Developer Note

Macintosh LC 630 and Macintosh Quadra 630 Computers

Developer Press

Apple Computer, Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Contents

Figures and Tables

About This Note

This developer note describes the Macintosh LC 630 and Macintosh Quadra 630 computers, pointing out their similarities to earlier models and emphasizing the features that are new or different. It is intended to help experienced Macintosh hardware and software developers design compatible products. If you are unfamiliar with Macintosh computers or would simply like more technical information, you may wish to read the related technical manuals listed in the section "Supplemental Reference Documents."

Contents of This Note

The information is arranged in six chapters, an appendix, and an index:

- Chapter 1, "Introduction," gives a summary of the features of the Macintosh LC 630 and Macintosh Quadra 630 computers, describes their appearance, and lists the available configurations and options.
- Chapter 2, "Architecture," describes the internal organization of the computers. It includes a block diagram and descriptions of the main components of the logic board.
- Chapter 3, "I/O Features," describes the built-in I/O devices and the external I/O ports. It also describes the external video monitors that can be used with the Macintosh LC 630 and Macintosh Quadra 630 computers.
- Chapter 4, "Expansion Features," describes the expansion slots of the Macintosh LC 630 and Macintosh Quadra 630 computers. This chapter provides guidelines for designing cards for the I/O expansion slot and brief descriptions of the expansion modules for the other slots.
- Chapter 5, "Software Features," summarizes the new features of the ROM software and the system software that accompany the Macintosh LC 630 and Macintosh Quadra 630 computers.
- Chapter 6, "Software for the IDE Hard Disk," gives the program interface for the system software and the driver that support the internal IDE hard disk drive.
- The appendix includes foldout pages with mechanical drawings for the I/O expansion card described in Chapter 4.

Supplemental Reference Documents

To supplement the information in this developer note, developers should have copies of the appropriate Motorola reference books for the MC68040 microprocessor. Software developers should have a copy of Motorola's *MC68040 Programmer's Reference Manual.* Hardware developers should have copies of Motorola's *MC68030 User's Manual*, *MC68040 User's Manual*, and *MC68040 Designer's Handbook.*

For additional information about the digital data format used in the video input module, refer to *Macintosh DAV Interface for NuBus Expansion Cards,* part of *Macintosh Developer Note Number 8,* APDA catalog number R0566LL/A. For information about the digital video interface, refer to the *SAA7194/6 Philips Desktop Video Handbook*.

Developers may also need copies of the appropriate Apple reference books. You should have the relevant books of the *Inside Macintos*h series, and particularly *Inside Macintosh: QuickTime Components.* You should also have *Guide to the Macintosh Family Hardware,* second edition, and *Designing Cards and Drivers for the Macintosh Family,* third edition. These books are available in technical bookstores and through APDA.

Information About Earlier Models

Many features of the Macintosh LC 630 and Macintosh Quadra 630 computers are similar to those of certain earlier Macintosh models, so you may wish to have the developer notes that describe those earlier machines:

- *Macintosh Developer Note Number 3,* APDA catalog number R0461LL/A
- *Macintosh Developer Note Number 4,* APDA catalog number R0528LL/A
- *Macintosh Developer Note Number 6,* APDA catalog number R0550LL/A

Macintosh Developer Note Number 3 includes information about the Macintosh LC III and the Macintosh Centris 610 and 650 computers. *Macintosh Developer Note Number 4* includes information about the Macintosh LC 520 computer. *Macintosh Developer Note Number 6* includes information about the Macintosh LC 475 and Macintosh Quadra 605 computers.

The numbered developer notes are available from APDA. Developer notes for individual models are also on the developer CDs.

For More Information

APDA is Apple's worldwide source for hundreds of development tools, technical resources, training products, and information for anyone interested in developing applications on Apple platforms. Customers receive the *APDA Tools Catalog* featuring all current versions of Apple development tools and the most popular third-party development tools. APDA offers convenient payment and shipping options, including site licensing.

To order products or to request a complimentary copy of the *APDA Tools Catalog*, contact

APDA
Apple Computer, Inc.
P.O. Box 319
Buffalo, NY 14207-0319

Telephone	1-800-282-2732 (United States) 1-800-637-0029 (Canada) 716-871-6555 (International)
Fax	716-871-6511
AppleLink	APDA
America Online	APDAorder
CompuServe	76666,2405
Internet	APDA@applelink.apple.com

Conventions and Abbreviations

This developer note uses the following typographical conventions and abbreviations.

Typographical Conventions

New terms appear in **boldface** where they are first defined.

Computer-language text—any text that is literally the same as it appears in computer input or output—appears in `Courier` font.

Hexadecimal numbers are preceded by a dollar sign ($). For example, the hexadecimal equivalent of decimal 16 is written as $10.

Note
A note like this contains information that is interesting but not essential for an understanding of the text. ◆

Sidebar

Sidebars are for digressions—information that is not part of the main discussion. A sidebar may contain background information that is interesting to know, information about a related subject, or technical details that are not required reading.

IMPORTANT

A note like this contains important information that you should read before proceeding. ▲

▲ **WARNING**

A note like this directs your attention to something that could cause damage or result in a loss of data. ▲

Standard Abbreviations

When unusual abbreviations appear in this book, the corresponding terms are also spelled out. Standard units of measure and other widely used abbreviations are not spelled out. Here are the standard units of measure used in this developer note:

A	amperes	mA	milliamperes
dB	decibels	µA	microamperes
GB	gigabytes	MB	megabytes
Hz	hertz	MHz	megahertz
in.	inches	mm	millimeters
k	1000	ms	milliseconds
K	1024	µs	microseconds
KB	kilobytes	ns	nanoseconds
kg	kilograms	Ω	ohms
kHz	kilohertz	sec.	seconds
kΩ	kilohms	V	volts
lb.	pounds	W	watts

Here are other abbreviations used in this developer note:

$*n*	hexadecimal value *n*
AC	alternating current
ADB	Apple Desktop Bus
CD-ROM	compact-disk read-only memory
CLUT	color lookup table
DESC	digital video decoder and scaler
EMI	electromagnetic interference

FPU	floating-point unit
IC	integrated circuit
IDE	integrated device electronics
IIC	inter-integrated circuit (an internal control bus)
I/O	input/output
IR	infrared
LS TTL	low-power Schottky TTL (a standard type of device)
MMU	memory management unit
MOS	metal-oxide semiconductor
NTSC	National Television Standards Committee (the standard system used for broadcast TV in North America and Japan)
NMI	nonmaskable interrupt
PAL	Phase Alternating Line system (the standard for broadcast TV in most of Europe, Africa, South America, and southern Asia)
PDS	processor-direct slot
PWM	pulse-width modulation
RAM	random-access memory
RGB	a video signal format with separate red, green, and blue components
RMS	root-mean-square
ROM	read-only memory
SANE	Standard Apple Numerics Environment
SCSI	Small Computer System Interface
SCC	serial communications controller
SECAM	the standard system used for broadcast TV in France and the former Soviet countries
SIMM	single inline memory module
S-video	a type of video connector that keeps luminance and chrominance separate; also called a Y/C connector
SWIM	Super Woz Integrated Machine, a custom IC that controls the floppy disk interface
TTL	transistor-transistor logic (a standard type of device)
VCR	video cassette recorder
VLSI	very large scale integration
VRAM	video RAM; used for display buffers
Y/C	a type of video connector that keeps luminance and chrominance separate; also called an S-video connector
YUV	a video signal format with separate luminance and chrominance components

CHAPTER 1

Introduction

The Macintosh LC 630 and Macintosh Quadra 630 computers are low-priced Macintosh models that incorporate Macintosh AV features (audio and video input and output) and an optional built-in TV tuner. These new computers have a low-profile case similar to that of the Macintosh Quadra 610 and require an external video monitor.

Summary of Features

Here is a summary of the hardware features of the Macintosh LC 630 and Macintosh Quadra 630 computers. Each feature is described more fully later in this note.

- Microprocessor: Motorola MC68040 or MC68LC040 microprocessor running at 66/33 MHz.
- RAM: 4 MB built in; expandable to 36 MB by means of one 72-pin SIMM.
- ROM: 1 MB on the main logic board.
- Case design: low-profile, for an external video monitor.
- Video output: built-in video interface provides up to 16 bits per pixel on Apple monitors up to 14-inch size plus the new 15-inch multiscan monitor.
- Video input: optional video card for real-time video display, capture, and overlay.
- TV receiver: optional internal TV tuner.
- Remote control: built-in remote control receiver for TV and audio CDs (TV tuner option includes remote control).
- Hard disks: one internal 3.5-inch IDE hard disk with 250 MB, 350 MB, or 500 MB capacity; external SCSI port for additional SCSI devices.
- Floppy disk: one internal 1.4 MB Apple SuperDrive.
- CD-ROM drive: optional internal double-speed CD-ROM player with tray loading.
- Standard Macintosh I/O ports: two serial ports, sound input and output jacks, a SCSI port, and an ADB port.
- Communications port: optional internal module can provide either a modem or an Ethernet interface.
- Sound: external jack for stereophonic sound in; front and rear jacks for stereophonic sound out; single built-in speaker; integrated sound from CD-ROM player.
- Expansion slot: accepts PDS cards designed for the Macintosh LC series.
- Power switch: controlled from keyboard and remote control.

Processor Clock Speeds

The MC68040 and MC68LC040 use two processor clocks: one for the system bus and another, at twice the speed, for the internal processor. In the Macintosh LC 630 and Macintosh Quadra 630 computers, the system bus clock runs at 33 MHz and the internal processor runs at 66 MHz.

Comparison With Macintosh Quadra 605 Computer

Electrically, the Macintosh LC 630 and Macintosh Quadra 630 computers are similar to the Macintosh Quadra 605. Table 1-1 compares the features of those computers.

Table 1-1 Comparisons with the Macintosh Quadra 605 computer

Features	Macintosh Quadra 605	Macintosh LC 630 and Macintosh Quadra 630
Processor type	MC68LC040	MC68040 (in the Quadra 630) MC68LC040 (in the LC 630)
Processor speed	50/25 MHz	66/33 MHz
Amount of RAM	4 MB–36 MB	4 MB–36 MB
RAM expansion	1 SIMM	1 SIMM
Video RAM	512 KB–1 MB (VRAM)	1 MB (DRAM)
Video input	none	optional module for video input, capture, and overlay
Sound capabilities	8 bits/channel; stereo in, mono record, stereo out	8 bits/channel; stereo in, mono record, stereo out
Remote control	none	built-in IR receiver
Floppy disk drive	1, internal	1, internal
ADB ports	1	1
Internal hard disk	1 (SCSI)	1 (IDE)
Internal CD-ROM	none	optional
External SCSI ports	1	1
Communications slot	none	1, for optional modem or Ethernet interface
Expansion slot	1 I/O slot (accepts PDS card for Macintosh LC series)	1 I/O slot (accepts PDS card for Macintosh LC series)

Models and Configurations

The Macintosh LC 630 and Macintosh Quadra 630 computers are available in several configurations, as shown inTable 1-2.

Table 1-2 Configurations of the computers

Model name	Amount of RAM	Size of hard disk	Built-in CD-ROM	Video input	TV tuner	FPU
Macintosh LC 630	4 MB	250 MB				
Macintosh LC 630	8 MB	250 MB	√			
Macintosh LC 630	8 MB	250 MB	√	√		
Macintosh LC 630	8 MB	350 MB	√			
Macintosh LC 630	8 MB	350 MB	√	√	√	
Macintosh Quadra 630	4 MB	250 MB				√

Note
A version of the Macintosh LC 630 will be sold in some regions as the Macintosh Performa 630. ◆

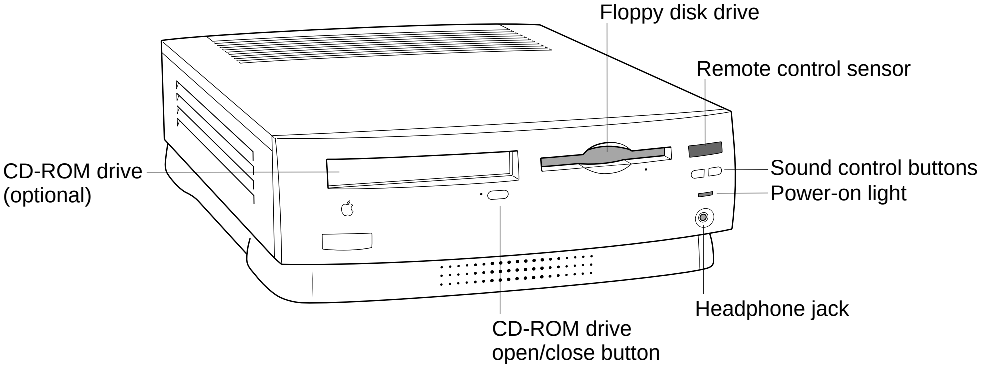
Exterior Features

The Macintosh LC 630 and Macintosh Quadra 630 computers have a low-profile case with about the same footprint as the Macintosh Quadra 605 computer. Either computer can function as a stand for any of the video monitors it can drive.

Front View

Figure 1-1 shows the front of a Macintosh LC 630 or Macintosh Quadra 630 computer. The front panel includes the built-in speaker, the openings for the floppy disk and CD-ROM drives, the CD-ROM open and close button, the sound control buttons, the headphone jack, the power-on light, and the IR sensor for the remote control.

Figure 1-1 Front view of the computer

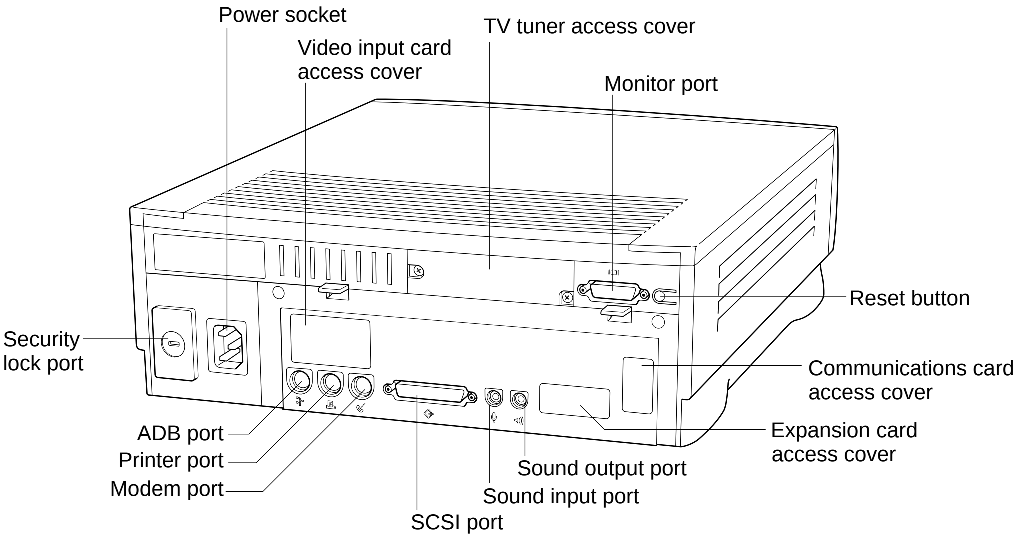


Back View

The back panel includes the power socket, the monitor port, the reset button, the I/O ports, and openings for I/O access to the expansion modules: the I/O expansion card, the communications card, and the video input card.

Figure 1-2 shows the back view of a Macintosh LC 630 or Macintosh Quadra 630 computer.

Figure 1-2 Back view of the computer



Access to the Logic Board

The logic board can be removed from the case so that the user can add expansion RAM or plug in an I/O expansion card. The logic board plugs into a connector at the front so that it can be removed by pulling it out the back.

Above the I/O connectors on the back of the computer are two screws and two projecting tabs. After removing the screws, the user can remove the back cover by pressing down on the tabs and pulling them outward. The user can then grasp the wire loop on the logic board and pull the board straight out the back of the case.

Sound Control Push Buttons

The Macintosh LC 630 and Macintosh Quadra 630 computers have a pair of push buttons on the front panel to control the sound volume: pressing the left button causes the volume to decrease, and pressing the right button causes the volume to increase. If the computer is playing a sound when the user presses one of the sound control push buttons, the volume changes as long as the user continues to press the button. If no sound is playing when the user presses a sound control push button, the computer plays an alert sound to confirm the new volume setting. The computer also allows the user to select sound features and control the volume by means of the Sound control panel.

Power On and Off

The user can turn the power off and on by pressing one of two buttons:

- the Power key on the keyboard
- the Power key on the remote control

If files are still open when the user attempts to turn off the computer by using either one of the Power keys or the Shut Down menu item, the system displays an alert box warning the user that files are open and should be closed to avoid loss of data.

Optional Modules

Several features of the Macintosh LC 630 and Macintosh Quadra 630 computers are implemented as plug-in modules available either as a configuration option at the time of purchase or as a later upgrade. The modules are designed so that they can be installed by the user.

TV Tuner Module

The TV tuner module turns the computer into a television receiver, complete with remote control. The features of the TV tuner module are similar to those of the TV tuner in the earlier Macintosh TV computer, with two improvements: the TV picture is in its own window on the desktop, and the TV signal is carried in YUV format for improved picture clarity.

Why YUV Looks Clearer

You may be wondering how the digital YUV format used in the Macintosh LC 630 and Macintosh Quadra 630 computers provides a clearer TV picture than the RGB format used in the Macintosh TV computer—after all, picture information can be freely converted between the two formats. The difference is due to the way the bits are allocated. The RGB format used in the Macintosh TV is a 16-bit format using 5 bits each for red, green, and blue, with the remaining bit unused. The YUV format used in the Macintosh LC 630 and Macintosh Quadra 630 computers is also a 16-bit format, with 8 bits for the Y (luminance) channel and 8 bits for the U and V (chrominance) channels to share by multiplexing. The YUV format looks clearer because the YUV format carries more levels of luminance information.

The features of the TV tuner module in the Macintosh LC 630 and Macintosh Quadra 630 computers are

- ability to tune 181 broadcast and cable channels (US version)
- coaxial connector for TV antenna or cable input (F-type connector in US and Japanese versions; IEC-type connector in Europe)
- TV picture in a resizable and movable window
- YUV format for improved clarity (see sidebar on next page)
- support for closed captioning and teletext
- software password protection
- automatic and manual channel programming
- single remote control for TV and for playback of audio CDs

The TV tuner module is available in versions for NTSC, PAL, and SECAM television systems.

The TV picture appears in its own window. The default size of the window is 320 by 240 pixels. The user can resize the TV window up to a maximum size of 640 by 480 pixels or down to a minimum size of 160 by 120 pixels. The resolution of the TV picture does not increase at the larger window sizes; instead, the image is enlarged by doubling the size of the pixels.

The TV tuner module works in conjunction with the video input module, which converts the video data into digital YUV format and stores it in the display buffer.

The TV tuner comes with a remote control device similar to the one used with the Macintosh TV computer. The user can switch channels either by using the remote control or by typing the channel numbers on the keyboard. The user can toggle between the current and previous channel by pressing the Tab key on the keyboard. Each time the channel changes, the computer displays the channel name (assigned by the user) on the picture in the video window.

The user can customize the operation of the TV tuner by adding or removing TV channels that are unused or unwanted. The computer can program the channels automatically, scanning through all available channels and disabling those that do not have a valid signal. When the user then scans for the next channel by using the remote control or the Tab key on the keyboard, the tuner skips the disabled channels.

The software that supports the TV tuner module is an application called Apple Video Player. The application includes password protection for the disabled channels. Parents might use this feature to prevent children from watching undesirable channels.

The software allows the user to capture or freeze a single frame of video or record a segment of video as a QuickTime Movie. The TV window cannot be resized while the computer is recording a movie.

Video Input Module

The video input module accepts video from an external source and dispays it in a window on the computer's display. The features of the video input module are

- accepts video input in NTSC, PAL, or SECAM format
- connectors for stereo sound, composite video, and S-video (Y/C)
- video display in a 320-by-240-pixel window
- pixel doubling for 640-by-480-pixel maximum display
- video overlay capability
- YUV format for digital video input
- a digital video connector for adding a video processor on an expansion card

The video input module provides AV features similar to those of the Macintosh Quadra 660AV, with one key improvement. Whereas the Macintosh Quadra 660AV digitizes color video using a 16-bit RGB format, the video input module uses a digital YUV format. Because a standard television signal has more information in its chrominance channel than in its luminance channels, digitizing the video signal as YUV format results in a clearer picture.

The video input module can accept video input from either an external device such as a VCR or camcorder, or from the internal TV tuner module. The external device can be connected to the video input module either through the composite video connector or the S-video connector.

The default window size is 320 by 240 pixels; the user can resize the window up to 640 by 480 pixels—the full screen on a 14-inch monitor. The large image uses pixel doubling of the 320 by 240 pixel image.

Note
The video input module does not work on video monitors with screens larger than 15-inch size. It will work with 800-by-600-pixel displays that have a 60 Hz refresh rate, but not with that size display at a 72 Hz refresh rate. ◆

The video input module plugs into a dedicated slot on the main logic board. The slot connector is a 60-pin microchannel connector. The module fits only its proper slot and only in the proper orientation so that the user can safely install the video input module.

The video input module has a separate connector called the DVA (digital video application) connector. The DVA connector makes the digitized video data available to a card in the I/O expansion slot. Such a card can contain a hardware video compressor or other video processor. For more information, see the section "The DVA Connector" beginning on page 55.

Video Output Module

The video output module is an external module that uses the 15-pin video output connector and provides a video signal for a separate television display. The video output can also be recorded on a VCR.

A standard television monitor has overscan: to ensure that the entire screen is filled, the image is larger than the screen, causing the outer parts of the picture to be cut off by the edges of the screen. The video output module provides a video signal with a 640-by-480-pixel display inside the safe display area so that no information is lost.

The video output module supports video mirror mode: in that mode, the image on the television display is the same as that on the computer's primary video monitor. That mode of operation is appropriate, for example, for presentations, so that the audience and the presenter can see the same displays.

Apple expects to provide separate video output modules for NTSC and PAL systems.

Communications Modules

The main logic board in the Macintosh LC 630 and Macintosh Quadra 630 computers has a communications slot that is compatible with the communications slot first introduced in the Macintosh LC 575 computer. The slot allows the computer to support a communications module without occupying the expansion slot. A communications module can be installed by either the user or the dealer.

The communications slot in the Macintosh LC 630 and Macintosh Quadra 630 computers supports all communications cards developed for the Macintosh LC 575, including

- the 10BaseT (twisted pair) ethernet card
- the 10Base2 (thin coax) ethernet card
- the AAUI (Apple standard) ethernet card
- the 14.4 fax/data modem card

Compatibility Issues

The Macintosh LC 630 and Macintosh Quadra 630 computers incorporate several changes from earlier desktop models. This section describes key issues you should be aware of to ensure that your hardware and software work properly with these new models. The topics described here are covered in more detail in later parts of this developer note.

