Cortland Custom ICs
Preliminary Notes

Writer: Wayne Lowry
Apple User Education
030-1290-PN8

February 14, 1986

Changes Since Previous Draft

These preliminary notes are the first draft. They are based primarily on the following ERS's:

- Mega // ERS 2.2, June 25, 1985
- Fast Processor Interface (FPI) ERS, September 5, 1985
- Video Graphics Controller (VGC) ERS 2.0, October 8, 1985
- Slot Maker ERS, May 1, 1985
- Keyglu ERS 1.0, June 25, 1985
- Sound Glu ERS, June 21, 1985
- Front Desk Bus ERS Rev B, June 13, 1985

Contents

Forword

These preliminary notes for the *Cortland Custom ICs* manual do not constitute a book. They are merely a collection of notes and fragmentary information and are not guaranteed to be either complete or correct. Their only reason to be is their availability now, so far in advance of the product they describe.

After the developer seeding, the information in these notes will be incorporated into the *Cortland Custom ICs* manual. That book will be the included in another manual (current plan) in the suite of technical manuals for the Cortland.

Chapter 1

Cortland Custom ICs System Overview

Introduction

The Cortland system is divided into two separate subsystems. One subsystem consists of the Mega II, 128K of standard Apple II memory, video generation logic, internal and external I/O slots (right of dotted line, Figure 1). This subsystem is referred to as the Mega II side of the system. The other subsystem consists of the 65C816 microprocessor, the Fast Processor Interface (FPI) IC, up to 4M bytes of fast dynamic RAM, 64K bytes of ROM, and additional internal I/O registers. This subsystem is referred to as the FPI side of the system (lower-left of dotted line, Figure 1).

Operation

The FPI IC is the processor and memory controller for the fast part of the Cortland system. It controls the system speed, manages and refreshes the expanded system memory.

The Mega II IC is the integration and enhancement of several Apple II chips: an Apple II on a chip. Thus, Apple II features are accessible within the Cortland system and all Apple II software is compatible.

System Speed

In normal operation, the Cortland system runs the FPI side of the system at 2.8 MHz and the Mega II side at 1.024 MHz. This allows faster processing without disturbing the standard 1 MHz I/O and video timing that is required for compatibility with existing peripheral units. To run the system optimally, the FPI side of the system provides a *shadowing* feature that allows access to I/O and the Mega II's video buffers.

Shadowing

The Cortland system uses the shadowing method to process the video displays while running programs in the fast RAM area. Write operations that are performed using shadowing write in both the fast RAM and the Mega II RAM, thus the term *shadowing*. Memory read operations, however, are performed only from the fast RAM. Fast RAM is used because the system must slow down to access the slower Mega II RAM.

The Video Graphics Controller

The Video Graphics Controller (VGC) IC supports and enhances existing Apple II video, interfaces with the Macintosh clock chip, provides interrupt processing for three interrupt sources, implements video modes, and provides test modes for chip and board-level testing. The VGC provides these features in concert with the Mega II (upper-right middle of Figure 1).

