-reg. into bits 7-4
        pha                                ;save onto stack
        plb                                ;get rid of 8-bits
        plb                                ;now restore the b-register
;Restore D-reg.
        lda DirectStack+2,s
        tcd

Exit0x
;Get rid of direct page from stack
        tsc                                ;get stack pointer
        clc
        adc #DS3                          ;take out direct page area and D & B reg. saves
        tcs

        txa                                ;restore A-reg.
        cmp #1                            ;sets carry if A-reg. not zero
        rtl
        ENDP

```