Expansion Slot

The I/O expansion slot in the Macintosh LC 630 and Macintosh Quadra 630 computers is compatible with the PDS slot in the Macintosh LC family of computers, but it is not a PDS slot. Like the expansion slot in the Macintosh Quadra 610, the I/O expansion slot in the Macintosh LC 630 and Macintosh Quadra 630 computers supports many PDS cards designed to operate with the MC68030 bus, including both bus masters, such as Apple Computer's ethernet expansion card, and bus slaves, such as display cards.

While the I/O expansion slot accepts PDS cards designed for the Macintosh LC II and LC III, some of those cards do not work. Cards that are incompatible with the I/O expansion slot include

- cards designed to work as coprocessors with an MC68020 or an MC68030 or as replacements for those microprocessors. Such cards include accelerators, 68882 FPU cards, and cache cards. That type of card won't work because the microprocessor is different and because the slot signals are not connected directly to the microprocessor.
- cards with drivers that include incompatible code. Some drivers that do not follow Apple Computer's programming guidelines won't work on machines that use the MC68040 microprocessor. For example, some of those drivers write directly to the cache control register in an MC68030. Such code won't work on an MC68040.
- cards with drivers that include code to check the `gestaltMachineType` value and refuse to run on a newer CPU. The idea is to protect users by refusing to run on a machine that the cards haven't been tested on. Such cards have compatibility problems with all new Macintosh models.

Apple IIe Card for the Macintosh LC

The Macintosh LC 630 and Macintosh Quadra 630 computers operate only in 32-bit addressing mode, so they do not support the Apple IIe card for the Macintosh LC.

IDE Hard Disk

The internal hard disk in the Macintosh LC 630 and Macintosh Quadra 630 computers is an IDE drive, not a SCSI drive. This could cause compatibility problems for hard disk utility programs.

CHAPTER 2

Architecture

This chapter describes the architecture of the Macintosh LC 630 and Macintosh Quadra 630 computers. It describes the major components of the main logic board: the microprocessor, the custom ICs, and the display RAM. It also includes a simplified address map.

Block Diagram and Main ICs

The architecture of the Macintosh LC 630 and Macintosh Quadra 630 computers is similar to the architecture of the Macintosh Quadra 605. Figure 2-1 shows the block diagram.

The computer architecture has two internal buses. The system bus is connected directly to the main processor and runs at the same clock rate. The I/O bus is partially buffered from the main processor and runs at a 16 MHz clock rate. The PrimeTime II custom IC, described later, buffers the data portion of the I/O bus and provides a compatible interface for I/O devices and software designed for use with the MC68030 miroprocessor.

Microprocessor

The Macintosh Quadra 630 computer uses the Motorola MC68040 microprocessor. The Macintosh LC 630 computer uses the Motorola MC68LC040, a low-cost version of the MC68040 without the built-in FPU. The MC68LC040 has all the other features of the MC68040; the performance of the MC68LC040 is the same as that of the MC68040 except for floating-point operations.

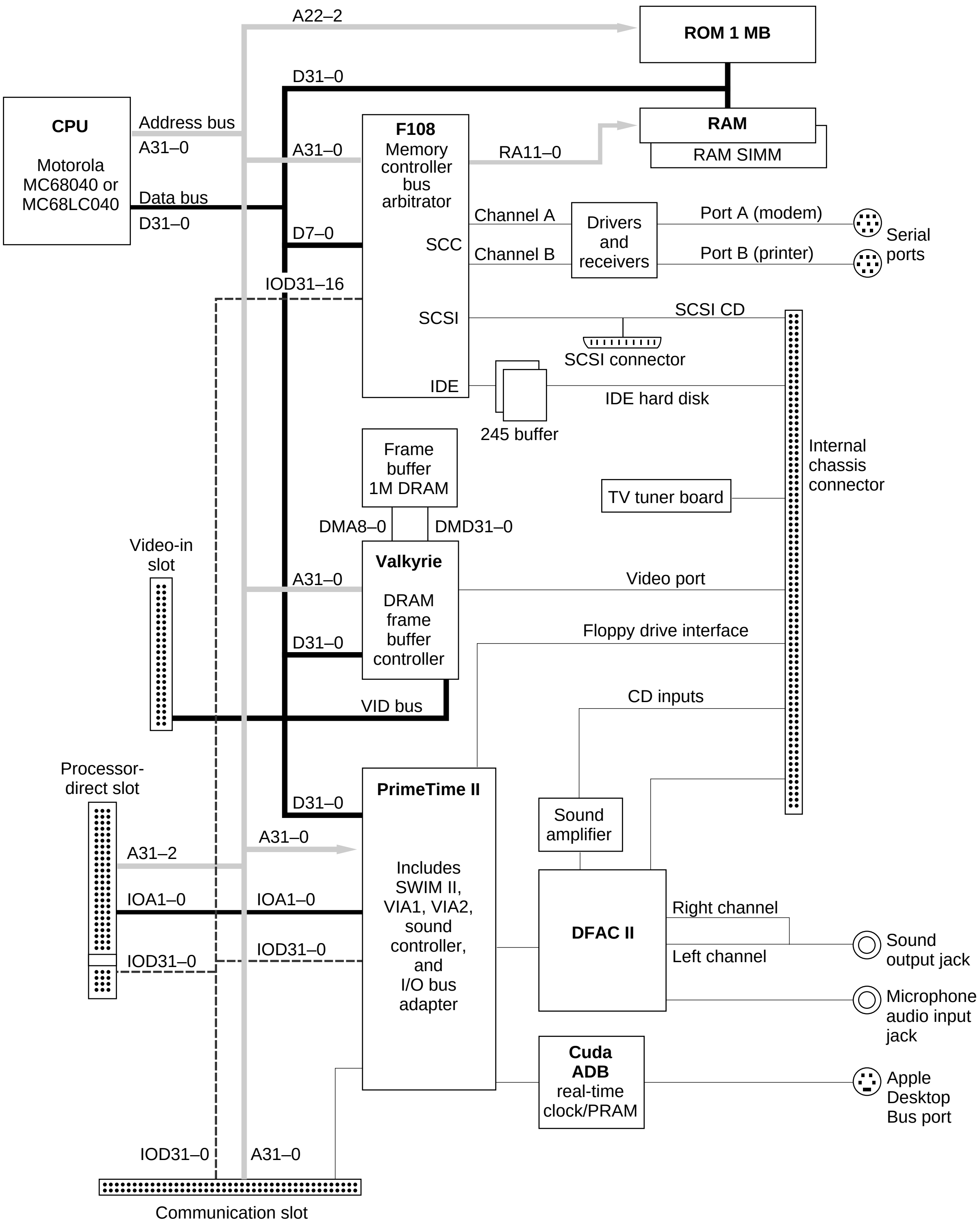
The MC68040 and MC68LC040 microprocessors use two processor clocks: one for the system bus and another, at twice the speed, for the internal circuits. In the Macintosh Quadra 630 and Macintosh LC 630 computers, the system bus clock runs at 33 MHz and the internal processor runs at 66 MHz.

The MC68LC040 microprocessor in the Macintosh LC 630 computer is installed in a socket. That makes it possible to upgrade to an MC68040 by removing the MC68LC040 from its socket and replacing it with the MC68040.

IMPORTANT

An expansion board cannot provide an FPU coprocessor because the MC68LC040 microprocessor does not support the coprocessor interface and the signals on the expansion connector are not connected directly to the CPU. ▲

Figure 2-1 Block diagram



Custom ICs

The architecture of the Macintosh LC 630 and Macintosh Quadra 630 computers is designed around five large custom VLSI integrated circuits:

- the F108 memory controller and I/O support IC
- the PrimeTime II I/O subsystem and buffer
- the DFAC II sound input processor
- the Cuda ADB controller
- the Valkyrie video CLUT and DAC

The computer also uses several standard ICs that are used in other Macintosh computers. This section describes only the custom ICs.

F108 IC

The F108 IC performs the system memory control functions. It also includes circuitry equivalent to the SCC and SCSI controller ICs. The functional blocks in the F108 include the following:

- control logic for the system ROM and RAM
- SCSI controller
- SCC serial I/O controller
- IDE hard disk interface controller

The F108 IC is attached to the system bus and provides the control and timing signals for the system ROM and RAM. The memory control logic supports byte, word, longword, and line accesses to the system memory. If an access is not aligned to the appropriate address boundary, that access requires multiple data transfers on the bus.

Note
The memory control logic in the F108 IC is similar to that in the MEMCjr IC used in the Macintosh Quadra 605. ◆

The SCSI controller in the F108 IC is just like an NCR 53C96. PrimeTime II provides the interface to the SCSI controller and also provides longword accumulation of SCSI data for better performance. In the Macintosh LC 630 and Macintosh Quadra 630 computers the clock signal to the SCSI controller is 16.5 MHz.

The SCC circuitry in the F108 IC is an 8-bit device just like the 8530 SCC. The `PCLK` signal to the SCC is an 8 MHz clock.

PrimeTime II IC

The PrimeTime II IC supports the I/O bus. It combines functions performed by several ICs in previous Macintosh designs. The PrimeTime II IC includes

- data bus buffers for the internal I/O bus
- address decoding for I/O devices

- dynamic bus sizing and data routing for the I/O bus
- interface adapters VIA1 and VIA2
- interrupt controls
- a SWIM II floppy disk controller
- sound control logic and buffers

The PrimeTime II IC serves as a bridge between the system bus and the I/O bus.

The PrimeTime II IC provides the data bus features of the MC68030 that the MC68040 does not provide. Those features are **byte steering,** which allows 8-bit and 16-bit devices to be connected to a fixed byte lane, and **dynamic bus sizing,** which allows software to read and write longwords to 8-bit and 16-bit devices. Those features allow the computer to work with existing I/O software designed for the MC68030.

The PrimeTime II IC also contains the sound control logic and the sound input and output buffers. There are three separate buffers—one for sound input and two for stereo sound output—so the computer can record sound input and process sound output simultaneously.

DFAC II IC

The DFAC II custom IC contains the sound input processing devices. The DFAC II includes

- input AGC comparators
- antialias filtering
- an A/D converter for input
- a PWM converter for output

The DFAC II IC does not include the sound countrol logic and the input and output buffers; those are part of the PrimeTime II IC.

For sound input, the DFAC II processes the signal from the sound input jack through a sound input amplifier with automatic gain control, an input filter, an A/D converter, and the necessary switching circuits. The DFAC II sends the resulting stream of digital sound data to the PrimeTime II, which stores the data in its input buffer.

For sound output, circuits in the DFAC II take data from the sound output buffers and generate stereo pulse-width-modulated (PWM) signals. The DFAC II merges the sound playthrough signal with the PWM signals and sends the combined signals to an external stereo PWM converter IC. After low-pass filtering in the PWM converter, the signals go to the sound output jacks and to a separate amplifier for the built-in speaker. See the section “Sound” beginning on page 30.

Cuda IC

The Cuda IC is a custom version of the Motorola MC68HC05 microcontroller. It provides several system functions, including

- the ADB interface
- parameter RAM
- the real-time clock
- program control of the power supply (soft power)
- the programming interface to devices on the IIC (inter-integrated circuit) bus

The devices on the IIC bus include the DFAC II sound IC, the digital video decoder and scaler (DESC) on the video input module, and the Cyclops IC, which is the controller for the remote control receiver. The computer reads and writes status and control information to those devices by commands to the Cuda IC.

Valkyrie IC

The Valkyrie IC is a custom IC containing the logic for the video display. It includes the following functions:

- display memory controller
- video CLUT (color lookup table)
- video DAC (digital-to-analog converter)

Note
The display memory controller in the Valkyrie IC is similar to the DAFB IC used in the Macintosh Quadra 700 and 900 computers and to the display portion of the MEMCjr IC used in the Macintosh Quadra 605 computer. ◆

A separate data bus handles data transfers between the Valkyrie IC and the display memory. The display memory data bus is 32 bits wide, and all data transfers consist of 32 bits at a time. The Valkyrie IC breaks each 32-bit data transfer into several pixels of the appropriate size for the current display mode—1, 2, 4, 8, or 16 bits per pixel. The Valkyrie IC does not support 24 bits per pixel.

To keep up with the large amount of data that must be transferred into and out of the display memory, the Valkyrie IC has several internal buffers. Besides input and output buffers for display data, the Valkyrie IC also has a buffer for both addresses and data being sent from the main processor to the display. That buffer can hold up to four transactions, allowing the main processor to complete a write instruction to the display memory and continue processing without waiting for some other transaction that might be taking place on the display memory bus.

The CLUT in the Valkyrie custom IC provides color palettes for 4-bit and 8-bit display modes. In 16-bit display mode, the CLUT is used to provide gamma correction for the stored color values. With a black-and-white or monochrome display mode, all three color components (R, G, and B) are the same.

The Valkyrie IC uses several clocks. Its transactions with the CPU are synchronized to the CPU_BCLK signal. Data transfers from the frame-buffer DRAM are clocked by the MEM_CLK signal, which runs at 60 MHz. Data transfers to the CLUT and the video output are clocked by the dot clock, which has a different rate for different display monitors.

For more information about the interaction between the Valkyrie IC, the display memory, and the main processor, see the section "Display RAM" on this page.

Bus Arbitration

The system bus can support one or two bus masters, including the main processor and one I/O bus master. The I/O master has higher priority. Either bus master can park on the bus as long as no higher priority master requests the bus.

The bus request from the I/O bus master is initiated by the PrimeTime II IC and comes from one of two sources: the I/O expansion card or the communications card. Devices on those cards are not connected directly to the system bus; they arbitrate the bus by way of the I/O bus and the PrimeTime II IC. See the section "Bus Master on a Card" beginning on page 53.

Display RAM

The display memory in the Macintosh LC 630 and Macintosh Quadra 630 computers is separate from the main memory. To reduce the cost of the computer, the display memory is implemented with DRAM devices instead of more expensive VRAM devices. The display memory consists of 1 MB of 60 ns DRAM devices configured to make a 32-bit data bus. The display memory cannot be expanded.

The display memory contains three separate frame buffers. The first frame buffer holds the graphics data—the display that is generated by the computer. The other two frame buffers hold video data from the video input module. The video data frame buffers are used alternately: while one is supplying data to be sent to the video monitor, the other is receiving the next frame of video input.

The display data generated by the computer can have pixel depths of 1, 2, 4, 8, or 16 bits for monitors up to 640 by 480 pixels and 1, 2, 4, or 8 for larger monitors and the 800-by-600-pixel display on the multiscan monitor. Data from the video input module is always stored and transferred at 16 bits per pixel. The video frame buffers support live video in a 320-by-240-pixel frame at 30 frames per second.

Note
The Macintosh LC 630 and Macintosh Quadra 630 computers cannot dispay live video from the video-in module on monitors larger than 800 by 600 pixels. Apple does not recommend the use of such monitors with these computers. ◆

The Macintosh LC 630 and Macintosh Quadra 630 computers can display video in a window inside the computer graphics display. The Valkyrie IC has registers that contain the starting location of the video window within the display, the starting address of the video data in the video buffer, and the size of the video window.

Note
Because the Macintosh LC 630 and Macintosh Quadra 630 computers operate only in 32-bit addressing mode, they do not support the Apple IIe Card for the Macintosh LC. ◆

Address Map

The Macintosh LC 630 and Macintosh Quadra 630 computers support only 32-bit addressing. Figure 2-2 shows a simplified address map.

Note
Developers should not use actual hardware addresses in applications; they should always communicate with hardware devices by means of system software. ◆

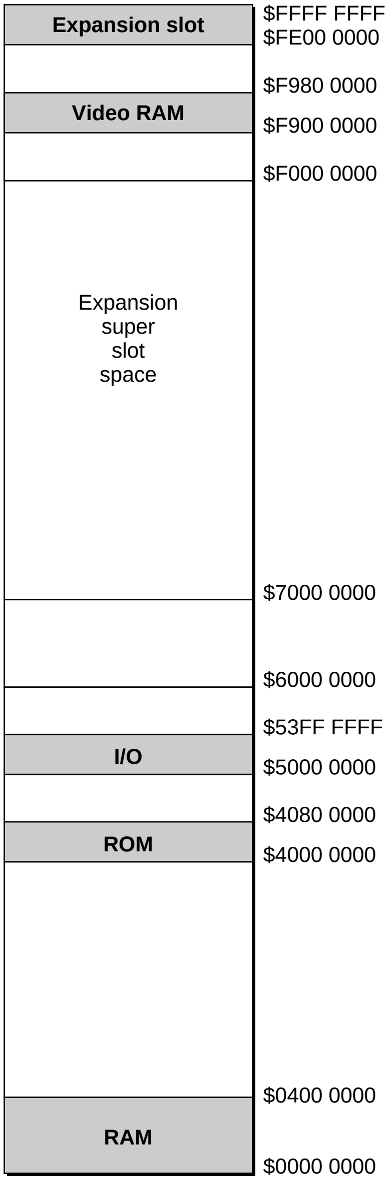
RAM Addresses

The first 1 GB of the address space is reserved for RAM. The actual amount of RAM installed can be from 4 MB to 36 MB. At startup time, a routine in the ROM determines the amount of RAM available and stores the size in a low-memory global variable.

Display RAM Addresses

The Macintosh LC 630 and Macintosh Quadra 630 computers use separate DRAM to store the display buffers. The display RAM occupies dedicated address space starting at $F900 0000, as shown in Figure 2-2.

Figure 2-2 Simplified address map



Addresses for I/O Expansion Cards

The I/O expansion card uses address space from $FE00 0000 to $FEFF FFFF, corresponding to NuBus slot $E, and from $E000 0000 to $EFFF FFFF, corresponding to NuBus™ Super Slot $E. For more information, see the section "Card Address Space" on page 55.

CHAPTER 3

I/O Features

This chapter describes both the built-in I/O devices and the interfaces for external I/O devices. It also describes the types of video monitors that can be used with the Macintosh LC 630 and Macintosh Quadra 630 computers.

Serial I/O Ports

The Macintosh LC 630 and Macintosh Quadra 630 computers have two serial ports, one for a printer and one for a modem. Both serial ports have 9-pin mini-DIN sockets that accept either 8-pin or 9-pin plugs. Figure 3-1 shows the mechanical arrangement of the pins on the serial port sockets; Table 3-1 shows the signal assignments.

Figure 3-1 Serial port sockets

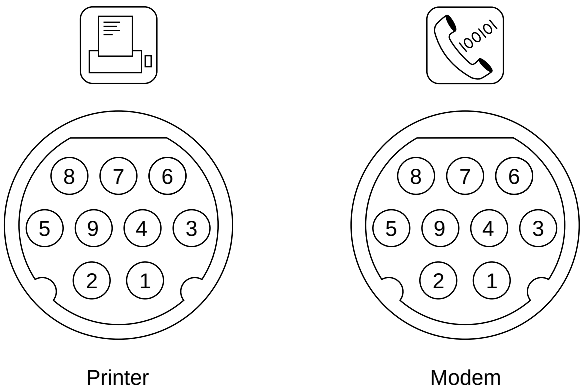


Table 3-1 Serial port signals

Pin number	Signal description
1	Handshake output
2	Handshake input
3	Transmit data –
4	Ground
5	Receive data –
6	Transmit data +
7	General-purpose input
8	Receive data +
9	+5 volts

Pin 9 on each serial connector provides +5 V power from the ADB power supply. An external device should draw no more than 100 mA from that pin. The total current available for all devices connected to the +5 V supply for the ADB and the serial ports is 500 mA. Excessive current drain will cause a fuse to interrupt the +5 V supply; the fuse automatically resets when the load returns to normal.

Both serial ports include the GPi (general-purpose input) signal on pin 7. The GPi signal for each port connects to the corresponding data carrier detect input on the SCC portion of the F108 custom IC, described in Chapter 2. On serial port A (the modem port), the GPi line can be connected to the receive/transmit clock (RTxCA) signal on the SCC. That connection supports devices that provide separate transmit and receive data clocks, such as synchronous modems. For more information about the serial ports, see *Guide to the Macintosh Family Hardware*, second edition.

ADB Port

The Apple Desktop Bus (ADB) port on the Macintosh LC 630 and Macintosh Quadra 630 computers is functionally the same as on other Macintosh computers.

The ADB is a single-master, multiple-slave serial communications bus that uses an asynchronous protocol and connects keyboards, graphics tablets, mouse devices, and other devices to the computer. The custom ADB microcontroller drives the bus and reads status from the selected external device. A 4-pin mini-DIN connector connects the ADB to the external devices. Table 3-2 lists the ADB connector pin assignments. For more information about the ADB, see *Guide to the Macintosh Family Hardware*, second edition.

Table 3-2 ADB connector pin assignments

Pin number	Name	Description
1	ADB	Bidirectional data bus used for input and output. It is an open-collector signal pulled up to +5 volts through a 470-ohm resistor on the main logic board.
2	PSW	Power-on signal that generates reset and interrupt key combinations.
3	+5V	+5 volts from the computer.
4	GND	Ground from the computer.

Note
The total current available for all devices connected to the +5V pins on the ADB and the modem port is 500 mA. Each device should use no more than 100 mA. ◆

Floppy Disk Drive

The Macintosh LC 630 and Macintosh Quadra 630 computers have one internal high-density floppy disk drive (Apple SuperDrive). The drive is connected to a 20-pin connector on a cable that is connected to the main logic board by the internal chassis connector, as shown in Figure 2-1 on page 13. Table 3-3 shows the pin assignments on the floppy disk connector.

Table 3-3 Pin assignments on the floppy disk connector

Pin number	Signal name	Signal description
1	GND	Ground
2	PH0	Phase 0: state control line
3	GND	Ground
4	PH1	Phase 1: state control line
5	GND	Ground
6	PH2	Phase 2: state control line
7	GND	Ground
8	PH3	Phase 3: register write strobe
9	n.c.	Not connected
10	/WRREQ	Write data request
11	+5V	+5 volts
12	SEL	Head select
13	+12V	+12 volts
14	/ENBL	Drive enable
15	+12V	+12 volts
16	RD	Read data
17	+12V	+12 volts
18	WR	Write data
19	+12V	+12 volts
20	n.c.	Not connected

IDE Hard Disk

The Macintosh LC 630 and Macintosh Quadra 630 computers have an internal hard disk using the standard IDE interface. This interface, used for IDE drives on IBM AT–compatible computers, is also referred to as the ATA interface. The implementation of the ATA interface on the Macintosh LC 630 and Macintosh Quadra 630 computers is a subset of the ATA interface specification, ANSI proposal X3T9.2/90-143, Revision 3.1.