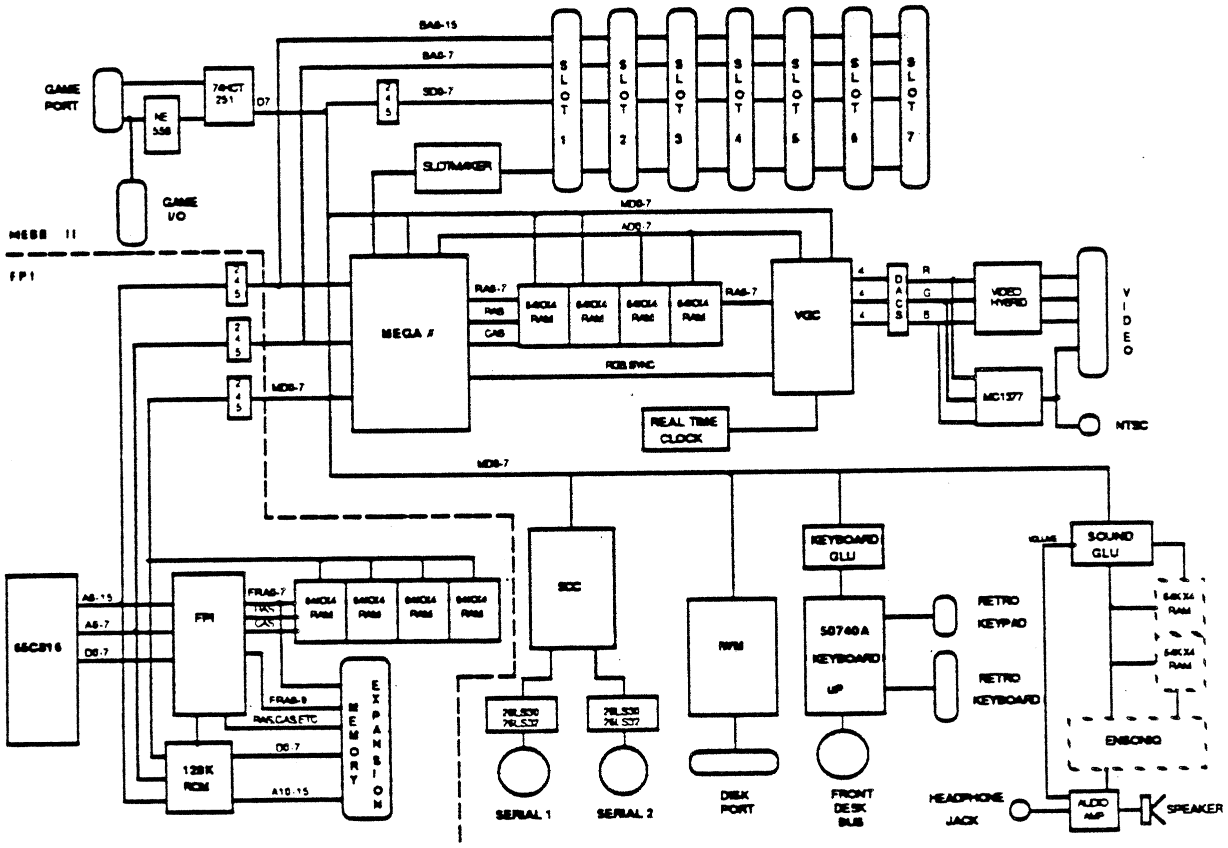


Figure 1. Cortland Block Diagram

Memory Allocation

A minimum Cortland system includes 256K bytes of RAM and 64K bytes of ROM. The Mega II side includes 128K bytes of RAM that corresponds to the current Apple II family Main and Aux memory banks. This RAM is used for text and graphics display buffers and to hold certain system software in reserved areas. The FPI side of the system includes a minimum of 128K bytes of RAM, 64K bytes of ROM, and is expandable to a maximum of 4128K bytes of RAM (see Figure 2).

