About the Device Manager Controlling data flow to and from devices

The **Device Manager** handles the flow of information between your application and disk drives, serial communications ports, printers, and slot devices. Drives, ports and printers are general, while slot devices are specific to the Mac II family and are treated separately, below.

The general devices common to all Macintoshes fall into two categories, character devices and block devices. Character devices deal with data one byte at a time while block devices handle groups (or blocks) of bytes together. Character devices deal with information sequentially, one byte at a time, no skipping and no going back. Block devices not only handle data in chunks, they can read or write any available block at any time.

The way the **Device Manager** controls such hardware as drives and printers is through device drivers, programs that convert data into actions and vice versa. The standard Operating System ROM device drivers are the Disk Driver, the Sound Driver and the ROM (as opposed to the RAM) Serial Driver. RAM device drivers include the Printer Driver, RAM Serial Driver, AppleTalk drivers and desk accessories. Additionally, RAM drivers are resources and reside in the system resource file, from which they are read as needed. After a driver is called, or opened, your program can read from it or write to it. When you close a driver you release the memory space it occupied.

You identify a device driver by name before it is opened and by its reference number afterwards. Driver names consist of a period followed by as many as 254 characters.

To communicate with a device driver, your application writes to and reads from an intermediate application storage area, a data buffer. Such communication can consist of applications data, control information directing the activity of the driver, and status information from the driver reporting back to the application. There are both high-level and low-level methods to effect this communication, with the greater degree of control being available at the low level. High-level control, however, is simpler and provides sufficient control for many uses.

Chooser

The Chooser is a desk accessory designed to make communicating with and managing devices simpler and more uniform. It lets users make selections among different options, as they relate to different drivers. For example, the Chooser prompts the user for a choice of printers or serial ports. The Chooser represents devices by device resource files in the startup System Folder and categorizes all devices into three basic types: 'PRES' resource files for serial printers; 'PRER' for parallel printers; and 'RDEV' for everything else.

There are further resources contained within 'PRER' and 'RDEV' resource files with the following identifications:

Name	ID	Description
'PACK'	-4096	Device package
'STR '	-4096	String with name for AppleTalk device
'GNRL'	-4096	AppleTalk device NBP timeout and retry data

'STR '	-4093	String for left button title
'STR '	-4092	String for right button title
'STR '	-4091	String used to label a list when selecting a device
'BNDL'	variable	Icon (see Finder Icons & BNDLs)
'STR '	-4090	Reserved

The device package resource contains information that includes a Device ID number, a version number to distinguish between different releases of the driver code, and a long integer field holding a flag with data as follows:

Bit	Description
31	Set for AppleTalk device
30	Reserved (0)
29	Reserved (0)
28	Set to indicate ability to select multiple instances
27	Set if device package uses the left button
26	Set for right button
25	Set if no zone name is saved
24	Set if zone names are used
23	Reserved (0)
22	Reserved (0)
21	Reserved (0)
20	Reserved (0)
19	Reserved (0)
18	Reserved (0)
17	Reserved (0)
16	Set to indicate acceptance of the newSel message
15	...of fillList message
14	...of getSel message
13	...of select message
12	...of deselect message
11	...of terminate message
10	Reserved (0)
9	Reserved (0)
8	Reserved (0)
7	Reserved (0)
6	Reserved (0)
5	Reserved (0)
4	Reserved (0)
3	Reserved (0)
2	Reserved (0)
1	Reserved (0)
0	Reserved (0)

Aside from specifying buttons and setting their names, a device package can position either or both buttons and can supply a custom list definition for the device list (such as an icon, a picture or a name, and a small icon). Also, if your application does its own housekeeping, you can bypass the warning message that would normally accompany the user's selection each time a new device is chosen.

Slot Devices

Applications that use add-in cards other than standard video cards rely on a specialized "slot device" . The Macintosh will start up from a slot device if the

card's configuration ROM contains startup code in an sResource (these are different from normal Macintosh resources, but conceptually similar). The Mac II ROM loads the slot startup code and calls its entry point. In many cases, as with traditional UNIX operating systems for example, the Mac ROM relinquishes control completely at this point. If a slot device is specified as the startup device but the normal Macintosh operating system is being used, a boot record for the slot device, sBootRecord, is called twice. First, the startup code loads and opens at least one driver for the slot device and installs it in the disk drive queue--which then becomes the initial driver to install the operating system. A second call to the slot-device boot record opens the card's remaining device drivers.