Hard Disk Specifications

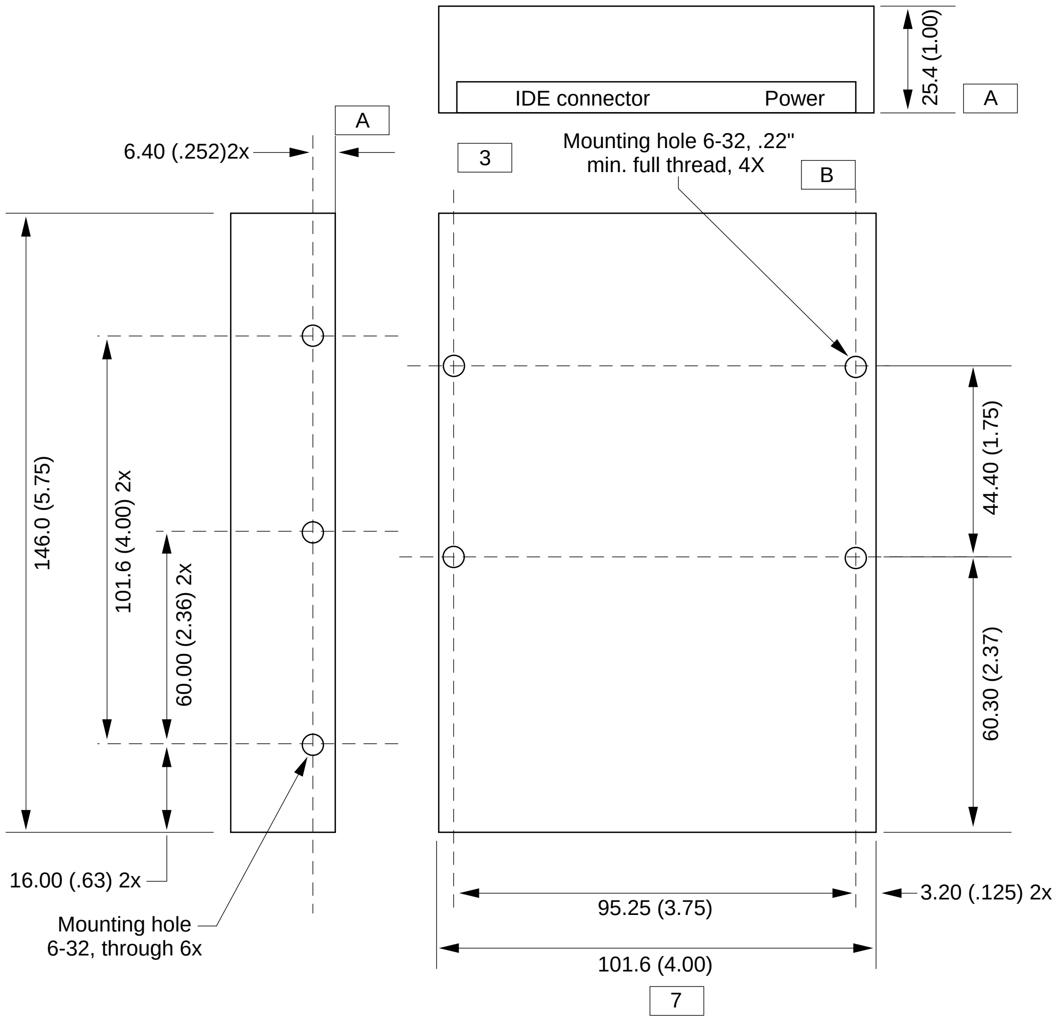
Figure 3-2 on page 26 shows the maximum dimensions of the hard disk and the location of the mounting holes. As the figure shows, the minimum clearance between conductive components and the bottom of the mounting envelope is 0.5 mm.

Hard Disk Connectors

The internal hard disk has a standard 40-pin IDE connector and a separate 4-pin power connector. The 40-pin connector cable is part of the cable harness attached to the main logic board by the internal chassis connector, as shown in Figure 2-1 on page 13. The power cable is attached directly to the power supply.

The exact locations of the IDE connector and the power connector are not specified, but the relative positions must be as shown in Figure 3-2 so that the cables and connectors will fit.

Figure 3-2 Maximum dimensions of the hard disk



Notes:

1 A Defined by plane of bottom mount holes
2 B Defined by center line of bottom mount holes
3 40-pin IDE and 4-pin power connector placement must not be reversed
4 All dimensions in MM (inch)
5 Drawing not to scale
6 Tolerances .X = $\pm$0.50, .XX = $\pm$0.25
7 Dimension to be measured at center line of side-mount holes
8 Minimum 0.5 MM clearance from any conductive PCB components to A

Pin Assignments

Table 3-4 shows the pin assignments on the 40-pin IDE hard disk connector. A slash (/) at the beginning of a signal name indicates an active-low signal.

Table 3-4 Pin assignments on the IDE hard disk connector

Pin number	Signal name	Pin number	Signal name
1	/RESET	2	GROUND
3	DD7	4	DD8
5	DD6	6	DD9
7	DD5	8	DD10
9	DD4	10	DD11
11	DD3	12	DD12
13	DD2	14	DD13
15	DD1	16	DD14
17	DD0	18	DD15
19	GROUND	20	KEY
21	Reserved	22	GROUND
23	DIOW	24	GROUND
25	DIOR	26	GROUND
27	IORDY	28	Reserved
29	Reserved	30	GROUND
31	INTRQ	32	/IOCS16
33	DA1	34	Reserved
35	DA0	36	DA2
37	/CS0	38	/CS1
39	Reserved	40	GROUND

Note

The IDE data bus is connected to the I/O bus through bidirectional bus buffers. To match the big-endian format of the MC68030-compatible bus, the bytes are swapped. The lower byte of the IDE data bus, DD(0–7), is connected to the high byte of the upper word of the I/O bus, IOD(24–31). The higher byte of the IDE data bus, DD(8–15), is connected to the low byte of the upper word of the I/O bus, IOD(16–23). ◆

IDE Signal Descriptions

Table 3-5 describes the signals on the IDE hard disk connector.

Table 3-5 Signals on the IDE hard disk connector

Signal name	Signal description
DA(0–2)	IDE device address; used by the computer to select one of the registers in the IDE drive. For more information, see the descriptions of the CS0 and CS1 signals.
DD(0–15)	IDE data bus; buffered from IOD(16–31) of the computer's I/O bus. DD(0–15) are used to transfer 16-bit data to and from the drive buffer. DD(8–15) are used to transfer data to and from the internal registers of the drive, with DD(0–7) driven high when writing.
/CS0	IDE register select signal. It is asserted low to select the main task file registers. The task file registers indicate the command, the sector address, and the sector count.
/CS1	IDE register select signal. It is asserted low to select the additional control and status registers on the IDE drive.
IORDY	IDE I/O ready; when driven low by the drive, signals the CPU to insert wait states into the I/O read or write cycles.
/IOCS16	IDE I/O channel select; asserted low for an access to the data port. The computer uses this signal to indicate a 16-bit data transfer.
/DIOR	IDE I/O data read strobe.
/DIOW	IDE I/O data write strobe.
INTRQ	IDE interrupt request. This active high signal is used to inform the computer that a data transfer is requested or that a command has terminated.
/RESET	Hardware reset to the drive; an active low signal.
Key	This pin is the key for the connector.

SCSI Bus

The Macintosh LC 630 and Macintosh Quadra 630 computers have a SCSI bus for the internal CD-ROM player and one or more external SCSI devices.

SCSI Connectors

The SCSI connector for the internal CD-ROM drive is a 50-pin connector with the standard SCSI pin assignments. The external SCSI connector is a 25-pin D-type connector with the same pin assignments as other Apple SCSI devices. Table 3-6 shows the pin assignments on the internal and external SCSI connectors.

Table 3-6 Pin assignments for the SCSI connectors

Pin number (internal 50-pin)	Pin number (external 25-pin)	Signal name	Signal description
2	8	/DB0	Bit 0 of SCSI data bus
4	21	/DB1	Bit 1 of SCSI data bus
6	22	/DB2	Bit 2 of SCSI data bus
8	10	/DB3	Bit 3 of SCSI data bus
10	23	/DB4	Bit 4 of SCSI data bus
12	11	/DB5	Bit 5 of SCSI data bus
14	12	/DB6	Bit 6 of SCSI data bus
16	13	/DB7	Bit 7 of SCSI data bus
18	20	/DBP	Parity bit of SCSI data bus
25	–	n.c.	Not connected
26	25	TPWR	+5 V terminator power
32	17	/ATN	Attention
36	6	/BSY	Bus busy
38	5	/ACK	Handshake acknowledge
40	4	/RST	Bus reset
42	2	/MSG	Message phase
44	19	/SEL	Select
46	15	/C/D	Control or data
48	1	/REQ	Handshake request
50	3	/I/O	Input or output
20, 22, 24, 28, 30, 34, and all odd pins except pin 25	7, 9, 14, 16, 18, and 24	GND	Ground

SCSI Bus Termination

In the Macintosh LC 630 and Macintosh Quadra 630 computers, the internal end of the bus is terminated by a 220/330 passive terminator. The terminator is located on the main logic board near the portion of the internal chassis connector that contains the signals for the internal CD-ROM drive. The internal CD-ROM drive does not include a terminator.

Sound

Like other Macintosh computers, the Macintosh LC 630 and Macintosh Quadra 630 computers can create sounds digitally and play the sounds through its internal speaker or send the sound signals out through the sound output jacks. These computers can also record sound from several sources: a microphone connected to the sound input jack, the video input module, or a compact disc in the CD-ROM player.

Sound Output Jacks

The Macintosh LC 630 and Macintosh Quadra 630 computers have two sound output jacks, one on the front and one on the back. Both output jacks are connected to the sound amplifier; the jack on the front is intended for ease of access when connected to a pair of headphones. Inserting a plug into either jack disconnects the internal speaker.

Sound Input Jack

The Macintosh LC 630 and Macintosh Quadra 630 computers have a sound input jack on the back for connecting an external microphone or other sound source. The sound input jack accepts a standard 1/8-inch phone plug, either monophonic or stereophonic (two signals plus ground).

The sound input jack accepts either the Apple PlainTalk line-level microphone or a pair of line-level signals by way of a separate adapter. The internal circuitry mixes the stereophonic signals into a monophonic signal for digitization.

Note
The Apple PlainTalk microphone requires power from the main computer, which it obtains by way of an extra-long, 4-conductor plug that makes contact with a 5-volt pin inside the sound input jack. ◆

IMPORTANT
The microphone for the Macintosh LC and LC II does not work with the Macintosh LC 630 and Macintosh Quadra 630 computers; they require the line-level signal provided by the Apple PlainTalk microphone. ▲

Sound Input Specifications

The sound input jack has the following electrical characteristics:

- Input impedance: 100k ohms
- Average line level: 100 mV RMS
- Average microphone level: 70 mV RMS
- Maximum input level: 1.8 V RMS

Routing of the Sound Signals

Sound input signals on the Macintosh LC 630 and Macintosh Quadra 630 computers can be routed in two ways: they can be recorded (digitized) or they can be sent directly to the sound outputs and speakers, bypassing the sound IC. Table 3-7 lists the sound sources and shows how each one can be routed.

Table 3-7 Sound sources and routing

Sound source	Record	Bypass
Sound input jack	√	–
Modem	√	–
CD-ROM player	√	√
Video sound (video input module)	√	√

Digitizing Sound

The Macintosh LC 630 and Macintosh Quadra 630 computers digitize and record sound as 8-bit samples. The computers can use either of two sampling rates: 11k samples per second and 22k samples per second. The sound circuits include input and output filters with switchable cutoff (–3 dB) frequencies that correspond to the two sampling rates: 3.5 kHz cutoff for the 11k sample rate and 7 kHz cutoff for the 22k sample rate.

The sound system always plays samples at the 22k sampling rate; when playing samples recorded at the 11k sampling rate, the software writes each sample to the sound buffer twice.

Sound Modes

The sound mode is selected by a call to the Sound Manager. The sound circuitry normally operates in one of three modes:

- Sound playback: computer-generated sound is sent to the speaker and the sound output jacks.
- Sound playback with playthrough: computer sound and sound input are mixed and sent to the speaker and the sound output jacks.

- Sound record with playthrough: input sound is recorded and also fed through to the speaker and the sound output jacks.

When recording from a microphone, applications should reduce the playthrough volume to prevent possible feedback from the speaker to the microphone.

The PrimeTime II IC provides separate sound buffers for input and for stereo output, so the computer can record and send digitized sound to the sound outputs simultaneously.

Keyboard

A keyboard is included with some models of the Macintosh LC 630 and Macintosh Quadra 630 computers.

The keyboard has a Power key, identified by the symbol p. When the user chooses Shut Down from the Special menu, a dialog appears telling the user that it is now safe to shut off the computer. The user then turns off the power by pressing the Power key .

There are no programmer’s switches on the Macintosh LC 630 and Macintosh Quadra 630 computers, so the user invokes the reset and NMI functions by pressing Command key combinations while holding down the Power key, as shown in Table 3-8. The Command key is identified by the symbols and x.

Note
The user must hold down a key combination for at least 1 second to allow the ADB microcontroller enough time to respond to the NMI or hard-reset signal. ◆

Table 3-8 Reset and NMI key combinations

Key combination	Function
Command-Power (x-p)	NMI (always active)
Control-Command-Power (Control-x-p)	Reset

Note
The NMI in the Macintosh LC 630 and Macintosh Quadra 630 computers can always be activated from the keyboard. This is a change from the Macintosh LC computer, where keyboard activation of the NMI function can be disabled by the software. ◆

External Video Monitors

The Macintosh LC 630 and Macintosh Quadra 630 computers require an external video monitor. The computers can work with several sizes of video monitors.

Table 3-9 shows the monitor types supported and the maximum pixel depths available. The pixel depth determines the maximum number of colors that can be displayed. The maximum pixel depth available depends on the size of the monitor's screen.

Table 3-9 Maximum pixel depths for video monitors

Monitor type	Display size, in pixels	Maximum pixel depth, in bits per pixel	Maximum number of colors displayed
12-inch color	512 by 384	16	32,768
14-inch color	640 by 480	16	32,768
15-inch multiscan	800 by 600	8	256
VGA	640 by 480	8	256
SVGA	800 by 600	8	256

For more information about the video monitors, see "Video Timing Parameters" on page 35.

Video Connector

The cable from the video monitor plugs into the computer's DB-15 external video connector; Table 3-10 shows the pin assignments.

Table 3-10 Pin assignments on the external video connector

Pin number	Signal name	Description
1	RED.GND	Red video ground
2	RED.VID	Red video signal
3	/CSYNC	Composite synchronization signal
4	SENSE0	Monitor sense signal 0
5	GRN.VID	Green video signal
6	GRN.GND	Green video ground

continued

Table 3-10 Pin assignments on the external video connector (continued)

Pin number	Signal name	Description
7	SENSE1	Monitor sense signal 1
8	n.c.	Not connected
9	BLU.VID	Blue video signal
10	SENSE2	Monitor sense signal 2
11	GND	CSYNC and VSYNC ground
12	/VSYNC	Vertical synchronization signal
13	BLU.GND	Blue video ground
14	HSYNC.GND	HSYNC ground
15	/HSYNC	Horizontal synchronization signal
Shell	SGND	Shield ground

Note
The video connector on the Macintosh LC 630 and Macintosh Quadra 630 computers is the same as the one on the Macintosh LC III. ◆

Monitor Sense Codes

To identify the type of monitor connected, the Macintosh LC 630 and Macintosh Quadra 630 computers use the Apple monitor sense codes and the extended sense codes. Table 3-11 shows the sense codes for each of the monitors these computers can support. Refer to the Macintosh Technical Note *M.HW.SenseLines* for a description of the sense code system.

Table 3-11 Monitor sense codes

	Standard sense code	Extended sense code		
Monitor type	(SENSE2–0)	(1,2)	(0,2)	(0,1)
12-inch RGB	0 1 0	—	—	—
14-inch RGB	1 1 0	—	—	—
VGA and SVGA	1 1 1	1 1	1 0	1 0
16-inch RGB	1 1 1	1 0	1 1	0 1
No monitor	1 1 1	1 1	1 1	1 1

Note
Both VGA and SVGA monitors have the same sense code. The first time the user starts up with an SVGA monitor, the computer treats it as a VGA monitor and shows a 640-by-480-pixel display. The user can switch to the 800-by-600-pixel SVGA mode from the Monitors control panel; when that happens, the computer changes the display to the 800-by-600-pixel display mode immediately, and continues to use that mode the next time it is started up. ◆

Video Timing Parameters

The Macintosh LC 630 and Macintosh Quadra 630 computers require an external video monitor. The monitor can be one of several different types and display sizes, as listed in Table 3-9.

Table 3-12 Monitors supported

Monitor type	Display size (pixels)
12-inch color	512 by 384
12-inch monochrome	640 by 480
14-inch color	640 by 480
15-inch multiscan	800 by 600*
NTSC or VGA	640 by 480
SVGA	800 by 600

* The 15-inch multiscan monitor can also operate as a 640-by-480-pixel display.

Figure 3-3 shows simplified timing diagrams and identifies the horizontal and vertical timing parameters in a video signal. Table 3-13 and Table 3-13 list the values of those parameters for the different types of monitors.

Figure 3-3 Video timing diagram

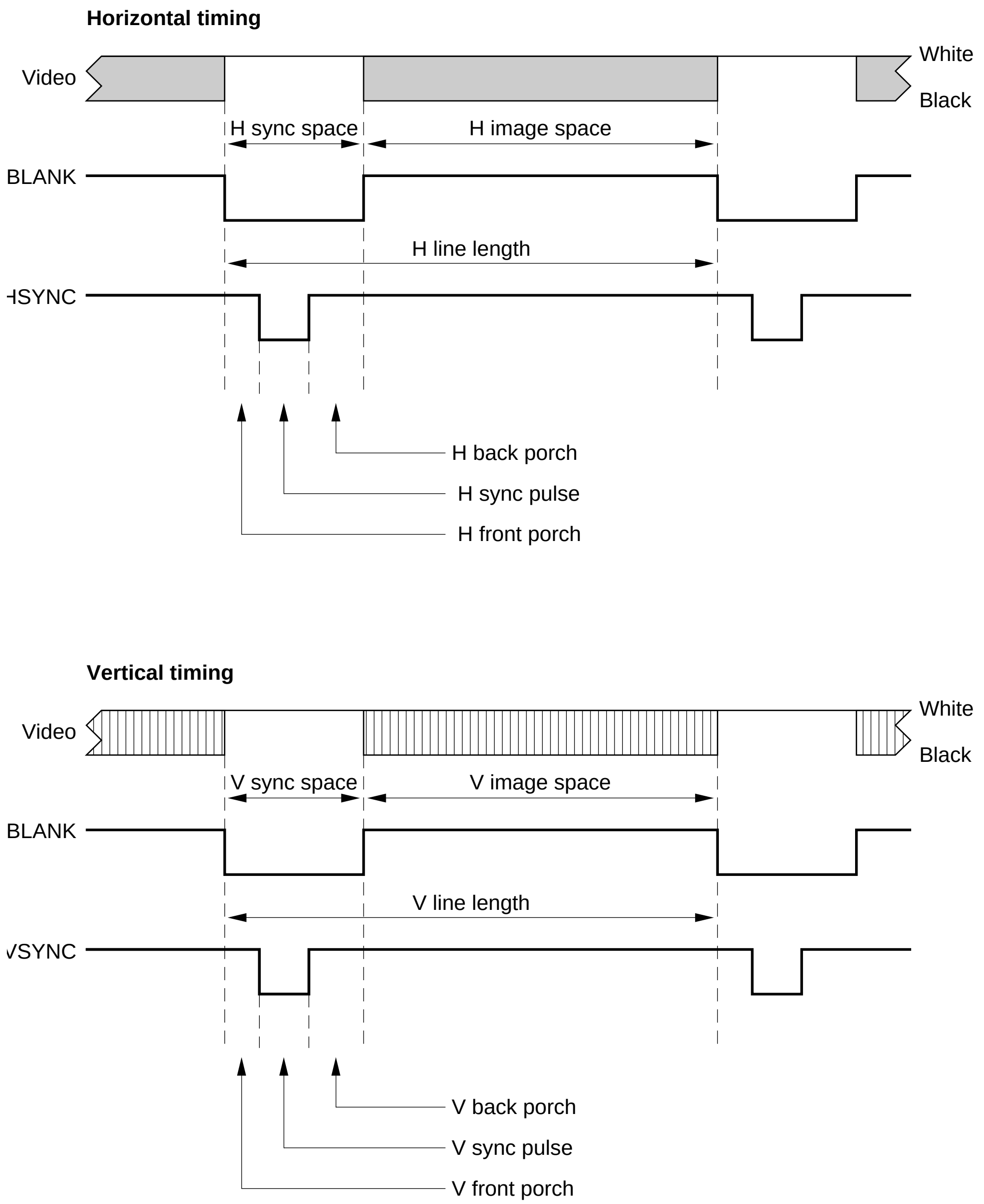


Table 3-13 lists the timing parameters for the smaller monitors listed: the 12-inch color monitor, the 14-inch color monitor, and a standard VGA monitor.

Table 3-13 Video timing parameters for smaller monitors

	Monitor type and dimensions		
Parameter	12-inch color (512 by 384)	14-inch color (640 by 480)	VGA (640 by 480)
Dot clock	15.67 MHz	30.24 MHz	25.18 MHz
Dot time	63.83 ns	33.07 ns	39.72 ns
Line rate	24.48 kHz	35.00 kHz	31.47 kHz
Line time	40.85 µs (640 dots)	28.57 µs (864 dots)	31.78 µs (800 dots)
Horizontal active video	512 dots	640 dots	640 dots
Horizontal blanking	128 dots	224 dots	160 dots
Horizontal front porch	16 dots	64 dots	16 dots
Horizontal sync pulse	32 dots	64 dots	96 dots
Horizontal back porch	80 dots	96 dots	48 dots
Frame rate	60.15 Hz	66.72 Hz	59.94 Hz
Frame time	16.63 ms (407 lines)	15.01 ms (525 lines)	16.68 ms (525 lines)
Vertical active video	384 lines	480 lines	480 lines
Vertical blanking	23 lines	45 lines	45 lines
Vertical front porch	1 line	3 lines	10 lines
Vertical sync pulse	3 lines	3 lines	2 lines
Vertical back porch	19 lines	39 lines	33 lines

Table 3-13 lists the timing parameters for the SVGA monitor at 60 and 72 frames per second.