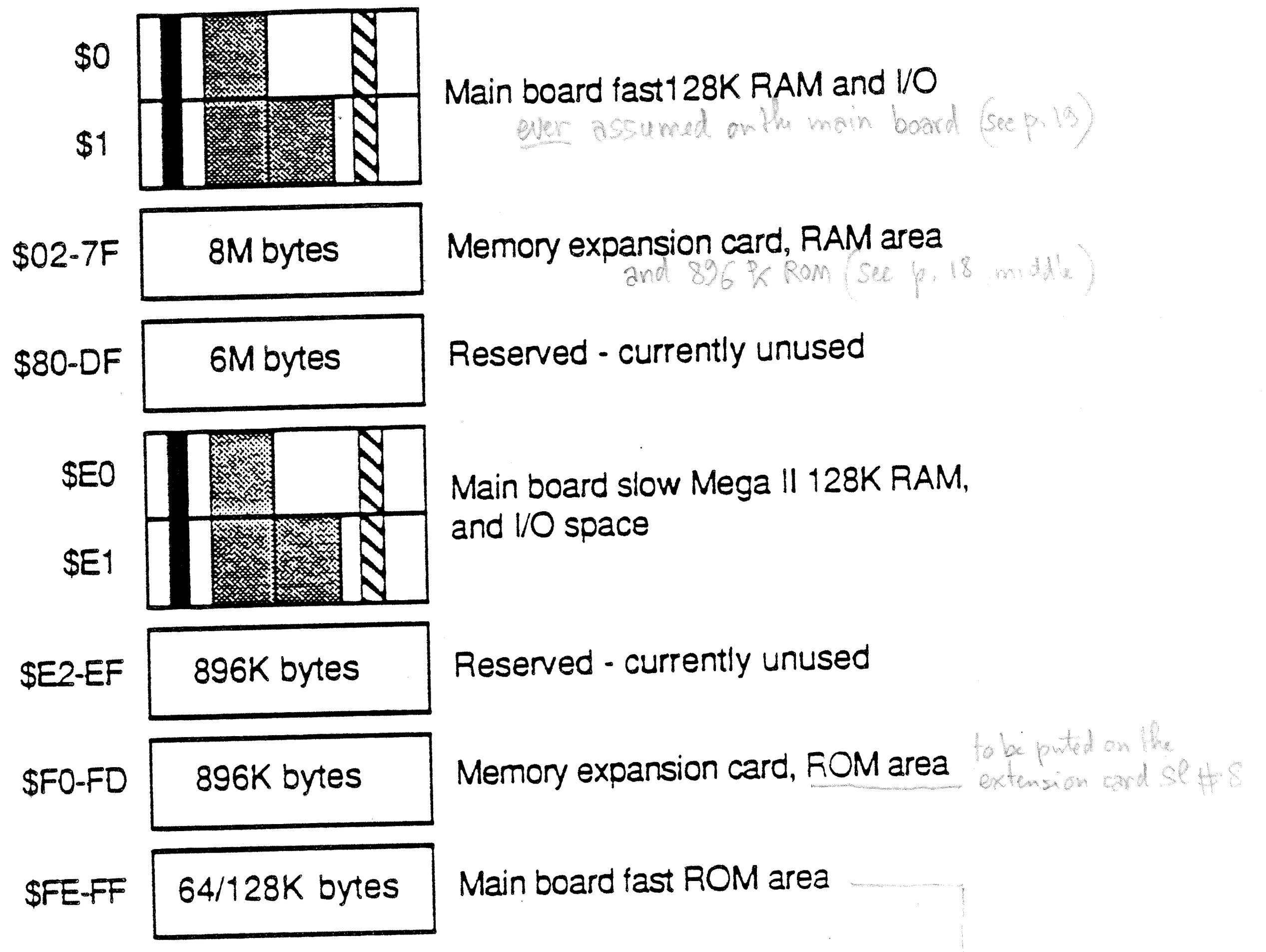

$0
$1
Main board fast128K RAM and I/O
ever assumed on the main board (see p. 19)
$02-7F
8M bytes
Memory expansion card, RAM area
and 896 K ROM (see p. 18 middle)
$80-DF
6M bytes
Reserved - currently unused
$E0
$E1
Main board slow Mega II 128K RAM, and I/O space
$E2-EF
896K bytes
Reserved - currently unused
$F0-FD
896K bytes
Memory expansion card, ROM area
to be puted on the extension card Sl #8
$FE-FF
64/128K bytes
Main board fast ROM area

Figure 2. Bank Memory Map

voir p. 17 : 32 k ROM addition
voir p. 18 : 64 k — on FE

– the on board syst. ROM is on FF bank

– extension card 1M to 4M to be designed with 1x256 or 1M x 1 chips (see p 18, middle)

Chapter 2

Fast Processor Interface

Introduction

The Fast Processor Interface IC (FPI) is the processor and memory controller for the fast part of the Cortland system. It controls the system speed, manages and refreshes the expanded system memory, and provides the *shadowing* that allows access to I/O and the Mega II's video buffers. This sub-system is referred to as the FPI side of the system (see Figures 1 and 3).