Once the operating system is installed, it, in turn, installs default video and startup drivers, then looks for additional device drivers on the bus to install and then installs whatever 'INIT' resources are specified. Every slot device on a card is described in the card's configuration ROM through a slot resource directory (sRsrcDir). Additionally, each slot device is specified in a slot resource list (sRsrcList) that gives its resource name (sRsrcName) and the offset to a table of drivers.

Slot resources (sResources) have their drivers located during startup via the following procedure:
1) the operating system searches the sResource list for an sRsrc_Flags field;

2) if the sRsrc_Flag's fOpenAtStart bit equals 1 (or if there is no flag field), the operating system looks for a driver. If fOpenAtStart equals 0, the operating system looks for the next resource;

3) assuming the flag bit was set to 1, the operating system next looks for a driver load record in the resource list. Finding it, the system copies the load record from the card's ROM to the system heap and executes it. A pointer in A0 to an seBlock is then passed to the routine and, on exit, the routine returns a handle in the seResult field of the same block to the driver it just loaded. Provided that the seStatus field equals 0, the System installs the driver;

4) without a driver load record, the operating system looks for a driver directory entry in the resource list. If it finds one and if it contains a driver type sMacOS68000 or sMacOS68020, it reads the diver from the card's ROM and installs it in the system heap.

The installation process itself consists of several steps. First the ROM loads the driver into the system heap, locks it if the dNeedsLock bit in the driver flags word is set, installs the driver with a DrvrInstall system call and, finally, initializes it with an Open call. Open calls that result in errors cause the driver to be marked as closed, the ioParameter block's refNum field is cleared and the driver is unlocked.

Going back to the start of the process, since the System file is initially unavailable, the video driver used at the very beginning of the System startup procedure has to be taken from a video card's configuration ROM. In Macs with more than one video card, the one with the lowest slot number is used unless a different card is specified by parameter RAM or unless the user installs an 'INIT' 31 resource that loads a new driver from a file and overrides the initial driver. The unit table data structure starts out with space for 64 devices but automatically expands to 128 if the particular computer's configuration goes beyond the default capacity.

Interrupts

Once the slot device is selected, installed, and opened, you address it through the VIA2 chip. The chip has an 8-bit register that, in turn, has a bit for each slot. There is, therefore, one interrupt line per card. To go further and locate the interrupt to a particular device on a particular card you institute a polling routine. The way the poll works is that the **Device Manager** maintains a slot-by-slot interrupt queue in which queue elements are listed in order of importance along with pointers to polling routines. When it gets a slot interrupt, the **Device Manager** calls each polling routine in the interrupt queue until it gets an acknowledgement that the interrupt was satisfied. An unsuccessful poll results in a System error dialog.

Driver Structure

RAM drivers are stored as 'DRVR' resources--with the resource and the driver having the same name. This duplication of identity is continued in another area: the driver's resource ID and its unit number are the same and must be in the range 0 - 31. A side effect of this duplicate numeration is that if you use the unit number of an existing driver, your new driver will completely replace the existing one.

The way a driver is constructed, it begins with flags to declare:1) if it is open; 2) if it is RAM based and; 3) if it is currently executing. Next come timing, event mask and location data. Finally, the rest of the structure is dedicated to offsets to the routines that perform the driver's job and the routines themselves.

Things that all drivers have in common are routines to handle Open and Close calls. Depending on what the driver is supposed to do, it can have routines for Read, Write, Control, Status and KillIO functions.

A data structure called a **Device Control Entry** (DCE) receives information about the driver the first time it is opened. It holds information on the location of the driver's routines, and is locked when the driver is open and unlocked when the driver is closed. When the driver serves a slot device, the DCE also contains information about the slot number, the sResource ID number and the external device ID number.