Table 3-14 Video timing parameters for larger monitors

	Monitor type and dimensions	
Parameter	SVGA (800 by 600 at 60 fields/sec)	SVGA (800 by 600 at 72 fields/sec)
Dot clock	40.00 MHz	50.00 MHz
Dot time	25.00 ns	20.00 ns
Line rate	37.88 kHz	48.08 kHz
Line time	26.4 µs (1056 dots)	20.80 µs (1040 dots)
Horizontal active video	800 dots	800 dots
Horizontal blanking	256 dots	240 dots
Horizontal front porch	40 dots	56 dots
Horizontal sync pulse	128 dots	120 dots
Horizontal back porch	88 dots	64 dots
Frame rate	60.31 Hz	72.18 Hz
Frame time	16.58 ms (628 lines)	13.85 ms (666 lines)
Vertical active video	600 lines	600 lines
Vertical blanking	28 lines	66 lines
Vertical front porch	1 line	37 lines
Vertical sync pulse	4 lines	6 lines
Vertical back porch	23 lines	23 lines

CHAPTER 4

Expansion Features

This chapter describes the expansion features of the Macintosh LC 630 and Macintosh Quadra 630 computers: the RAM expansion slot, the I/O expansion slot, and the DVA connector on the video input module.

Note
Apple does not support development of third-party cards for the video input slot. ◆

RAM Expansion

The Macintosh LC 630 and Macintosh Quadra 630 computers come with 4 MB or 8 MB of system RAM on the main logic board. The user can expand the RAM up to a maximum of 36 MB by plugging in a 72-pin SIMM.

Note
The video display buffer uses separate on-board DRAM. The display DRAM cannot be expanded. ◆

RAM Configurations

Figure 4-1 shows the RAM configurations for different amounts of RAM. For more information, see the section "RAM Addresses" on page 18.

Signals on the RAM SIMM Slot

Table 4-1 gives the signal assignments for the pins of the RAM SIMM slot.

IMPORTANT
RAM SIMMs used in Macintosh computers must meet the timing and electrical standards of those machines. SIMMs designed for other computers may not work. ▲

Figure 4-1 RAM configurations

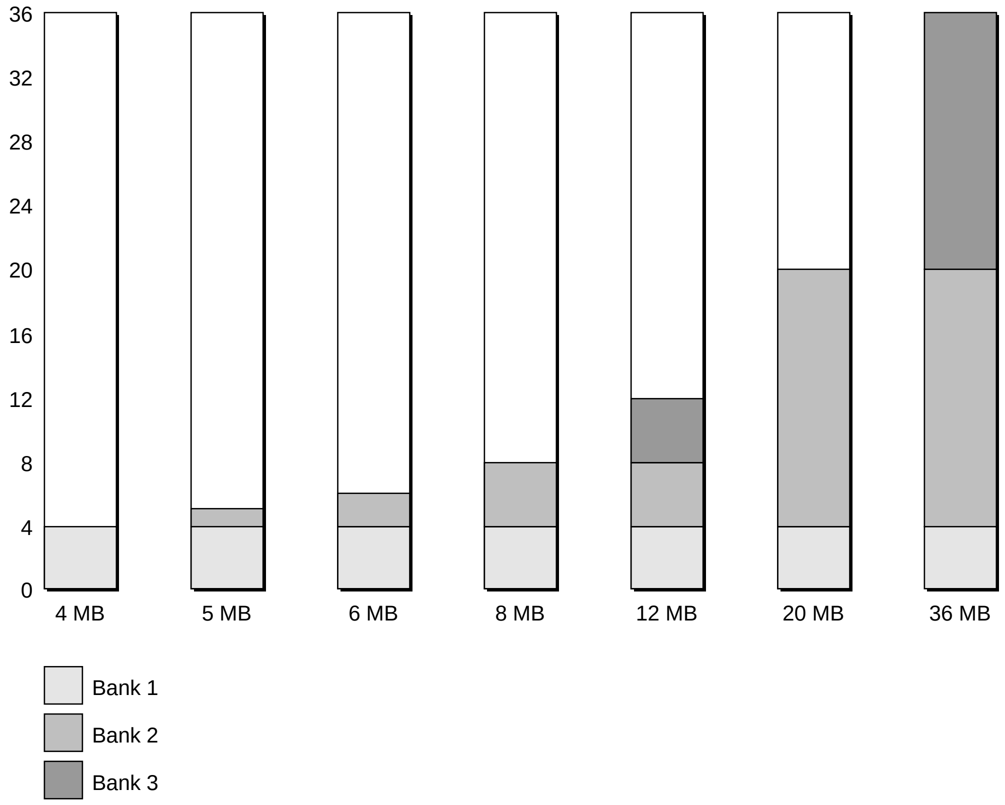


Table 4-1 Signal assignments on the RAM SIMM slot

Pin	Signal name	Description
1	GND	Ground
2	DQ0	Data input/output bus, bit 0
3	DQ16	Data input/output bus, bit 16
4	DQ1	Data input/output bus, bit 1
5	DQ17	Data input/output bus, bit 17
6	DQ2	Data input/output bus, bit 2
7	DQ18	Data input/output bus, bit 18
8	DQ3	Data input/output bus, bit 3
9	DQ19	Data input/output bus, bit 19
10	+5V	+5 volts

continued

Table 4-1 Signal assignments on the RAM SIMM slot (continued)

Pin	Signal name	Description
11	n.c.	Not connected
12	A0	Address bus, bit 0
13	A1	Address bus, bit 1
14	A2	Address bus, bit 2
15	A3	Address bus, bit 3
16	A4	Address bus, bit 4
17	A5	Address bus, bit 5
18	A6	Address bus, bit 6
19	A10	Address bus, bit 10
20	DQ4	Data input/output bus, bit 4
21	DQ20	Data input/output bus, bit 20
22	DQ5	Data input/output bus, bit 5
23	DQ21	Data input/output bus, bit 21
24	DQ6	Data input/output bus, bit 6
25	DQ22	Data input/output bus, bit 22
26	DQ7	Data input/output bus, bit 7
27	DQ23	Data input/output bus, bit 23
28	A7	Address bus, bit 7
29	A11	Address bus, bit 11
30	+5V	+5 volts
31	A8	Address bus, bit 8
32	A9	Address bus, bit 9
33	/RAS3	Row address strobe 3
34	/RAS2	Row address strobe 2
35	—	Reserved
36	—	Reserved
37	—	Reserved
38	—	Reserved
39	GND	Ground
40	/CAS0	Column address strobe 0
41	/CAS2	Column address strobe 2

continued

Table 4-1 Signal assignments on the RAM SIMM slot (continued)

Pin	Signal name	Description
42	/CAS3	Column address strobe 3
43	/CAS1	Column address strobe 1
44	/RAS0	Row address strobe 0
45	/RAS1	Row address strobe 1
46	n.c.	Not connected
47	/W	Write enable
48	n.c.	Not connected
49	DQ8	Data input/output bus, bit 8
50	DQ24	Data input/output bus, bit 24
51	DQ9	Data input/output bus, bit 9
52	DQ25	Data input/output bus, bit 25
53	DQ10	Data input/output bus, bit 10
54	DQ26	Data input/output bus, bit 26
55	DQ11	Data input/output bus, bit 11
56	DQ27	Data input/output bus, bit 27
57	DQ12	Data input/output bus, bit 12
58	DQ28	Data input/output bus, bit 28
59	+5V	+5 volts
60	DQ29	Data input/output bus, bit 29
61	DQ13	Data input/output bus, bit 13
62	DQ30	Data input/output bus, bit 30
63	DQ14	Data input/output bus, bit 14
64	DQ31	Data input/output bus, bit 31
65	DQ15	Data input/output bus, bit 15
66	n.c.	Not connected
67	—	Reserved
68	—	Reserved
69	—	Reserved
70	—	Reserved
71	n.c.	Not connected
72	GND	Ground

RAM Devices

The RAM controller in the F108 IC supports 1 MB, 4 MB, and 16 MB devices; it does not support 64 MB devices.

A RAM SIMM can support either one or two banks. A one-bank SIMM using 1 Mbit, 4 Mbit, or 16 Mbit devices provides RAM expansion of 1 MB, 4 MB, or 16 MB, respectively. A two-bank SIMM using the same devices provides 2 MB, 8 MB, or 32 MB.

IMPORTANT
You should not use 1-bit-wide DRAM devices in a RAM SIMM for the Macintosh LC 630 and Macintosh Quadra 630 computers because the required number of devices adds excessive capacitive loading to the address and control buses. ▲

The access time of the DRAM devices must be 80 ns or less. The RAM controller in the F108 IC performs the refresh function, using CAS before RAS refresh and refreshing the DRAM devices within 15.6 µs. DRAM devices that require refreshing within 7.8 µs are not supported.

The RAM controller in the F108 IC supports line burst transfers. The number of cycles for system RAM accesses are:

- single read 6
- single write 5
- burst read 6-3-3-3
- burst write 5-2-2-2

The RAM controller does not support interleaved accesses.

Addressing RAM

Signals A[0–11] make up a 12-bit multiplexed address bus that can support several different types of DRAM devices.

Depending on their internal design and size, different types of DRAM devices require different row and column address multiplexing. The F108 custom IC provides for two addressing modes, selected individually for each bank of DRAM. The system software initializes the address mode bits as part of the process of determining the amount of RAM installed in the computer.

Table 4-2 shows how the signals are multiplexed during the row and column address phases for each of the addressing modes.

Table 4-2 Address multiplexing for DRAM devices

	Individual signals on DRAM_ADDR bus											
	A[11]	A[10]	A[9]	A[8]	A[7]	A[6]	A[5]	A[4]	A[3]	A[2]	A[1]	A[0]
Address mode = 1												
Row address bits	A23	A22	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11
Column address bits	A24	A23	A21	A10	A9	A8	A7	A6	A5	A4	A3	A2
Address mode = 0												
Row address bits	A21	A20	A10	A19	A18	A17	A16	A15	A14	A13	A12	A11
Column address bits	A24	A23	A25	A22	A9	A8	A7	A6	A5	A4	A3	A2

Table 4-3 shows the address modes used with several types of DRAM devices. The devices are characterized by their bit dimensions: for example, a 256K by 4-bit device has 256 addresses and stores 4 bits at a time.

Table 4-3 Address modes for various DRAM devices

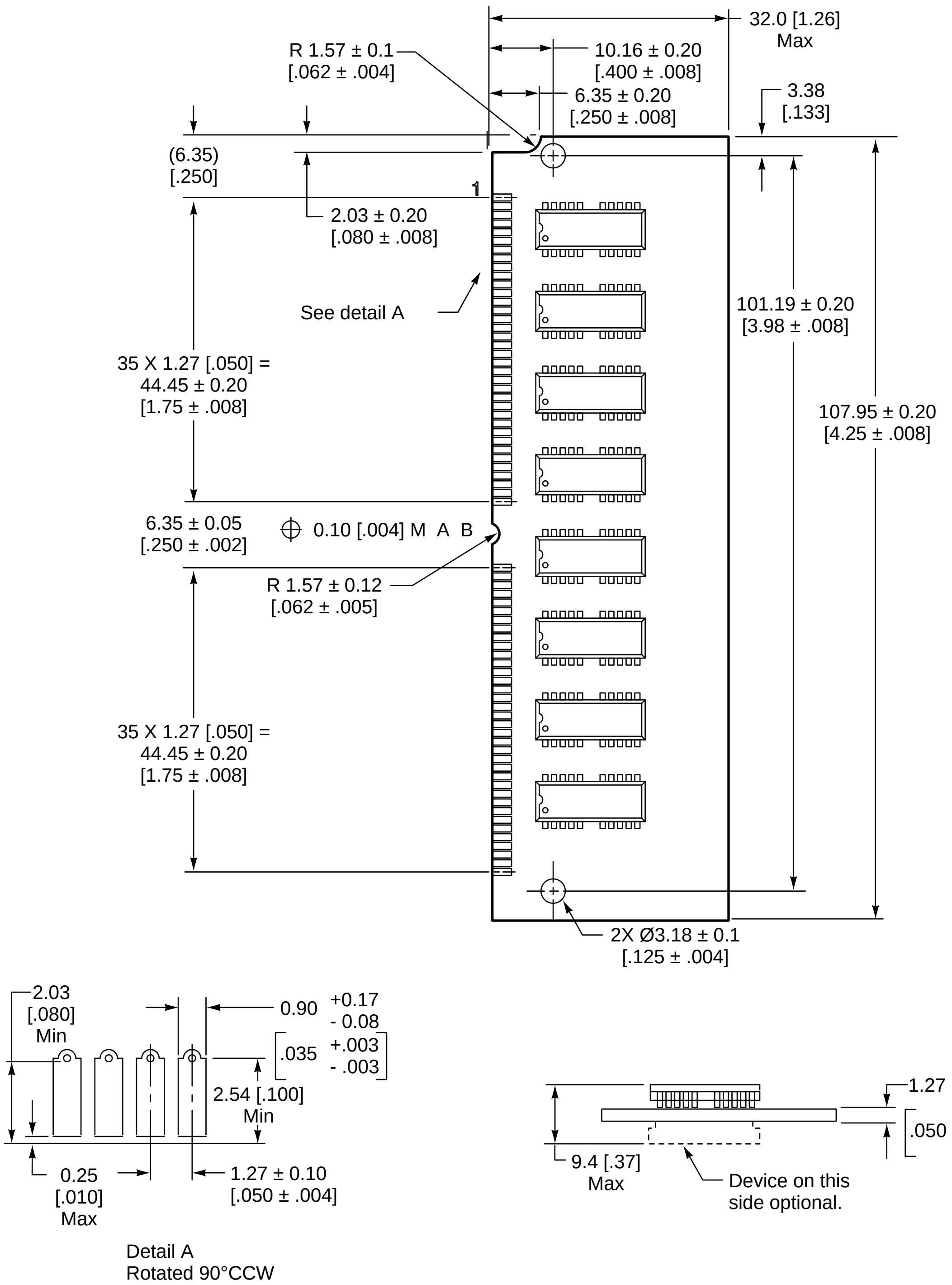
Device size	Device type	Row bits	Column bits	Address mode
1 megabit	1 M by 1	10	10	1
1 megabit	256K by 4	9	9	1
4 megabits	4 M by 1	11	11	1
4 megabits	1 M by 4	10	10	1
4 megabits	512K by 8	10	9	1
4 megabits	256K by 16	10	8	0
16 megabits	16 M by 1	12	12	0
16 megabits	4 M by 4	11	11	1
16 megabits	4 M by 4	12	10	1
16 megabits	2 M by 8	11	10	1
16 megabits	2 M by 8	12	9	0
16 megabits	1 M by 16	12	8	0

RAM SIMM Mechanical Specifications

The RAM SIMM for the Macintosh LC 630 and Macintosh Quadra 630 computers is mechanically the same as the 72-pin RAM SIMMs used in the Macintosh LC III, Macintosh LC 475, and Macintosh Quadra 605 and 610 computers. The mechanical design of the 72-pin RAM SIMM is based on the industry standard design defined in the JEDEC Standard Number 21-C.

Figure 4-2 shows the mechanical specifications of the RAM SIMM. Pin contacts must be tin, not gold or copper, and the circuit board must dedicate one layer to power and one to ground.

Figure 4-2 RAM expansion SIMM



Note: Dimensions are in millimeters with inches in brackets.

The I/O Expansion Slot

The I/O expansion slot in the Macintosh LC 630 and Macintosh Quadra 630 computers can accept either of two types of expansion cards: a 96-pin card similar to the PDS card used in the Macintosh LC II or a 114-pin card similar to the PDS card used in the Macintosh LC III.

IMPORTANT
The I/O expansion slot in the Macintosh LC 630 and Macintosh Quadra 630 computers is not a PDS (processor-direct slot) because it is not connected directly to the main processor. PDS cards designed to interact with the main processor—to provide, for example, a RAM cache or an FPU—will not work in the I/O expansion slot. ▲

The I/O Expansion Connector

The I/O expansion connector in the Macintosh LC 630 and Macintosh Quadra 630 computers is mechanically the same as the PDS connector in the Macintosh LC III. It is essentially a 120-pin Euro-DIN connector with six pins removed to make a notch. The notch divides the connector into two sections: a 96-pin section that accepts the 96-pin connector used on PDS cards for the Macintosh LC II, and a separate 18-pin section for additional signals. For more information see the section “Card Connectors” on page 54.

Connector Pin Assignments

Table 4-4 gives the pin assignments for the I/O expansion connector. Pins 1 through 32 in all three rows (A, B, and C) correspond to the 96-pin section of the connector. Pins 33 and 34 in all three rows are missing—those pins correspond to the notch in the connector. Pins 35 through 40 in all three rows make up the 18-pin section of the connector.

Except for one signal, 16MASTER (on pin B31 and described in Table 4-5 on page 51), the pin assignments on the 96-pin section of the extended PDS are the same as those on the PDS in the Macintosh LC II. On the Macintosh LC II, pin B31 is the Apple II video clock input.

Note
Signal names starting with a slash (/) are active when their signal lines are driven to a logical zero (0). ◆

IMPORTANT
Under no circumstances should you use the Analog GND pin (Pin 1, Row B) for a digital ground on your expansion card. Doing so will cause digital noise to be coupled into the audio system, resulting in degraded sound quality. ▲

Table 4-4 Pin assignments for the expansion connector

Pin number	Row A	Row B	Row C
1	n. c.	Analog GND	/FPU.SEL
2	/SLOTIRQ	/R/W	/DS
3	/PDS.AS	+5V	/BERR
4	/PDS.DSACK1	+5V	/PDS.DSACK0
5	/HALT	SIZ1	SIZ0
6	FC2	GND	FC1
7	FC0	CLK16M	/RESET
8	/RMC	GND	/BG.SLOT
9	D31	D30	D29
10	D28	D27	D26
11	D25	D24	D23
12	D22	D21	D20
13	D19	D18	D17
14	D16	D15	D14
15	D13	D12	D11
16	D10	D9	D8
17	/BGACK	/BR_SLOT	A0
18	A1	A31	A27
19	A26	A25	A24
20	A23	A22	A21
21	A20	/IPL2	/IPL1
22	/IPL0	D3	D4
23	D2	D5	D6
24	D1	D0	D7
25	A4	A2	A3
26	A6	A12	A5
27	A11	A13	A7
28	A9	A8	A10
29	A16	A15	A14
30	A18	A17	A19

continued

Table 4-4 Pin assignments for the expansion connector (continued)

Pin number	Row A	Row B	Row C
31	n. c.	16MASTER	FC3
32	+12V	GND	–5V
33	(pin not present)	(pin not present)	(pin not present)
34	(pin not present)	(pin not present)	(pin not present)
35	A28	/BG.SLOT	C16M
36	A29	+5V	A30
37	/CIOUT	/CPU.AS	/STERM
38	/CBACK	n. c.	/CBREQ
39	n. c.	/CPU.DSACK0	n. c.
40	n. c.	GND	/CPU.DSACK1

All the signals on the expansion connector are capable of driving at least one TTL load (1.6 mA sink, 400 µA source). Most of the signals are connected to other MOS devices on the main logic board; for those signals, the DC load on the bus signals is small. All the data lines (D0–D31) are connected to the PrimeTime II custom IC so they have CMOS-type loads.

Signal Descriptions

The I/O expansion slot in the Macintosh LC 630 and Macintosh Quadra 630 computers is intended to be compatible with expansion cards designed for computers that use the MC68030 microprocessor (the Macintosh LC III and Macintosh LC 520 computers). Because the bus protocols of the MC68040 microprocessor are not the same as those of the MC68030, many of the signals on the I/O expansion slot are not connected directly to the main processor . Instead, those signals are connected to the PrimeTime II custom IC, which emulates the MC68030 control and data buses.

The upper 30 address lines (A31–2) are connected directly to the MC68040 microprocessor. The I/O bus adapter logic in the PrimeTime II IC provides the buffered data bus (IOD31–0) and the two lowest address lines (A1–0).

Table 4-5 describes the signals on the I/O expansion connector.

Table 4-5 Descriptions of the signals on the I/O expansion connector

Signal name	Signal description
A0–A31	Address lines.
/BERR	Bus error; bidirectional signal indicating that an error occurred during the current bus cycle; when /HALT is also asserted, /BERR causes the bus cycle to be retried.
/BGACK	Bus grant acknowledge; input signal indicating that a device on the card has become bus master.
/BG.SLOT	Bus grant to the slot; signal that a device on the card can become bus master following completion of current processor bus cycle (when /PDS.AS, /BGACK, and all the /DSACK signals are inactive).
/BR.SLOT	Bus request from the slot; input signal indicating that a device on the card is requesting to become bus master.
/CBACK	CPU burst acknowledge; used with /STERM during a burst transfer to indicate that an individual element of a burst transfer is ready.
/CBREQ	CPU burst request; used to initiate a quadruple longword burst transfer; tied to a 4.7K pull-up resistor.
/CIOUT	Cache inhibit out signal from main processor, indicating that a second-level cache is allowed to participate in the current bus transaction; tied to a 300 Ω pull-down resistor.
C16M	Same signal as CLK16M.
CLK16M	Independent clock running at 15.6672 MHz; provided for compatibility with Macintosh LC and LC II PDS cards.
/CPU.AS	Address strobe; three-state signal indicating that an active bus transaction is occurring.
/CPU.DSACK0, /CPU.DSACK1	Data strobe acknowledge signals; asserted by the addressed bus slave to end a bus transaction; also used to inform the master of the size of the slave's data port. These signals are electrically connected to the corresponding /PDS.DSACK signals.
D0–D31	Data lines.
/DS	Data strobe. During a read operation, /DS is asserted when a device on the card should place data on the data bus; during a write operation, /DS is asserted when the main processor has put valid data on the data bus.
FC0–FC2	Function code used to identify address space of current bus cycle; tied to pull-up and pull-down resistors to indicate supervisor data space accesses.
FC3	Additional function code bit, used to indicate that the software is running in 32-bit address mode. (As in the Macintosh LC II, the software always runs in 32-bit mode.)

continued

Table 4-5 Descriptions of the signals on the I/O expansion connector (continued)

Signal name	Signal description
/FPU.SEL	Select signal for an optional MC68881 or MC68882 FPU; tied to a 4.7K pull-up resistor; never asserted by the logic board in a Macintosh LC 475 or Macintosh Quadra 605 computer.
/HALT	Used in conjunction with the /BERR signal to terminate a bus cycle with a retry response; not used to stop processor execution.
/IPL0–IPL2	Interrupt priority-level lines.
/PDS.AS	Address strobe; synchronized to 16 MHz regardless of the actual processor speed; asserted only when a valid slot address is being generated by the bus master. When the card is the active bus master, the card may drive either this signal or /CPU.AS, but not both.
/PDS.DSACK0, /PDS.DSACK1	Data strobe acknowledge signals; asserted by the addressed bus slave to end a bus transaction; also used to inform the master of the size of the slave's data port. These signals are electrically connected to the corresponding /CPU.DSACK signals.
/RESET	Bidirectional signal that initiates system reset.
/RMC	Three-state output signal that identifies the current bus cycle as part of an indivisible bus cycle such as a read-modify-write operation.
/R/W	Read/write; three-state output signal that defines the direction of the bus transfer with respect to the current bus master; logical one (1) indicates a bus-master read, zero (0) indicates bus-master write.
16MASTER	Indicates the width of the data port when the card is alternate bus master. A logical one (1) indicates a 16-bit port; logical zero (0) indicates a 32-bit port. The signal is pulled high on the main logic board.
SIZ0–SIZ1	Three-state output signals that work in conjunction with the PrimeTime II IC's dynamic bus sizing capabilities and indicate the number of bytes remaining to be transferred during the current bus cycle.
/SLOTIRQ	Interrupt request line from the card; reported to the system by way of the SLOT.E request; when low, generates a level-2 interrupt if the slot interrupt enable bit is set.
/STERM	Indicates termination of a synchronous transfer by a card using the MC68030 synchronous cycle.