During normal operation, the Cortland system runs the FPI side of the system at 2.8 MHz and the Mega II side at 1 MHz. This allows faster processing without disturbing the standard 1 MHz I/O and video timing that is required for compatibility with existing peripheral units.

When it is necessary for the CPU to access an I/O location or the Mega II RAM, the system is briefly slowed to 1 MHz and synchronized with the Mega II timing so that the access can be accomplished. When the access is complete, the FPI side returns to 2.8 MHz operating speed.

Operation Control

The various operating options in the FPI are controlled by a combination of existing Apple II soft switches, Mega II soft switches, and FPI control registers. The control registers include: the Shadow register, the CYA register, and the State register. The following paragraphs describe these options.

Standard Soft Switches

To insure optimal compatibility with existing Apple II computers, the Cortland system includes the standard Apple II soft switches. These soft switches operate in any bank where shadowing is enabled and when the I/O enable bit, in the Shadow register, is set to zero. Most of the soft switches reside in the Mega II and cause the system to slow momentarily when they're accessed. However, some of the soft switches are duplicated in the FPI to provide ROM control, language-card operation in fast RAM, and to provide the information necessary to properly implement video shadowing. The soft switches include: 80STORE, RAMRD, RAMWRT, SLOTC3ROM, INTCXROM, ALTZP, ROMBANK, PAGE2, HIRES, and the internal slot-3 switch. The soft switches are designated *write only* in the FPI and *read/write* in the Mega II. Any access to the individual switches slows the system momentarily.