The /BG.SLOT signal appears on two pins; there is no separate CPU.BG signal. The following signals on the expansion slot are permanently connected:

- /PDS.DSACK0 is connected to /CPU.DSACK0
- /PDS.DSACK1 is connected to /CPU.DSACK1

Unlike those signals, the /PDS.AS signal and the /CPU.AS signal are not connected together. The /PDS.AS signal is used only for addresses in the slot $E address range; the

/CPU.AS signal is used for addresses in expansion slot and Super Slot spaces $6–$8, $A–$D, and $F (the slot $9 address spaces are used for built-in video circuitry).

IMPORTANT
The I/O expansion slot does not support MC68040 bus transfers. The expansion slot does not support a processor operating at a clock frequency other than 16 MHz. ▲

Bus Master on a Card

The I/O expansion slot will support a card with an MC68020 or MC68030 bus master. The PrimeTime II custom IC controls bus arbitration between the card's bus master and the MC68040 microprocessor so that either bus master will eventually obtain the bus. The MC68020 or MC68030 will obtain the I/O data bus and the address bus. The MC68040 will obtain the processor data bus and the address bus. Because there is only one address bus, there can be only one bus master at a time.

Asynchronous transfers are the preferred method for data transfers to and from an I/O expansion card. When an I/O expansion card contains an active bus master, the PrimeTime II IC terminates successful data transfers using the DSACK signals. A slave on the expansion card can also terminate a transfer using DSACK signals.

The PrimeTime II IC can never be a synchronous slave on the I/O bus, so PrimeTime II cannot terminate data tansfers as a slave using /STERM. On the other hand, a bus slave on an expansion card can terminate a 32-bit wide synchronous transfer using /STERM. PrimeTime II supports /STERM terminations as a master on the I/O bus, and all transfers from PrimeTime II to the expansion slot are based on the 16 MHz clock.

Incompatibility With Older Cards

While the I/O expansion slot will accept PDS cards designed for the Macintosh LC II and LC III, some of those cards do not work. Cards that are incompatible with the expansion slot include

- cards designed to work as coprocessors with an MC68020 or an MC68030 or as replacements for those microprocessors. Such cards include accelerators, 68882 FPU cards, and cache cards. That type of card won't work because the microprocessor is different and because the slot signals are not connected directly to the microprocessor.
- cards with drivers that include incompatible code. Some drivers that do not follow Apple Computer's programming guidelines won't work on machines that use the MC68040 microprocessor. For example, some of those drivers write directly to the cache control register in an MC68030. Such code won't work on an MC68040.
- cards with drivers that include code to check the `gestaltMachineType` value and refuse to run on a newer CPU. The idea is to protect users by refusing to run on a machine that the cards haven't been tested on. Such cards have compatibility problems with all new Macintosh models.

Designing an I/O Expansion Card

The I/O expansion card for the Macintosh LC 630 and Macintosh Quadra 630 computers is approximately 3 inches wide by 5 inches long. It fits parallel to the main logic board and reaches to an opening in the back of the case (normally filled by a snap-out cover). The opening provides access to a 15-pin D-type connector on the card for external I/O. For mechanical specifications of the I/O expansion card, see Appendix A.

The appendix "Foldout Drawings" contains drawings showing the recommended mechanical design guidelines for the expansion card. Foldout 1 shows the maximum dimensions of the expansion card and the location of the expansion connector. Foldout 2 provides component height restrictions for the expansion card. Foldout 3 shows how the card is installed on the main logic board.

Note
The I/O expansion card is the same size and shape as the PDS card for the Macintosh LC III computer. ◆

Card Connectors

The custom 114-pin PDS connector on the computer's main logic board accepts either a 96-pin or 120-pin standard Euro-DIN connector. You can order connectors meeting Apple specifications from Amp Incorporated, Harrisburg, PA 17105 or from Augat Incorporated, Interconnect Products Division, P. O. Box 779, Attleboro, MA 02703. Refer to *Designing Cards and Drivers for the Macintosh Family*, third edition, for more information about those connectors.

Power for the Card

The maximum current available at each supply voltage is shown in Table 4-6. The card must not dissipate more than 4 W total; for example, if the card uses the maximum current at –5 V and +12 V, it must not use more than 300 mA from the +5 V supply.

Table 4-6 Power available for the expansion card

Voltage	Current
+5	800 mA
–5	20 mA
+12	200 mA

▲ **WARNING**
Cards dissipating more than 4 watts may overheat and damage the computer's circuitry or cause it to become inoperable. ▲

Card Address Space

The address space for the I/O expansion card appears in physical address spaces \$E000 0000–\$EFFF FFFF and \$FE00 0000–\$FEFF FFFF. To match the conventions used by the Slot Manager, software should address the card as if it were in slot space \$E: either the 16 MB slot space \$FE00 0000–\$FEFF FFFF or the super slot space \$E000 0000–\$EFFF FFFF.

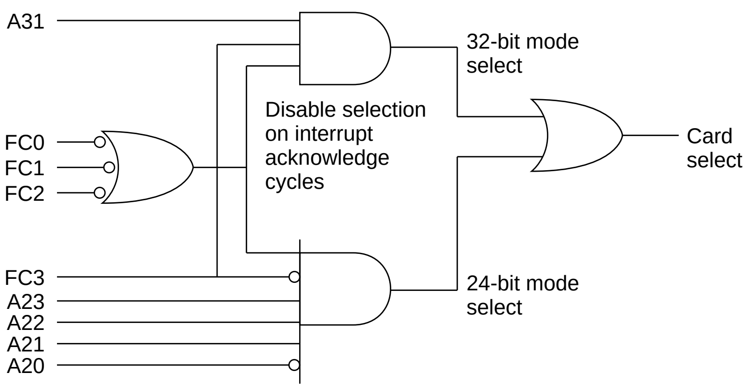
Card Select Signal

The I/O expansion card must generate its own select signal from the address and function code signals on the connector. The card select signal must be disabled when FC0, FC1, and FC2 are all active; that condition corresponds to a function code of 111 (CPU space). Figure 4-3 shows a typical logic circuit for generating the card select signal.

IMPORTANT

To ensure compatibility with future hardware and software, you should minimize the chance of address conflicts by decoding all the address bits. To ensure that the Slot Manager recognizes your card, the card's declaration ROM must reside at the upper address limit of the 16 MB address space (\$FE00 0000–\$FEFF FFFF). ▲

Figure 4-3 Generating the card select signal



The DVA Connector

The video input module has a separate connector called the DVA (digital video application) connector. The DVA connector provides access to the video input modules's 4:2:2 unscaled YUV video input data bus and associated control signals. By means of a cable to the DVA connector, an I/O expansion card can gain access to the digital video bus on the video input card and use it to transfer real-time video data to the computer.

Such an I/O expansion card could contain a hardware video compressor or other video processor.

The DVA connector is a 34-pin flat ribbon connector located at the top edge of the video input module. Figure 4-4 is a view of the main logic board showing the I/O expansion card and the location of the DVA connector on the video input module.

Figure 4-4 Location of the DVA connector

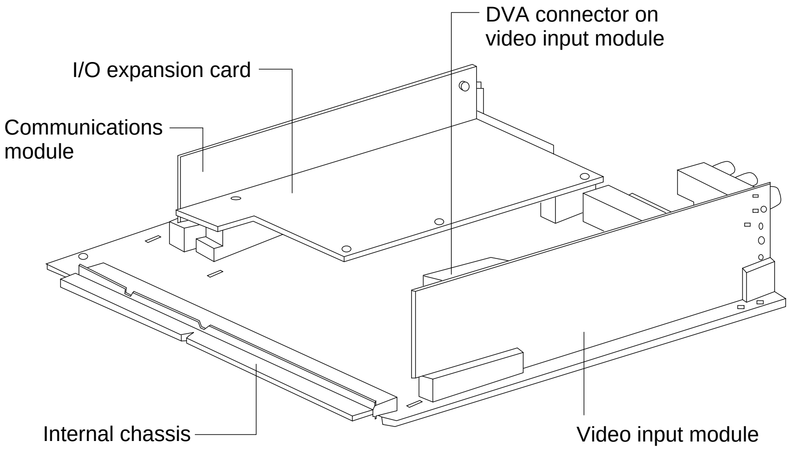
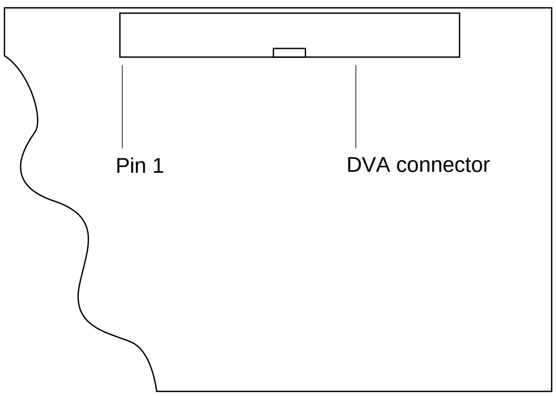


Figure 4-5 shows the orientation of the DVA connector on the video input module.

Figure 4-5 Orientation of the DVA connector



The DVA connector accepts YUV video and analog sound from the expansion card but does not itself generate YUV video output or audio output signals.

IMPORTANT

The DVA connector on the video input module in the Show provides some of the functionality of the DAV connectors found on the Power Macintosh models and the Macintosh Quadra AV models, but it is not compatible with either of those connectors. Refer to *Macintosh DAV Interface for NuBus Expansion Cards* in *Developer Note Number 8* for more information. ▲

Pin Assignments

Table 4-7 shows the pin assignments on the DVA connector.

Table 4-7 Pin assignments on the DVA connector

Pin number	Signal name	Pin number	Signal name
1	Y(7)	2	Y(6)
3	Y(5)	4	Y(4)
5	Y(3)	6	Y(2)
7	Y(1)	8	Y(0)
9	UV(7)	10	UV(6)
11	UV(5)	12	UV(4)
13	UV(3)	14	UV(2)
15	UV(1)	16	UV(0)
17	Ground	18	LLCB
19	Ground	20	CREFB
21	Ground	22	VS
23	Ground	24	HS
25	Ground	26	HREF
27	Ground	28	DIR
29	Reserved	30	Reserved
31	Ground	32	YUV_SND_LEFT
33	YUV_RET	34	YUV_SND_RIGHT

Signal Descriptions

Table 4-5 gives descriptions of the signals on the DVA connector.

Table 4-8 Descriptions of the signals on the DVA connector

Signal name	Signal description
CREFB	Clock reference signal
DIR	YUV directional signal
HREF	Horizontal reference signal
HS	Horizontal sync signal
LLCB	Line-locked clock signal
UV(0–7)	Digital chrominance data bus
VS	Vertical sync signal
Y(0–7)	Digital luminance data bus

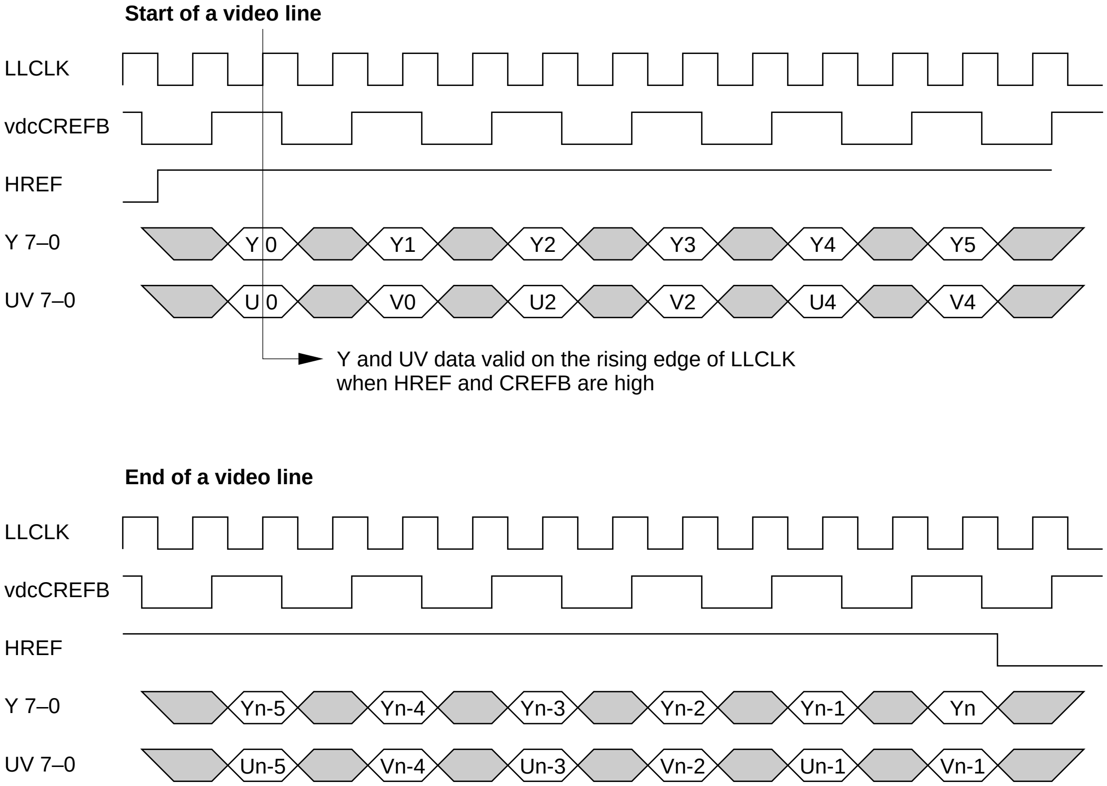
Using the YUV Bus

The video input module contains a digital video decoder and scaler (DESC) , the Philips SAA7196 IC. Logic on the video input card uses the CVBS port on the DESC and pulls the DIR signal low, disabling the YUV bus. For an expansion card to use the YUV bus, the software associated with the card must set the DIR signal high so that the DESC will accept data on the YUV bus. To do that, the software can use the Cuda Dispatch Manager to issue an IIC command to write to register $E of the DESC. For information about using the registers in the DESC IC, please refer to the *SAA7194/6 Philips Desktop Video Handbook.*

Video Data Format

Digital video data is transmitted as lines and fields. Each line consists of an even number of samples on the Y and UV buses as shown in Figure 4-6. HREF is high during a video line and low during the horizontal blanking interval. The falling edge of the VS signal indicates the beginning of a video field. For more information about digital video data in YUV format, see *Macintosh DAV Interface for NuBus Expansion Cards* in *Developer Note Number 8.*

Figure 4-6 Video data timing



C H A P T E R 5

Software Features

The first part of this chapter describes the software in the ROM of the Macintosh LC 630 and Macintosh Quadra 630 computers. The second part describes the system software that supports the new features of those computers.

For a description of the system software for the internal IDE hard disk, see Chapter 6, "Software for the IDE Hard Disk."

ROM Software

The ROM in the Macintosh LC 630 and Macintosh Quadra 630 computers is based on the ROM for the Macintosh Quadra 610 and 650 models with the necessary changes to support machine-specific hardware.

The sections that follow describe the following changes in the ROM:

- machine identification
- new memory map
- new video controller
- push button interrupts
- ADB and soft power switch
- power saver

Machine Identification

The ROM includes new tables and code for identifying the machine.

Applications can find out which computer they are running on by using the Gestalt Manager routines; see *Inside Macintosh: Overview*. The `gestaltMachineType` value is 98 (hexadecimal $62).

New Memory Map

The ROM code has been modified to support the memory addressing used by the Macintosh LC 630 and Macintosh Quadra 630 computers. ROM code determines the size of RAM and sets up the MMU to make the RAM addresses contiguous. The ROM includes descriptions of the memory space needed for setting up the MMU.

The ROM code also creates the physical-space tables the computer needs in order to run virtual memory. The computer uses the 32-bit Memory Manager and runs in 32-bit mode.

New Video Controller

The video frame buffer controller in the Valkyrie custom IC is similar to the DAFB IC used in the Macintosh Quadra models. The video driver code has been changed to accommodate the differences between the Valkyrie IC and the previous IC and to support a new video clock generator IC.

The video driver supports the video displays shown in Table 5-1.

Table 5-1 Maximum pixel depths for video monitors

Monitor type	Display size, in pixels	Pixel depths, in bits per pixel
12-inch color	512 by 384	1, 2, 4, 8, 16
14-inch color	640 by 480	1, 2, 4, 8, 16
15-inch multiscan	800 by 600	1, 2, 4, 8
VGA	640 by 480	1, 2, 4, 8
SVGA	800 by 600	1, 2, 4, 8

Push Button Interrupts

The ROM includes routines for initializing the push button interrupt bits in the interrupt enable and flag registers and for initializing other new registers that support the push buttons.

Pushing one of the two push buttons on the front of the case causes the computer to set a bit in the push button register, which in turn causes a level-2 interrupt. The interrupt handler disables the push button interrupt until the button that caused the interrupt is released.

ADB and Soft Power Switch

The Cuda IC provides the ADB interface and the parameter RAM. The ROM code to support the Cuda IC was originally developed for the Macintosh LC 520 computer.

The Cuda IC has a command and a signal line to support an LED on the front panel. That LED is used to indicate that the power is on.

To reduce system overhead for reading the parameter RAM, the Cuda IC uses a write-through cache. To maintain data integrity, data is written to both the cache and the Cuda. The memory for the cache is located in the global storage for the Cuda Manager.

The ROM software provides the soft power-down capability, shutting down the computer when the user presses one of the Power keys or selects Shut Down in the Special menu of the Finder.

Power Saver

The ROM software includes a screen control driver that reduces power consumption by disabling the video clocks to the monitor whenever the computer is unused for a period of time selected by the user. In addition to turning off the video clocks, the driver also shuts down the internal hard disk drive. When the user moves the mouse or presses a key, the driver turns the video clocks back on and spins up the hard disk drive.

During the time period while the video monitor is warming up, the driver periodically sends a tone to the loudspeaker to let the user know that something is happening.

System Software

The Macintosh LC 630 and Macintosh Quadra 630 computers require System 7.1.2 or a later version of system software. The disk labeled “Install Me First” includes a system enabler file that contains the resources the system needs to start up and initialize the computer.

The system disk includes an installer application to install the control panels for the new features of the machine.

The system software includes the following new features:

- a system enabler for these machines
- a bootable CD-ROM
- Power Saver control panel
- video input digitizing component

The system disk also includes the application that controls the optional TV tuner: Apple Video Player.

System Enabler

Starting with the international release of System 7.1, each reference release of the Macintosh system software supports a new startup extension, the system enabler. A *system enabler* is a software resource that is able to perform the correct startup process for one or more Macintosh computers.

As soon as the system software on disk takes over the startup process, it searches for all system enablers that can start up the particular machine. Each system enabler contains a resource that specifies which computers it is able to start up and the time and date of its creation. If the system software finds more than one enabler for the particular computer, it passes control to the one with the most recent time and date.

In general, the system enabler included in each reference release of system software is able to start up all previous computers. The enablers for computers introduced after a reference release may be independent or may use resources from the previous reference release.

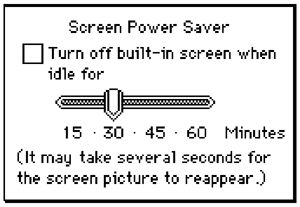
Booting From a CD-ROM

The Macintosh LC 630 and Macintosh Quadra 630 computers can start up (boot) from a built-in CD-ROM drive. Starting up in this fashion is not recommended, because the system software was not designed to operate from a locked storage device—one that the software can't write to. The system software that Apple Computer uses on the system CD-ROM includes only one control panel file—for setting the startup disk—along with installer software to install the system onto a hard disk. Developers may wish to use a similar arrangement to distribute bulky software.

Power Saver Control Panel

The Macintosh LC 630 and Macintosh Quadra 630 computers have a Power Saver control panel with a slider to set the idle time for the Power Saver. The user can adjust the slider by dragging with the mouse. Figure 5-1 shows the Power Saver control panel.

Figure 5-1 Power Saver control panel



Video Input Digitizing Component

The Macintosh LC 630 and Macintosh Quadra 630 computers include a QuickTime component to support digitization of video from the video input module. For information about the use of digitizer components, refer to *Inside Macintosh: QuickTime Components.*

C H A P T E R 6

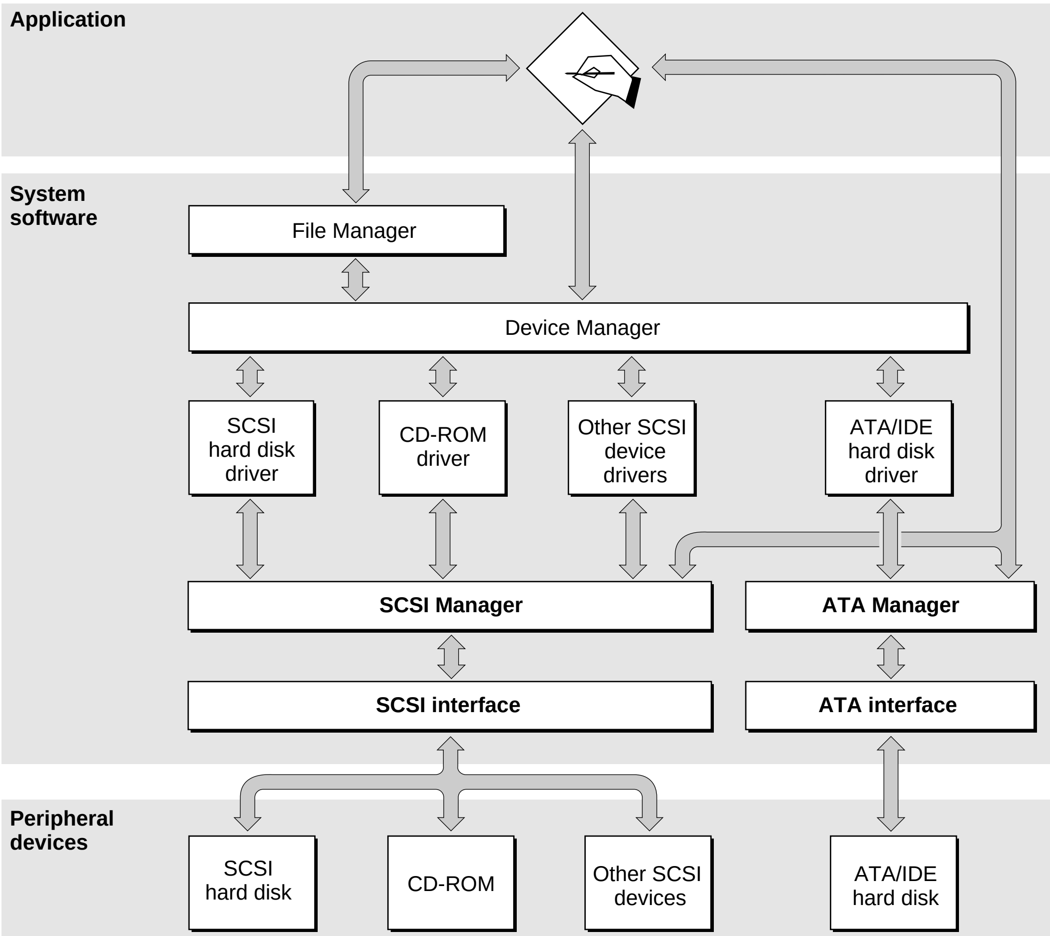
Software for the IDE Hard Disk

This chapter describes the system software that controls an IDE hard disk drive installed in a Macintosh computer. To use the information in this chapter, you should already be familiar with writing programs for the Macintosh computer that call device drivers to manipulate devices directly. You should also be familiar with the ATA IDE specification, ANSI proposal X3T9.2/90-143, Revision 3.1.

Introduction to IDE Software

Support for IDE (integrated drive electronics) hard disk drives is incorporated in the ROM software. System software for controlling IDE hard drives is included in a new IDE hard disk drive device driver and the ATA Manager. The relationship of the IDE hard disk drive device driver and the ATA Manager to the Macintosh system architecture is shown in Figure 6-1.

Figure 6-1 Relationship of the ATA Manager to the Macintosh system architecture



At the system level, the IDE device driver and ATA Manager work in the same way that the SCSI Manager and associated SCSI device drivers work. The IDE hard disk device driver provides drive partition, data management, and error-handling services for the operating system as well as support for determining device capacity and controlling device-specific features. The ATA Manager provides an interface to the IDE hard disk drive for the IDE device driver.

IDE hard disk drives appear on the desktop the same way SCSI hard disk drives currently do. Except for applications that perform low-level services, such as formatting and partitioning utilities, applications interact with the IDE hard disk drives in a device-independent manner through the File Manager or Printing Manager.

The IDE software for the Macintosh LC 630 and Macintosh Quadra 630 computers supports only synchronous data transfers.

IDE Hard Disk Device Driver

The IDE hard disk device driver for the Macintosh LC 630 and Macintosh Quadra 630 computers provides operating system-dependent services through a set of driver function calls required to interface with the Macintosh operating system. In addition, it provides additional control and status calls that are specific to the IDE hard disk driver driver implementation. The required driver calls, as specified in *Inside Macintosh: Devices*, are `open`, `close`, `prime`, `control`, and `status`.

In addition to the required function calls, the IDE hard disk device driver provides support for device-specific features. IDE hard disk device driver control and status calls are defined in “IDE Hard Disk Driver Reference” beginning on page 70.

At system startup time, if a RAM-based driver is not found on the IDE drive media, the IDE device driver in the ROM is installed as one of the device drivers. Note that this is different from the driver loading sequence for SCSI hard drive devices, which are RAM based drivers that are always loaded from the device media.

The IDE hard disk device driver has a driver reference number of –54 (decimal) and a driver name of `.ATDrvr.` Like all Macintosh device drivers, the IDE hard disk device driver can be called by using either the refNum –54 or the driver name `.ATDrvr.`

The IDE hard disk device driver does not provide request queuing. All driver requests are either completed immediately or are passed to the ATA Manager for further processing. For further information about the control calls for the IDE hard disk device driver, see “IDE Hard Disk Driver Reference.”

ATA Manager

The Macintosh ATA Manager schedules I/O requests from the IDE hard disk device driver, the operating system, and from applications. It is also responsible for managing the hardware interface to the IDE controller electronics.

When making calls to the ATA Manager you have to pass and retrieve parameter information through a parameter block. The size and contents of the parameter block depends on the call being made. However, all calls to the ATA Manager have a common parameter block header structure. The structure of the `ataPBHdr` parameter block is common to all ATA parameter block data types. Several additional ATA parameter block data types have been defined for the various function calls to the ATA Manager. The additional parameter block data types, which are specific to the function call being made, are described in "ATA Manager Reference" beginning on page 81.

IDE Hard Disk Driver Reference

This section describes the Macintosh device driver services provided by the IDE hard disk device driver. The information in this section assumes that you are already familiar with how to use device driver services on the Macintosh computer. If you are not familiar with Macintosh device drivers, refer to the chapter "Device Manager" in *Inside Macintosh: Devices* for additional information.

High-Level Device Manager Routines

The IDE hard disk driver supports the required set of high-level Device Manager routines, as defined in the chapter "Device Manager" of *Inside Macintosh: Devices*. Those routines are briefly defined here for convenience. Additional control functions supported in the IDE hard disk driver are defined in "IDE Hard Disk Driver Control Calls" beginning on page 73.

open

The `open` routine opens the IDE hard disk device driver during the boot sequence after the driver code is retrieved from the ROM. The `open` routine returns a reference number to the driver; that number is used in subsequent calls to the driver.

The following operations take place at boot time:

- memory allocation and driver globals and internal variables initialization
- power-on drive diagnostics
- device detection and verification
- device initialization
- device information uploading
- drive queue management and event posting

After booting, the driver responds with `noErr` to subsequent calls to the `open` routine, and does not repeat the operations performed at boot time.

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`openErr`	–23	Could not open the driver
`DRVRCantAllocate`	–1793	Global memory allocation error
`ATABufFail`	–1796	Device buffer test failed

close

The `close` routine deallocates the driver memory storage, removes the drive queue entry point, and closes the IDE hard disk device driver.

RESULT CODES

`noErr`	0	Successful completion, no error occurred

prime

The `prime` routine performs either a `read` or `write` command as specified by the caller. During this process the following operations take place:

- byte to block translation
- address translation
- update of the `IOParameter` block
- high-level error recovery and retry algorithm
- ATA Manager parameter block management

Refer to “ATA Manager” on page 69 for more information about the parameter block structure for the ATA Manager.

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`ioErr`	–36	I/O error
`paramErr`	–50	Invalid parameter specified
`nsDrvErr`	–56	No such drive installed

status

The `status` routine returns status information about the IDE hard disk device driver. The type of information returned is specified in the `csCode` field and the information itself is pointed to by the `csParamPtr` field.

The IDE hard disk device driver implements the same status calls supported by the SCSI hard disk device driver. The status calls supported by the IDE hard disk driver are shown in Table 6-1.

Table 6-1 Status calls supported by the IDE hard disk driver

Value of csCode	Definition
8	Return drive status information
43	Return driver Gestalt information
70	Power mode status information

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`statusErr`	–18	Unimplemented status call; could not complete requested operation
`nsDrvErr`	–56	No such drive installed

control

The `control` routine sends control information to the IDE hard disk device driver. The type of control call is specified in `csCode`.

The IDE driver implements the same control calls supported by the SCSI hard disk driver. The control calls are listed below and described in "IDE Hard Disk Driver Control Calls" beginning on page 73.

Value of csCode	Definition
1	Kill I/O
5	Verify media
6	Format media
7	Eject media
21	Return drive icon
22	Return media icon
23	Return drive characteristics
65	Need time code
70	Power-mode status management control

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`controlErr`	–17	Unimplemented control call; could not complete requested operation
`nsDrvErr`	–56	No such drive installed

IDE Hard Disk Driver Control Calls

The IDE hard disk driver supports a standard set of control calls for accessing IDE hard disk drive devices. The IDE hard disk driver also supports control calls for power management.

Standard Control Calls

This section describes the standard control calls defined within the IDE hard disk device driver.

killIO

The `killIO` function returns a `noErr` result if the logical drive number is valid, however, it performs no other operation on an IDE hard disk drive.

IMPORTANT
This function is not supported on the Macintosh LC 630 and Macintosh Quadra 630 computers. A call to `KillIO` returns a `controlErr` status. ▲

INPUT PARAMETERS

`csCode`	A value of 1
`ioVRefNum`	The logical drive number
`csParam[]`	None defined

OUTPUT PARAMETERS

`ioResult`	See result codes

RESULT CODES

`controlErr`	The control call is not supported

verify

The `verify` function requests a read verification of the data on the IDE hard drive media. This function performs no operation.

INPUT PARAMETERS

`csCode`	A value of 5
`ioVRefNum`	The logical drive number
`csParam[]`	None defined

OUTPUT PARMETERS

`ioResult`	See result codes

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	The specified logical drive number does not exist

format

The `format` function initializes the hard drive for use by the operating system. Because IDE hard drives are low-level formatted at the factory, this function does not perform any operation. The driver always returns `noErr` if the logical drive number is valid.

INPUT PARAMETERS

`csCode`	A value of 6
`ioVRefNum`	The logical drive number
`csParam[]`	None defined

OUTPUT PARAMETERS

`ioResult`	See result codes

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	The specified logical drive number does not exist

ejectMedia

The `ejectMedia` function prepares and initiates an eject operation from the specified drive. This function applies only to drives with removable media.

Note
The `ejectMedia` function is not supported in the Macintosh LC 630 and Macintosh Quadra 630 computers; this function returns a `noErr` if the logical drive number is valid. ◆

INPUT PARAMETERS

`csCode`	A value of 7
`ioVRefNum`	The logical drive number
`csParam[]`	None defined

OUTPUT PARAMETERS

`ioResult`	See result codes

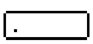
RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	The specified logical drive number does not exist

return drive icon

The `return drive icon` function returns a pointer to the device icon and the device name string. The drive icon is the same as the media icon for IDE hard disk drives. The drive icon for IDE hard disk devices is shown in Figure 6-2.

Figure 6-2 IDE hard disk drive icon



INPUT PARAMETERS

`csCode`	A value of 21
`ioVRefNum`	The logical drive number
`csParam[]`	None defined

OUTPUT PARAMETERS

`csParam[0–1]`	Address of drive icon and name string (information is in ICN# format)
`ioResult`	See result codes

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	The specified logical drive number does not exist

return media icon

The `return media icon` function returns a pointer to the media icon and the name string. The media icon is the same as the drive icon for IDE hard disk drives. The media icon for IDE hard disk devices is shown in Figure 6-2.

INPUT PARAMETERS

`csCode`	A value of 22
`ioVRefNum`	The logical drive number
`csParam[]`	None defined

OUTPUT PARAMETERS

`csParam[0–1]`	Address of media icon and name string (information is in ICN# format)
`ioResult`	See result codes

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	The specified logical drive number does not exist

Return Drive Characteristics

The `return drive characteristics` function returns information about the characteristics of the specified drive as defined in *Inside Macintosh*, Volume V.

INPUT PARAMETERS

`csCode`	A value of 23
`ioVRefNum`	The logical drive number
`csParam[]`	None defined

OUTPUT PARAMETERS

`csParam[0–1]`	Drive information
	$0601 = primary, fixed, SCSI, internal
	$0201 = primary, removable, SCSI, internal
`ioResult`	See result codes

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	The specified logical drive number does not exist

needTime code

The `needTime code` function provides time for the driver to perform periodic operations such as checking for media insertion or ejection events related to removable cartridge drives. For additional information about how this function is used, see the description of the driver flag `dNeedTime` in the chapter "Device Manager" of *Inside Macintosh: Devices*. This function performs no operation on the IDE hard disk drive in a Macintosh LC 630 or Macintosh Quadra 630 computer.

INPUT PARAMETERS

`csCode`	A value of 65
`csParam[]`	None defined

OUTPUT PARAMETERS

`ioResult`	See result codes

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	The specified logical drive number does not exist

Power Management Control Calls

Power management functions can be used to reduce drive power consumption and decrease system noise levels by putting the hard drive into a standby state.

Note
Power management control calls are most useful on PowerBook computers, where they can be used to reduce drive power consumption and thereby extend useful battery life. ◆

Power Management

The `power management` function provides three modes of operation for IDE hard disk drives: idle, standby, and sleep.

In the idle state, the non-essential electronics on the IDE hard drive are disabled. For example, the read and write channels are disabled during the idle state. The spindle motor remains enabled during the idle state, so the drive still responds immediately to any commands requesting media access.

In the standby state, the head is parked and the spindle motor is disabled. The drive interface remains active and is still capable of responding to commands. However, it can take several seconds to respond to media access commands, because the drive's spindle motor must return to full speed before a media access can take place.

In the sleep state, the drive interface and spindle motor are disabled. To return the drive to full operation after the sleep state has been enabled, the user must restart or reset the computer.

INPUT PARAMETERS

`csCode`	A value of 70
`ioVRefNum`	The logical drive number
`csParam[0]`	The most significant byte contains one of the following codes: 0 = enable the active mode 1 = enable the standby mode 2 = enable the idle mode 3 = enable the sleep mode

OUTPUT PARAMETERS

`ioResult`	See result codes

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`controlErr`	–17	The power management information couldn't be returned due to a manager error
`nsDrvErr`	–56	The specified logical drive number does not exist

IDE Hard Disk Driver Status Calls

This section describes three functions for retrieving status and error information from the IDE hard disk device driver.

drive status info

The IDE hard disk device driver provides a drive status function for retrieving status information from the drive. The `drive status info` function returns the same type of information that disk drivers are required to return for the `DriveStatus` function, as described in the chapter "Device Manager" in *Inside Macintosh: Devices*.

INPUT PARAMETERS

`csCode`	A value of 8
`ioVRefNum`	The logical drive number
`csParam[]`	None defined

OUTPUT PARAMETERS

`csParam[]`	The `csParam` field contains status information about the internal IDE disk drive
`ioResult`	See result codes

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	The specified logical drive number does not exist

return driver gestalt

The `return driver gestalt` function provides the application information about the IDE hard disk driver and the attached device. Several calls are supported under the function. A Gestalt selector is used to specify a particular call.

The `DriverGestaltParam` data type defines the IDE Gestalt parameter block:

```
typedef struct DriverGestaltParam
{
   ataPBHdr             /* See definition on page 69 */
   short                ioVRefNum;              /* refNum of device */
   short                csCode;                 /* Driver gestalt code */
   OSType               driverGestaltSelector;  /* Gestalt selector */
   driverGestaltInfo    driverGestaltResponse;  /* Returned result */
} DriverGestaltParam;
```

The fields `driverGestaltSelector` and `driverGestaltResponse` are 32-bit fields.

INPUT PARAMETERS

`csCode`	A value of 43		
`ioVRefNum`	The logical drive number		
`driverGestaltSelector`	Gestalt function selector. This is a 32-bit ASCII field containing one of the following selectors:		
		sync	Indicate synchronous or asynchronous driver
		`devt`	Specify type of device the driver is controlling
		`intf`	Specify the device interface
		`boot`	Specify PRAM value to designate this driver or device
		`vers`	Specify the version number of the driver

OUTPUT PARAMETERS

`driverGestaltResponse`	Returned result based on the driver gestalt selector. The possible four-character return values are:		
		`TRUE`	If the `sync` driver selector is specified, this Boolean value indicates that the driver is synchronous; a `FALSE` value indicates asynchronous
		`disk`	If the `devt` driver selector is specified, this value indicates a hard disk driver
		`ide`	If the `intf` driver selector is specified, this value indicates the interface is IDE
		`0`	If the `boot` driver selector is specified, this value indicates that this is the boot driver or boot device
		nnnn	If the `vers` selector is specified, the current version number of the driver is returned
`ioResult`	See result codes		

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	The specified logical drive number does not exist
`statusErr`	–18	Unknown selector was specified

power-mode status

The `power-mode status` call returns the current power mode state of the internal hard disk.

INPUT PARAMETERS

`csCode`	A value of 70
`ioVRefNum`	The logical drive number
`csParam[]`	None defined

OUTPUT PARAMETERS

`csParam[]`	The most significant byte of this field contains one of the following values: 1 = drive is in standby mode 2 = drive is in idle mode 3 = drive is in sleep mode
`ioResult`	See result codes

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	The specified logical drive number does not exist
`statusErr`	–18	The power management information couldn't be returned due to a manager error

ATA Manager Reference

This section defines the data structures and functions that are specific to the ATA Manager. The section "The ATA Parameter Block" shows the data structure of the ATA parameter block. The "Functions" section describes the functions for managing and performing data transfers through the ATA Manager.

The ATA Parameter Block

This section defines the fields that are common to all ATA Manager functions that use the ATA parameter block. The fields that are used for specific functions are defined in the description of the functions to which they apply. You use the ATA parameter block for all calls to the ATA Manager. The `ataPBHdr` data type defines the ATA parameter block.

The parameter block includes a field, `MgrFCode`, in which you specify the function selector for the particular routine to be executed; you must specify a value for this field. Each ATA function may use different fields of the ATA parameter block for parameters specific to that function.

An arrow preceding the comment indicates whether the parameter is an input parameter, an output parameter, or both.

Arrow	**Meaning**
→	Input
←	Output
↔	Both

The ATA parameter block header structure is defined as follows:

```
typedef    unsigned char   uchar;
typedef    unsigned short  ushort;
typedef    unsigned long   ulong;

typedef     struct   ataPBHdr        /* ATA Manager parameter block
                                      header structure */
{
            Ptr      ataLink;        /* Reserved */
            short    ataQType;       /* Type byte */
            uchar    ataPBVers;      /* → Parameter block
                                        version number */
            uchar    hdrReserved;    /* Reserved */
            Ptr      hdrReserved2;   /* Reserved */
            ProcPtr  ataCompletion;  /* Completion routine */
            short    ataResult;      /* ← Returned result */
            uchar    MgrFCode;       /* → Manager function code */
            uchar    ataIOSpeed;     /* → I/O timing class */
            ushort   ataFlags;       /* → Control options */
            short    hdrReserved3;   /* Reserved */
            long     deviceID;       /* → Device ID */
            ulong    TimeOut;        /* → Transaction timeout
                                        value */
            Ptr      ataPtr1;        /* Client storage Ptr 1 */
            Ptr      ataPtr2;        /* Client storage Ptr 2 */
```

```
            ushort   ataState;      /* Reserved, init to 0 */
            short    hdrReserved4;  /* Reserved */
            long     hdrReserved5;  /* Reserved */
} ataPBHdr;
```

Field descriptions

`ataLink` This field is reserved for use by the ATA Manager. It is used internally for queuing I/O requests. It must be initialized to 0 before calling the ATA Manager.

`ataQType` This field is the queue type byte. It should be initialized to 0 before calling the ATA Manager.

`ataPBVers` This field contains the parameter block version number. A value of 1 is the only value currently supported. Any other value results in a `paramErr`.

`hdrReserved` Field reserved for future use. To ensure future compatibility, all reserved fields should be set to 0.

`hdrReserved2` Field reserved for future use. To ensure future compatibility, all reserved fields should be set to 0.

`ataCompletion` This field contains the completion routine pointer that is to be called upon completion of the request. When this field is set to 0, it indicates a synchronous I/O request; a non-zero value indicates an asynchronous I/O request. The routine pointed by this field is called when the request has finished without error, or when the request has terminated due to an error. This field is valid for any manager request. The completion routine is called as follows:

```
pascal void (*RoutinePtr) (ataIOPB *)
```

The completion routine is called with the associated manager parameter block in the stack.

`ataResult` Completion status. This field is returned by the ATA Manager after the request has been completed. Refer to Table 6-6 on page 100 for a list of the possible error codes returned in this field.

`MgrFCode` This field is the function selector for the ATA Manager. The functions are defined in Table 6-4 on page 87. An invalid code in this field results in an `ATAFuncNotSupported` error.

`ataIOSpeed` This field specifies the I/O cycle timing requirement of the specified IDE drive. This field should contain word 51 of the identify drive data. Currently values 0 through3 are supported, as defined in the ATA specification. See the ATA specification for the definitions of the timing values. If a timing value higher than one supported is specified, the manager operates in the fastest timing mode supported by the manager. Until the timing value is determined by examining the identify drive data returned by the `ATA_Identify` routine, the client should request operations using the slowest mode (mode 0).

`ataFlags` This 16-bit field contains control settings that indicate special handling of the requested function. The control bits are defined in Table 6-3 on page 85.

`hdrReserved3` Field reserved for future use. To ensure future compatibility, all reserved fields should be set to 0.

`deviceID` A short word that uniquely identifies an IDE device. The field consists of the following structure:

```
typedef struct     /* device ID structure */
{
ushort Reserved;   /* The upper word is reserved */
ushort deviceNum;  /* consists of device ID and
                      bus ID */
} deviceIdentification;
```

Bit 15 of `deviceNum` field indicates master (=0) /slave (=1) selection. Bits 14 through 0 contain the bus ID (for example, 0x0 = master unit of bus 0, 0x80 = slave unit of bus 0). The present implementation allows only one device in the master configuration. This value is always 0.

`TimeOut` This field specifies the transaction timeout value in milliseconds. A value of zero disables the transaction timeout detection.

`ataPtr1` This pointer field is available for application use. It is not modified by the ATA Manager.

`ataPtr2` This pointer field is available for application use. It is not modified by the ATA Manager.

`ataState` This field is used by the ATA Manager to keep track of the current bus state. This field must contain zero when calling the manager. Bus states are defined in Table 6-2.

`hdrReserved4` Field reserved for future use. To ensure future compatibility, all reserved fields should be set to 0.

Table 6-2 IDE drive bus states

Code	Name	Description
$00	Initial	Parameter block processing started
$01	Started	Command delivery state
$0F	Data	Data delivery state
$1F	Status	Status delivery state
$3F	Complete	Bus transaction complete state
$FF	Idle	Waiting to return state

Table 6-3 Control bits in the `ataFlags` field

Name	Bit	Definition
–	0–2	Reserved.
`RegUpdate`	3	When set to 1 this bit indicates that a set of device registers should be reported back upon completion of the request. This bit is valid for the `Execute I/O` function only. Refer to the description of `Execute I/O` for details. The following device registers are reported back: Sector count register Sector number register Cylinder register(s) SDH register
–	4–7	Reserved.
`SGType`	8, 9	This 2-bit field specifies the type of scatter gather list passed in. This field is only valid for read/write operations. The following types are defined: 00 = Scatter gather disabled 01 = Scatter gather type I enabled 10 = Reserved 11 = Reserved When set to 0, this field indicates that the `ioBuffer` field contains the host buffer address for this transfer, and the `ioReqCount` field contains the byte transfer count. When set to 1, this field indicates that the `ioBuffer` and the `ioReqCount` fields of the parameter block for this request point to a host scatter gather list and the number of scatter gather entries in the list, respectively. The format of the scatter gather list is a series of the following structure definition:

```
typedef struct       /* SG entry structure */
{
  uchar* ioBuffer;   /* → Data buffer pointer */
  ulong  ioReqCount; /* → Byte count */
} IOBlock;
```

continued

Table 6-3 Control bits in the `ataFlags` field (continued)

Name	Bit	Definition
`QLockOnError`	10	When set to 0, this bit indicates that an error during the transaction should not freeze the I/O queue for the device. When an error occurs on an I/O request with this bit set to 0, the next queued request is processed following this request. When an error occurs on an I/O request with this bit set to 1, the user must issue an `I/O Queue Release` command to continue. A status code of 717 hex is returned for subsequent asynchronous I/O requests until the `I/O Queue Release` command is issued.
`Immediate`	11	When this bit is set to 1, it indicates that the request is executed as soon as possible and the status of the request is returned. It forces the request to the head of the I/O queue for immediate execution. When this bit is set to 0, the request is queued in the order it is received and is executed according to that order.
`ATAioDirection`	12, 13	This bit field specifies the direction of data transfer. Bit values are binary and defined as follows: 00 = No data transfer 10 = Data direction in (read) 01 = Data direction out (write) 11 = Reserved
—	14	Reserved.
`ByteSwap`	15	When set to 1, this bit indicates that every byte of data prior to transmission on write operations and upon reception on read operations is to be swapped. When this bit is set to 0, it forces bytes to go out in the LSB-MSB format that is compatible with IBM clones. Typically, this bit should be set to 0. Setting this bit has performance implications because the byte swap is performed by the software. Use this bit with caution.

Functions

This section describes the ATA Manager services that are used to manage and perform data transfers. Each service is requested through a parameter block specific to that service. A request for an IDE service is specified by a function code within the parameter block. The entry point for all the functions is the same.

The function names and ATA Manager function codes are shown in Table 6-4.

Table 6-4 ATA Manager functions

Function name	Code	Description
`ATA_NOP`	$00	No operation
`ATA_ExecIO`	$01	Execute ATA I/O
`ATA_BusInquiry`	$03	Bus inquiry
`ATA_QRelease`	$04	I/O queue release
`ATA_Abort`	$10	Terminate command
`ATA_ResetBus`	$11	Reset IDE bus
`ATA_RegAccess`	$12	ATA device register access
`ATA_Identify`	$13	Get the drive identification data
`ATA_DrvrRegister`	$85	Register the drive reference number
`ATA_FindRefNum`	$86	Lookup driver reference number
`ATA_DrvrDeregister`	$87	Deregister the driver reference number
`ATA_MgrInquiry`	$90	ATA Manager inquiry
`ATA_MgrInit`	$91	Initialize ATA Manager
`ATA_MgrShutDown`	$92	Shutdown ATA Manager

ATA_ExecI/O

You can use the `ATA_ExecIO` function to perform all data I/O transfers to or from an IDE device. Your application must provide all of the parameters needed to complete the transaction prior to calling the ATA Manager. Upon return, the parameter block contains the result of the request.

A prior call to the `ATA_MgrInit` function to initialize the ATA Manager must be performed before issuing the `ATA_ExecIO` function. See page 91 for information about calling the `ATA_MgrInit` function.

The manager function code for the `ATA_ExecIO` function is 1.

The parameter block associated with the `ATA_ExecIO` function is defined below:

```
typedef       struct         /* ATA_ExecIO structure */

{
   ataPBHdr                  /* See definition on page 82 */
   char      ataStatusReg;   /* ← Last device status register
                                 image */
   char      ataErrorReg;    /* ← Last device error register
                                 image (valid if bit 0 of Status
```

```
                                field is set) */
   short     ataReserved;    /* Reserved */
   ulong     BlindTxSize;    /* → Data transfer size */
   IOBlock   IOBlk;          /* → Data transfer
                                descriptor block */
   ulong     ataActualTxCnt; /* ← Actual number of bytes
                                transferred */
   ulong     ataReserved2;   /* Reserved */
   devicePB  RegBlock;       /* → Device register images */
   uchar*    packetCDBPtr;   /* ATAPI packet command block
                                pointer */
   ushort    ataReserved3[6];/* Reserved */

} ATA_ExecIO;
```

Field descriptions

`ataPBHdr` — See the definition of the `ataPBHdr` structure on page 82.

`ataStatusReg` — This field contains the last device status register image. See the ATA specification for status register bit definitions.

`ataErrorReg` — This field contains the last device error register image. This field is valid only if the error bit (bit 0) of the `Status` register is set. See the ATA specification for error register bit definitions.

`ataReserved` — Reserved. All reserved fields are set to 0 for future compatibility.

`BlindTxSize` — This field specifies the maximum number of bytes that can be transferred per interrupt or detection of DRQ. Bytes are transferred in blind mode (no byte level handshake). Once an interrupt or a DRQ condition is detected, the manager transfers up to the number of bytes specified in the field from or to the selected device. The typical number is 512 bytes.

`IOBlk` — This field contains either the host buffer address and the requested transfer length, or the pointer to a scatter gather list and the number of scatter gather entries. If the `SGType` bits of the `ataFlags` field are set, the `IOBlk` field contains the scatter gather information. The `IOBlk` field is defined as follows:

```
typedef      struct
{
     uchar*  ioBuffer;   /* ↔ Data buffer ptr */
     ulong   ioReqCount; /* ↔ Transfer length */
} IOBlk;
```

`ioBuffer` — This field contains the host buffer address for the number of bytes specified in the `ioReqCount` field. Upon returning, the `ioBuffer` field is updated to reflect data transfers. When the `SGType` bits of the ataFlags field are set, the ioBuffer field points to a scatter gather list. The scatter gather list consists of a series of IOBlk entries.

ioReqCount — This field contains the number of bytes to transfer either from or to the buffer specified in ioBuffer. Upon returning, the ioReqCount field is updated to reflect data transfers (0 if successful; otherwise, the number of bytes that remained to be transferred prior to the error condition). When the SGType bits of the ataFlags field are set, the ioReqCount field contains the number of scatter gather entries in the list pointed to by the ioBuffer field.

ataActualTxCnt — This field contains the total number of bytes transferred for this request. Currently not supported.

ataReserved2 — This field is reserved.

RegBlock — This field contains the IDE device register image structure. Values contained in this structure are written out to the device during the command delivery state. The caller must provide the image prior to calling the ATA Manager. The IDE device register image structure is defined as follows:

```
typedef      struct     /* Device register images */
{     uchar  Features;  /* → Features register
                             image */
      uchar  Count;     /* ↔ Sector count */
      uchar  Sector;    /* ↔ Sector start/finish */
      uchar  Reserved;  /* Reserved */
      ushort Cylinder;  /* ↔ Cylinder 68000 format */
      uchar  SDH;       /* ↔ SDH register image
      uchar  Command;   /* → Command register image */
} Device_PB;
```

packetCDCPtr — This field contains the packet pointer for ATAPI. The current version of the ATA Manager doesn't support the ATAPI protocol. For ATA commands, the packetCDCPtr field should contain 0 in order to ensure compatibility in the future.

ataReserved3[6] — These fields are reserved.

RESULT CODES

noErr	0	Successful completion, no error occurred
nsDrvErr	–56	Specified logical drive number does not exist
ATAMgrNotInitialized	–1802	ATA Manager not initialized
ATABusyErr	–1790	Selected device busy (BUSY bit set)
ATATransTimeOut	–1806	Timeout: transaction timeout detected
AT_CorDataErr	–1785	Data corrected bit set in status register
AT_ASeekErr	–1783	Seek complete bit not set upon completion
AT_WrFltErr	–1782	Write fault bit set in status register

`AT_UncDataErr`	–1784	Uncorrected data bit set in error register
`AT_IDNFErr`	–1788	ID not found bit set in error register
`AT_DMarkErr`	–1787	Data mark not found bit set in error register

continued

`AT_BadBlkErr`	–1786	Bad block bit set in error register
`AT_RecalErr`	–1781	Track 0 not found bit set in error register
`AT_AbortErr`	–1780	Command aborted bit set in error register
`ATAUnknownState`	–1808	Device in unknown state
`ATAPBInvalid`	–1803	Invalid device base address detected (= 0)
`ATAQLocked`	–1809	I/O queue locked—cannot proceed
`ATAReqInProg`	–1807	I/O channel in use—cannot proceed

ATA_MgrInquiry

The `ATA_MgrInquiry` function gets information, such as the version number, about the ATA Manager. This function may be called prior to the manager initialization, however the system configuration information may be invalid.

The manager function code for the `ATA_MgrInquiry` function is $90.

The parameter block associated with this function is defined below:

```
typedef        struct           /* IDE inquiry structure */
{
   ataPBHdr                     /* See definition on page 82 */
   NumVersion  MgrVersion
   ushort      MGRPBVers;       /* ← Manager PB version number
                                    supported */
   ushort      Reserved1;       /* Reserved */
   ushort      ataBusCnt;       /* ← Number of ATA buses in
                                    system */
   ushort      ataDevCnt;       /* ← Number of ATA devices
                                    detected */
   unchar      ataMaxMode;      /* ← Maximum I/O speed mode */
   ushort      Reserved2;       /* Reserved */
   ushort      IOClkResolution; /* ← IO Clock in nsec */
   ushort      Reserved[17];    /* Reserved */
} ATA_MgrInquiry;
```

Field descriptions

`ataPBHdr` See the `ataPBHdr` parameter block definition on page 82.

`MgrVersion` Upon return, this field contains the version number of the ATA Manager.

`MGRPBVers` This field contains the number corresponding to the latest version of the parameter block that is supported. A client may use any parameter block definition up to this version.

`Reserved` Reserved. All reserved fields are set to 0 for future compatibility.

`ataBusCnt` Upon return, this field contains the total number of ATA buses in the system. This field contains a zero if the ATA Manager has not been initialized.

`ataDevCnt` Upon return, this field contains the total number of ATA devices detected on all ATA buses. The current architecture allows only one device per bus. This field will contain a zero if the ATA Manager has not been initialized.

`ataMaxMode` This field specifies the maximum I/O speed mode that the ATA Manager supports. Refer to the ATA specification for information on mode timing.

`IOClkResolution`
This field contains the I/O clock resolution in nanoseconds. The current implementation doesn't support the field (returns 0).

RESULT CODES

`noErr`	0	Successful completion, no error occurred

ATA_MgrInit

You must call the `ATA_MgrInit` function prior to issuing `ATA_ExecIO`, `ATA_BusInquiry`, `ATA_QRelease`, or `ATA_Abort` commands. Otherwise, an `IDEMgrNotInitialized` error is returned. This call initializes internal variables and IDE hard disk drive hardware. Consecutive initialization calls from either the same or another application are ignored and the `NoErr` result is returned.

The manager function code for the `ATA_MgrInit` function is $91.

The parameter block associated with this function is defined below:

```
typedef       struct          /* IDE Init structure */
{
   ataPBHdr                   /* See definition on page 82 */
   ushort     Reserved[24]    /* Reserved */
} ATA_MgrInit;
```

Field descriptions

`ataPBHdr` See the definition of the `ataPBHdr` parameter block on page 82.

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`ATAInitFail`	–1795	ATA Manager initialization failure

ATA_BusInquiry

The `ATA_BusInquiry` function gets information about a specific ATA bus. This call is provided for possible future expansion of the Macintosh ATA architecture.

The manager function code for the `ataBusInquiry` function is $03.

The parameter block associated with this function is defined below:

```
typedef struct                      /* IDE bus inquiry structure */
{
  ataPBHdr                          /* See definition on page 82 */
ushort      ataEngineCount;         /* ← TBD; zero for now */
  ushort    ataReserved;            /* Reserved */
  ulong     ataDataTypes;           /* ← TBD; zero for now */
  ushort    ataIOpbSize;            /* ← Size of ATA IO PB */
  ushort    ataMaxIOpbSize;         /* ← TBD; zero for now */
  ulong     ataFeatureFlags;        /* ← TBD */
  uchar     ataVersionNum;          /* ← HBA Version number */
  uchar     ataHBAInquiry;          /* ← TBD; zero for now
  ushort    ataReserved2;           /* Reserved */
  ulong     ataHBAPrivPtr;          /* ← Ptr to HBA private data */
  ulong     ataHBAPrivSize;         /* ← Size of HBA private data */
  ulong     ataAsyncFlags;          /* ← Capability for callback */
  ulong     ataReserved3[4];        /* Reserved */
  ulong     ataReserved4;           /* Reserved */
  char      ataReserved5[16];       /* TBD */
  char      ataHBAVendor[16];       /* ← HBA Vendor ID */
  char      ataContrlFamily[16];/* ← Family of ATA controller */
  char      ataContrlType[16];      /* ← Model num of controller */
  char      ataXPTversion[4];       /* ← version number of XPT */
  char      ataReserved6[4];        /* Reserved */
  char      ataHBAversion[4];       /* ← version number of HBA */
  uchar     ataHBAslotType;         /* ← type of slot */
  uchar     ataHBAslotNum;          /* ← slot number of the HBA */
  ushort    ataReserved7;           /* Reserved */
  ulong     ataReserved8;           /* Reserved */
} ATA_BusInquiry;
```

Field descriptions

`ataPBHdr`	See the definition of the `ataPBHdr` on page 82.
`ataEngineCount`	Currently 0.
`ataReserved`	Reserved. All reserved fields are set to 0.
`ataDataTypes`	Not supported by current ATA architecture. Returns a bit map of data types supported by this HBA. The data types are numbered from 0 to 30; 0 through 15 are reserved for Apple definition and 16 through 30 are available for vendor use. Returns 0.
`ataIOpbSize`	This field specifies the size of the parameter block supported. This field contains the size of the I/O parameter block.
`ataMaxIOpbSize`	This field specifies the maximum I/O size for the HBA. This field is currently not supported and returns 0.
`ataFeatureFlags`	This field specifies supported features. Not supported; returns 0.
`ataVersionNum`	The version number of the HBA is returned. The current version returns a 1.
`ataHBAInquiry`	Reserved.
`ataHBAPrivPtr`	This field contains a pointer to the HBA's private data area. Not supported; returns 0.
`ataHBAPrivSize`	This field contains the byte size of the HBA's private data area. Not supported; returns 0.
`ataAsyncFlags`	These flags indicate which types of asynchronous events the HBA is capable of generating. Not supported; returns 0.
`ataHBAVendor`	This field contains the vendor ID of HBA. This is an ASCII text field. Not supported.
`ataContrlFamily`	Reserved.
`ataContrlType`	This field identifies the specific type of ATA controller. Not supported; returns 0.
`ataXPTversion`	Reserved.
`ataHBAversion`	This field specifies the version of the HBA. Not supported; returns 0.
`ataHBAslotType`	This field specifies the type of slot. Not supported; returns 0.
`ataHBAslotNum`	This field specifies the slot number of the HBA. Not supported; returns 0.

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`ATAMgrNotInitialized`	–1802	ATA Manager not initialized

ATA_QRelease

The `ATA_QRelease` function releases the frozen I/O queue of the selected device.

When the ATA Manager detects an I/O error and the `QLockOnError` bit of the parameter block is set for the request, the ATA Manager freezes the queue for the selected device. No pending or new requests are processed or receive status until the queue is released through the `ATA_QRelease` command. Only those requests with the `Immediate` bit set in the `ATAFlags` field of the `ataPBHdr` parameter block are processed. Consequently, for the ATA I/O queue release command to be processed, it must be issued with the `Immediate` bit set in the parameter block. An ATA I/O queue release command issued while the queue isn't frozen returns the `noErr` status.

The manager function code for the `ATA_QRelease` function is $04.

There are no additional function-specific variations on `ataPBHdr` for this call.

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	Specified drive does not exist
`ATAMgrNotInitialized`	–1802	ATA Manager not initialized

ATA_NOP

The `ATA_NOP` function performs no operation across the interface and does not change the state of either the manager or the device. It returns `noErr` if the drive number is valid.

The manager function code for the `ATA_NOP` function is $00.

There are no additional function-specific variations on `ataPBHdr` for this call.

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	Specified drive does not exist

ATA_Abort

The `ATA_Abort` function terminates a specified queued I/O request. This function applies to asynchronous I/O requests only. The `ATA_Abort` function searches through the I/O queue associated with the selected device and aborts the matching I/O request. The current implementation does not abort if the found request is in progress. If the specified I/O request is not found or has started processing, an `ATAUnableToAbort` status is returned. If aborted, the `ATAReqAborted` status is returned.

It is up to the application that called the `ATA_Abort` function to clean up the aborted request. Clean up includes parameter block deallocation and O/S reporting.

The manager function code for the `ATA_Abort` function is $10.

The parameter block associated with this function is defined as follows:

```
typedef      struct        /* IDE abort structure */
{
  ataPBHdr                 /* See definition on page 82 */
  ATA_PB*    AbortPB       /* Address of the parameter block of
                              the function to be aborted */
  ushort     Reserved      /* Reserved */
} ATA_Abort;
```

Field descriptions

`ataPBHdr` See the definition of the `ataPBHdr` parameter block on page 82.

`AbortPB` This field contains the address of the I/O parameter block to be aborted.

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	Specified drive does not exist
`ATAMgrNotInitialized`	–1802	ATA Manager not initialized
`ATAReqAborted`	–1810	The request was aborted
`ATAUnableToAbort`	–1811	Request to abort couldn't be honored

ATA_RegAccess

The `ATA_RegAccess` function enables access to a particular device register of a selected device. This function is used for diagnostic and error recovery processes.

The manager function code for the `ATA_RegAccess` function is $12.

The parameter block associated with this function is defined below:

```
typedef       struct          /* Register access structure */
{
   ataPBHdr                   /* See definition on page 82 */
   ushort     RegSelect;      /* → Device register selector */
   uchar      RegValue;       /* ↔ Register value
                                to read or to be written */
   uchar      Reserved;       /* ↔ Used for data register
                                (LSB) only */
   uchar      Reserved[22]    /* Reserved */
} ATA_RegAccess;
```

Field descriptions

`ataPBHdr`	See the definition of the `ataPBHdr` parameter block on page 82.
`RegSelect`	This field specifies which of the device registers to access. The selectors for the registers supported by the `ATA_RegAccess` function are listed in Table 6-5.
`RegValue`	This field represents the value to be written (`IDEioDirection = 01b`) or the value read from the selected register (`ATAioDirection` = `10` binary). For `DataReg`, it is assumed this field is a 16-bit field; for other registers, an 8-bit field.

Table 6-5 IDE register selectors

Selector name	Selector	Register description
`DataReg`	0	Data register (16-bit access only)
`ErrorReg`	1	Error register (R) or features register (W)
`SecCntReg`	2	Sector count register
`SecNumReg`	3	Sector number Register
`CylLoReg`	4	Cylinder low register
`CylHiReg`	5	Cylinder high register
`SDHReg`	6	SDH register
`StatusReg` `CmdReg`	7	Status register (R) or command register (W)
`AltStatus` `DevCntr`	14	Alternate status (R) or device control (W)
`AddrReg`	15	Digital input register

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	Specified drive does not exist

ATA_Identify

The `ATA_Identify` function returns the device identification data from the selected device. The identification data contains information necessary to perform I/O to the device. Refer to the ATA Specification for the format and the information description provided by the data.

The manager function code for the `ATA_Identify` function is $13.

The parameter block associated with this function is defined below:

```
typedef     struct
{
  ataPBHdr
  ushort    Reserved1[4];     /* Reserved */
  uchar     *DataBuf;         /* ↔ Buffer for the data */
  ushort    Reserved2[18];    /* Reserved */
} ATA_Identify;
```

Field descriptions

`ataPBHdr`	See the definition of the `ataPBHdr` parameter block on page 82.
`DataBuf`	Pointer to the data buffer for the device identify data. The length of the buffer must be at least 512 bytes.

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	Specified drive does not exist

ATA_ResetBus

The `ATA_ResetBus` function resets the specified IDE bus. This function performs a soft reset operation to the selected IDE bus. The IDE interface doesn't provide a way to reset individual units on the bus. Consequently, all devices on the bus will be affected.

IMPORTANT

This function should be used with caution since it may terminate any active requests to devices on the bus. ▲

The manager function code for the `ATA_ResetBus` function is $11.

The parameter block associated with this function is defined below:

```
typedef     struct          /* IDE reset structure */
{
  ataPBHdr                  /* See definition on page 82 */
  char      Status;         /* ← Last ATA status register image */
  char      Reserved;       /* Reserved */
  ushort    Reserved[23]; /* Reserved */
} ATA_ResetBus;
```

Field descriptions

`ataPBHdr`	See the definition of the `ataPBHdr` parameter block on page 82.
`Status`	This field contains the last device status register image following the bus reset. See the ATA specification for defintions of the status register bits.

RESULT CODES

noErr	0	Successful completion, no error occurred
nsDrvErr	–56	Specified drive does not exist

ATA_MgrShutDown

The ATA_MgrShutDown function shuts down the ATA Manager. It is the complement to the ATA_MgrInit function. This function deallocates all of the global space currently in use by the ATA Manager. After calling ATA_MgrShutDown, the ATA Manager must be reinitialized before any IDE I/O requests can take place.

The ATA_MgrShutDown function always returns a status of noErr.

The manager function code for the ATA_MgrShutDown function is $92.

There are no additional function-specific variations on ataPBHdr for this call.

IMPORTANT
This function should be used with caution if multiple client applications are present. ▲

RESULT CODES

noErr	0	Successful completion, no error occurred

ATA_DrvrRegister

The ATA_DrvrRegister function registers the driver reference number passed in for the selected drive. The function doesn't check for the existence of another driver.

The manager function code for the ATA_DrvrRegister function is $85.

The parameter block associated with this function is defined below:

```
typedef      struct          /* Driver register structure */
{
   ataPBHdr                  /* See definition on page 82 */
   short     drvrRefNum;     /* → Driver reference number */
   ushort    FlagReserved;   /* Reserved */
   ushort    deviceNextID;   /* Not used */
   short     Reserved[21];   /* Reserved */
} ATA_DrvrRegister;
```

Field descriptions

`ataPBHdr`	See the `ataPBHdr` parameter block definition on page 82.
`drvrRefNum`	This field specifies the driver reference number to be registered. This value must be less than 0 to be valid.
`FlagReserved`	Reserved.
`deviceNextID`	Not used by this function.

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	Specified drive does not exist

ATA_DrvrDeregister

The `ATA_DrvrDeRegister` function deregisters the driver reference number passed in for the selected drive. After successful completion of this function, the driver reference number for the drive is set to 0, which indicates that there is no driver in control of this device.

The manager function code for the `ATA_DrvrDeRegister` function is $87.

There are no additional function-specific variations on `ataPBHdr` for this call.

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	Specified drive does not exist

ATA_FindRefNum

The `ATA_FindRefNum` function allows an application to determine whether a driver has been installed for a given device. You pass in a device ID and the function returns the current driver reference number registered for the given device. A value of 0 indicates that no driver has been registered. The `deviceNextID` field contains a device ID of the next device in the list. The end of the list is indicated with a value of `0xFF`.

To create a list of all drivers for the attached devices, pass in `0xFF` for `deviceID`. This causes `deviceNextID` to be filled with the first device in the list. Each successive driver can be found by moving the value returned in `deviceNextID` into `deviceID` until the function returns `0xFF` in `deviceNextID`, which indicates the end of the list.

The manager function code for the `ATA_FindRefNum` function is $86.

The parameter block associated with this function is defined as follows:

```
typedef      struct
{
   ataPBHdr
   short      drvrRefNum;     /* ← Contains the driver refNum */
   ushort     FlagReserved;   /* Reserved */
   ushort     deviceNextID;   /* ← Contains the next drive ID */
   short      Reserved[21];   /* Reserved */
} ATA_FindRefNum;
```

Field descriptions

`ataPBHdr`	See the `ataPBHdr` parameter block definition on page 82.
`drvrRefNum`	Upon return, this field contains the reference number for the device specified in the `deviceID` field of the `ataPBHdr` data.
`deviceNextID`	Upon return, this field contains the `deviceID` of the next device on the list.

RESULT CODES

`noErr`	0	Successful completion, no error occurred
`nsDrvErr`	–56	Specified drive does not exist

Error Code Summary

A summary of the IDE hard disk drive error codes is provided in Table 6-6.

Table 6-6 IDE hard disk drive error codes

Error code	Name	Description
0	`noErr`	No error detected on the request operation.
–17	`controlErr`	Unimplemented control call. Requested control operation could not complete.
–18	`statusErr`	Unimplemented status call. Requested status operation could not complete.
–23	`openErr`	Unimplemented open call. Open operation could not complete.
–36	`ioErr`	An I/O error detected while processing the request.
–50	`paramErr`	Error in parameter block.
–56	`nsDrvErr`	No such drive. No device attached to the specified port.

continued

Table 6-6 IDE hard disk drive error codes (continued)

Error code	Name	Description
–65	`offLinErr`	No disk in drive (removable media).
–1780	`AT_AbortErr`	Command aborted by drive. Unsupported command.
–1781	`AT_RecalErr`	Recalibrate failure detected by device.
–1782	`AT_WrFltErr`	Write fault detected by device.
–1783	`AT_SeekErr`	Seek error detected by device.
–1784	`AT_UncDataErr`	Unable to correct data (possibly bad data).
–1785	`AT_CorDataErr`	Data was corrected (good data)— notification.
–1786	`AT_BadBlkErr`	Bad block detected by device.
–1787	`AT_DMarkErr`	Data mark not found reported by device.
–1788	`AT_IDNFErr`	Sector ID not found; error reported by device.
–1789	`AT_DRQErr`	Timeout waiting for DRQ active.
–1790	`AT_BusyErr`	Timeout waiting for BSY inactive.
–1791	`AT_NRdyErr`	Drive ready condition not detected.
–1793	`DRVRCantAllocate`	Driver memory allocation error during driver open.
–1794	`NoIDEMgr`	No ATA Manager installed in the system (`MgrInquiry` failure).
–1795	`IDEInitFail`	ATA Manager initialization failed.
–1796	`IDEBufFail`	Power on device test failed. Device failure detected. Interface communication error.
–1802	`IDEMgrNotInitialized`	ATA Manager has not been initialized. The request function cannot be performed until initialized.
–1803	`IDEPBInvalid`	Invalid IDE port address detected (manager initialization problem).
–1804	`IDEFuncNotSupported`	An unknown manager function code specified.
–1805	`IDEBusy`	Selected device is busy. The device isn't ready to go to next phase yet.
–1806	`IDETransTimeOut`	Timeout condition detected. The operation hasn't completed within the user specified time limit.
–1807	`IDEReqInProg`	Device busy; the device on the port is busy processing another command.
–1808	`IDEUnknownState`	The device status register reflects an unknown state.

continued

Table 6-6 IDE hard disk drive error codes (continued)

Error code	Name	Description
–1809	`IDEQLocked`	I/O Queue for the port is locked due to a previous I/O error (must be unlocked prior to continuing).
–1810	`IDEReqAborted`	The I/O queue entry was aborted due to an abort command.
–1811	`IDEUnableToAbort`	The I/O queue entry could not be aborted. Too late to abort or the entry not found.

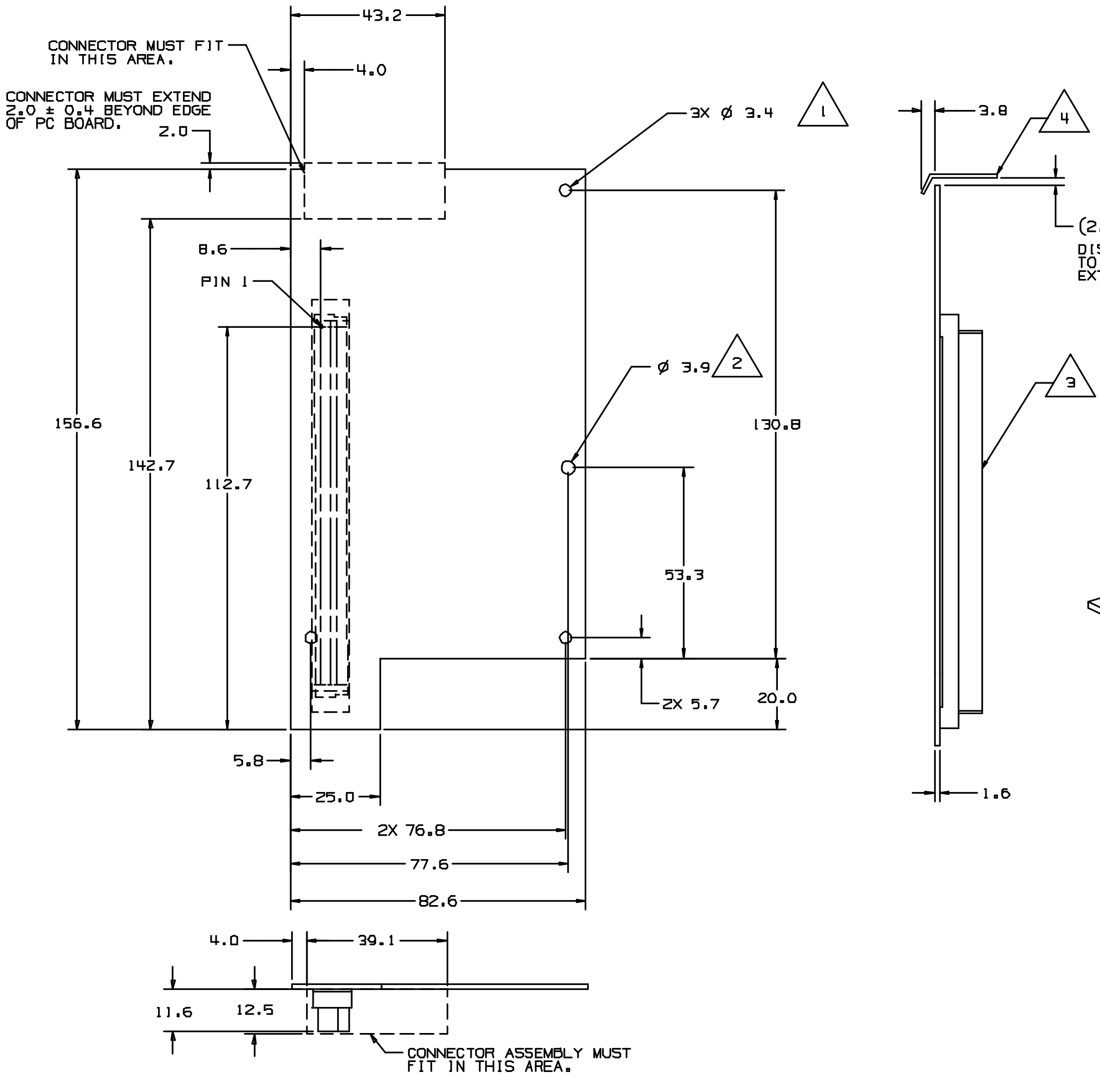
APPENDIX

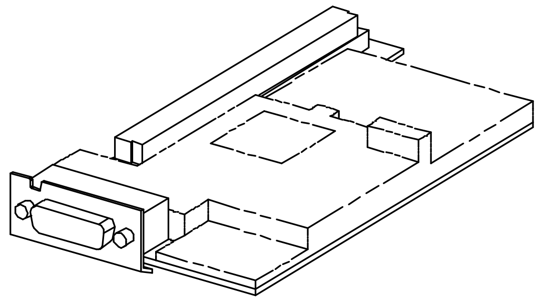
Foldout Drawings

This appendix contains foldout pages with the mechanical drawings for the I/O expansion card. Foldout 1 is a design guide showing the dimensions of the expansion card. Foldout 2 shows the maximum component height permitted on the card. Foldout 3 is an assembly guide for the expansion card.

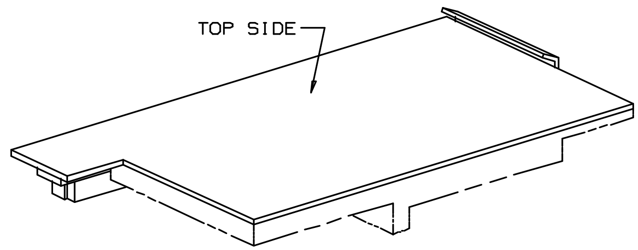
NOTES: UNLESS OTHERWISE SPECIFIED

1. OPTIONAL TOOLING HOLES; IF USED WITH STANDOFF REFER TO APPLE P/N 815-0308.
2. HOLE RECOMMENDED FOR STANDOFF. REFER TO APPLE P/N 815-0177.
3. CONNECTOR, STRAIGHT HEADER : 96-PIN, APPLE P/N 515-0860, COMPATABLE W/ LC FAMILY 120-PIN, APPLE P/N 515-0861, COMPATABLE W/ LCII AND SUBSEQUENT VERSIONS.
4. SHIELD PLATE REQUIRED TO MAINTAIN INTEGRITY OF EMI/RFI SEAM. REFER TO APPLE P/N 062-0489.
5. DO NOT PLACE HOT COMPONENTS IN THIS AREA.



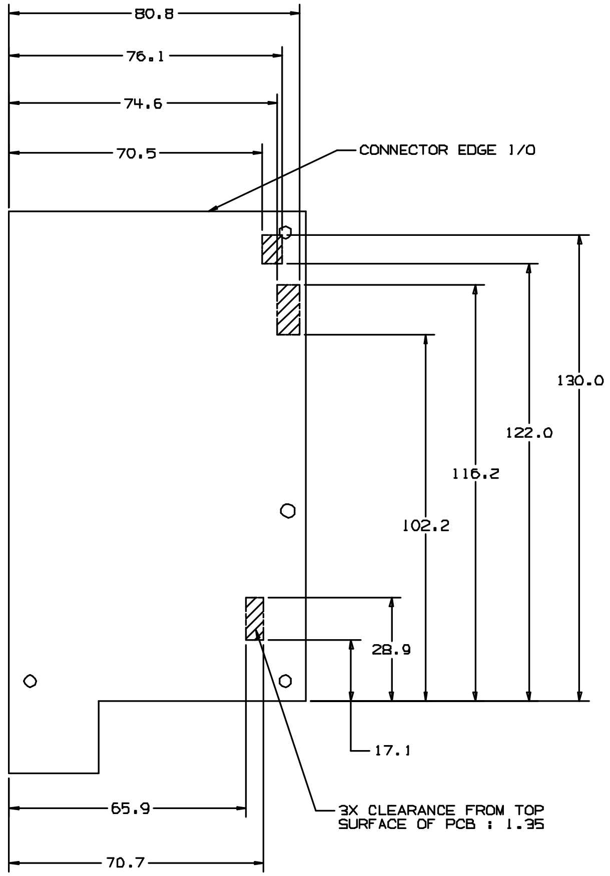


INVERTED ISOMETRIC VIEW



TRI ISO VIEW

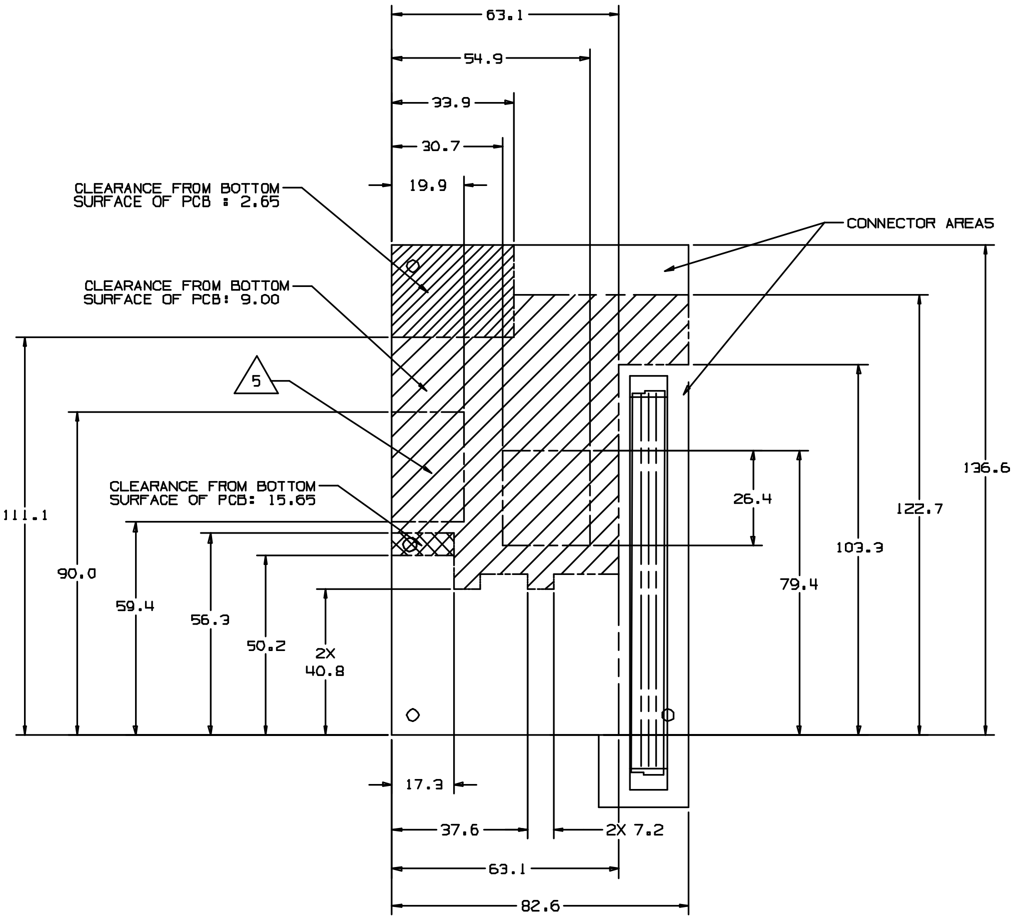
FOLDOUT 1	
Expansion card design guide	
062-0487-B	**1 of 3**



UNLESS NOTED CLEARANCE = 8.0

DETAIL A

TOP VIEW
LEAD HEIGHT RESTRICTION ZONES
SOLDER SIDE OF BOARD

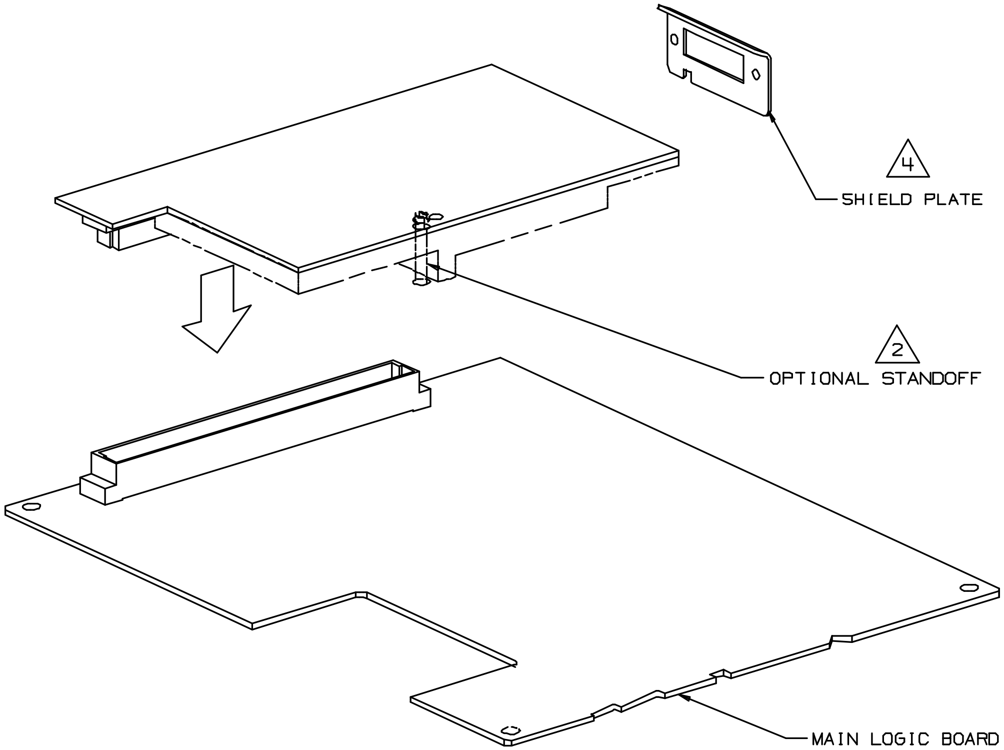


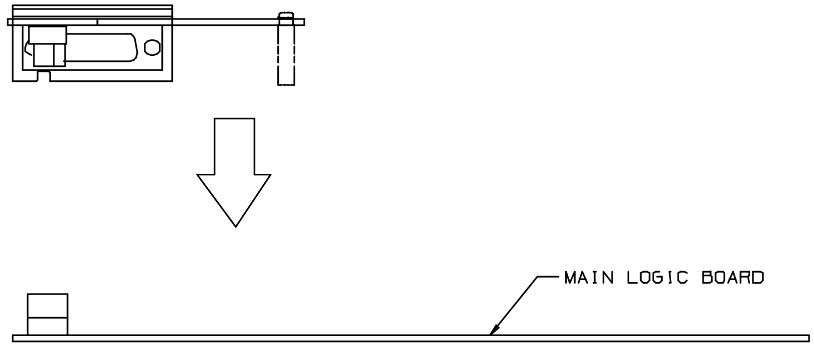
UNLESS NOTED CLEARANCE = 6.25MM

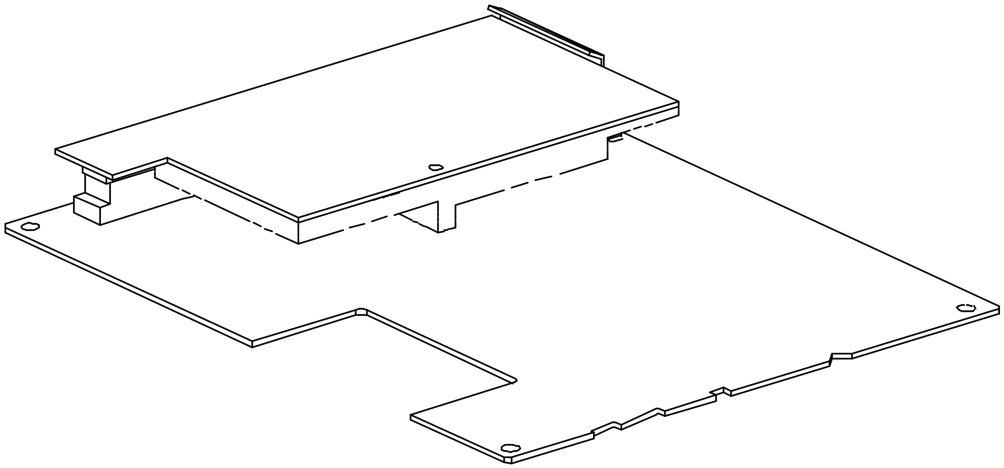
DETAIL B

BOTTOM SIDE
COMPONENT HEIGHT RESTRICTION ZONES
COMPONENT SIDE OF BOARD

FOLDOUT 2	
Expansion card component height restrictions	
062-0487-B	2 of 3







ASSEMBLED VIEW

ASSEMBLED VIEW

FOLDOUT 3	
Expansion card assembly guide	
062-0487-B	**3 of 3**

Index

Numerals

A

B

C

D

E

F

G

H

I, J

S

T, U

V, W, X

Y, Z

This Apple manual was written, edited, and composed on a desktop publishing system using Apple Macintosh computers and FrameMaker software. Proof pages were created on an Apple LaserWriter IINTX printer. Final pages were created on an Apple LaserWriter Pro 630. Line art was created using Adobe™ Illustrator. PostScript™, the page-description language for the LaserWriter, was developed by Adobe Systems Incorporated.

Text type is Palatino® and display type is Helvetica®. Bullets are ITC Zapf Dingbats®. Some elements, such as program listings, are set in Apple Courier.

WRITER
Allen Watson III

COPY EDITOR
Wendy Krafft

ILLUSTRATOR
Shawn Morningstar

Special thanks to Earl Edwards, Linda Frager, John Harding, Sandee Karr, Buay Hock Ng, Phil Nguy, and Steve Schwander