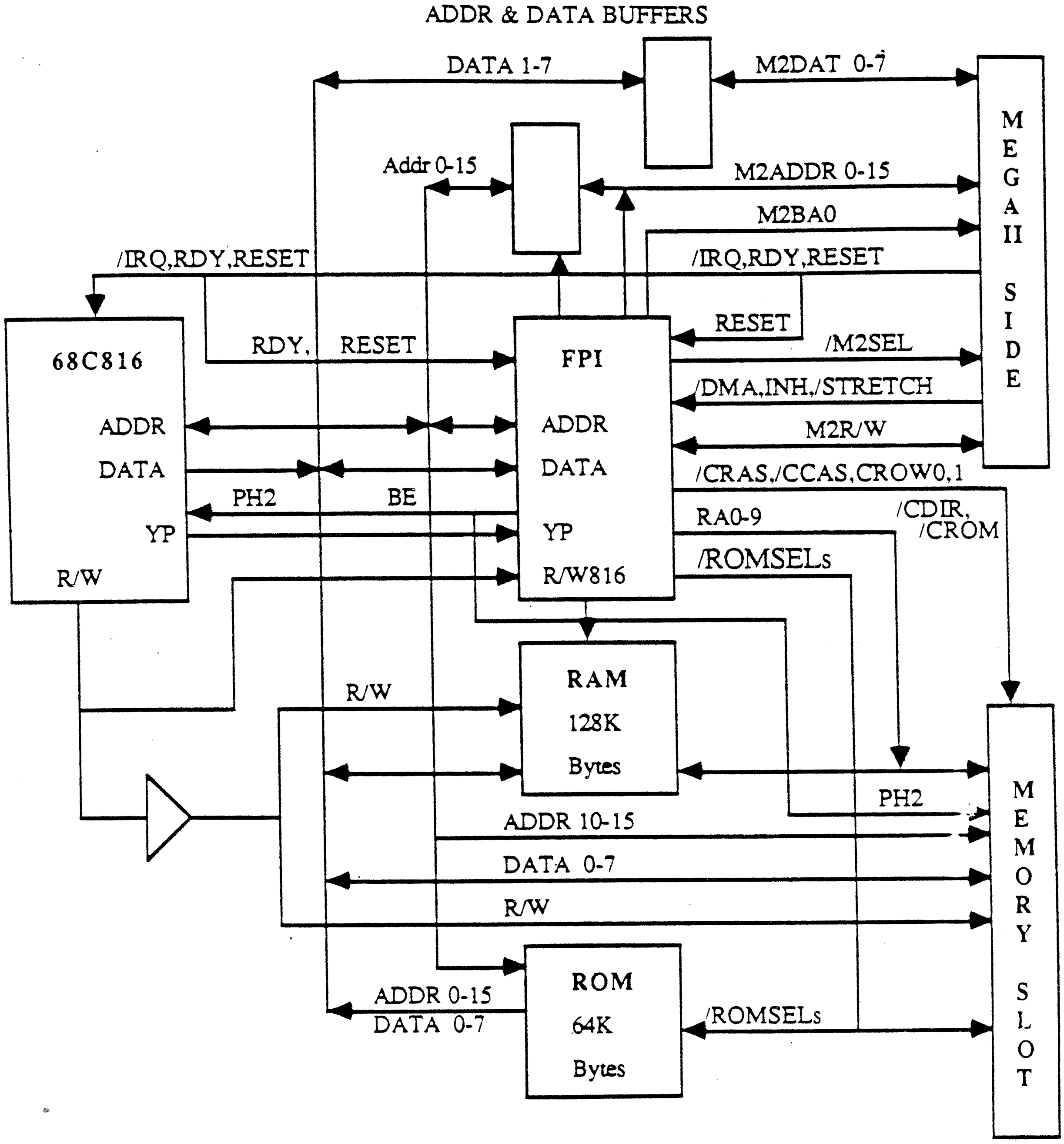


Figure 3. FPI System Block Diagram

Memory Allocation

The FPI side of the Cortland system includes a minimum of 128K bytes of RAM, 64K bytes of ROM, and is expandable to a maximum of 4128K bytes of RAM. This RAM is used for text and graphics display buffers, and holds system software in reserved areas.

The State Register See p 34 chapter 3

The FPI also duplicates the Mega II State register, allowing access to eight of the commonly-used soft switches in a single transfer (see Figure 4). In this case, however, the FPI version of the register is read/write, not write-only. Reading the State register does not slow the system since it is read from the FPI, not the Mega II. Write operations to the State register slows the system momentarily while the write to the Mega II takes place.

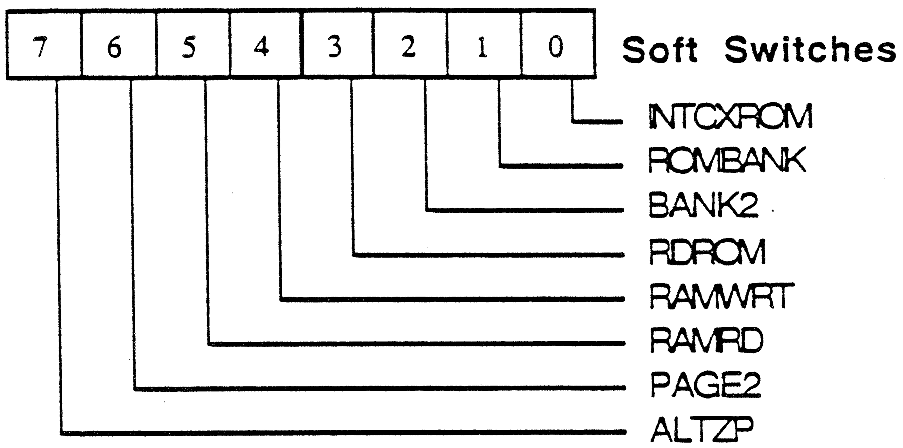


Figure 4. State Register = C068 = FPI and MEGA II

Shadowing

The Cortland uses a method called *shadowing* to handle the video displays while running programs in fast RAM. All write operations to shadowed memory areas are written in both the fast RAM and the Mega II RAM. Memory read operations, however, are normally performed from the fast RAM. Because the system must slow down to access the slower Mega II RAM, using the fast RAM minimizes the impact of display updates on the overall system speed.

The Shadow Register

The Shadow register controls and determines which areas of fast RAM are shadowed into the Mega II RAM display areas. The Shadow register also determines whether or not the I/O space and language-card areas are implemented (see Figure 5).

beware

Shadowing is available in banks 0 and 1, or in any RAM bank where shadowing is enabled by the CYA register. Direct access to the video buffers and I/O is available through Mega II banks $E0-E1.

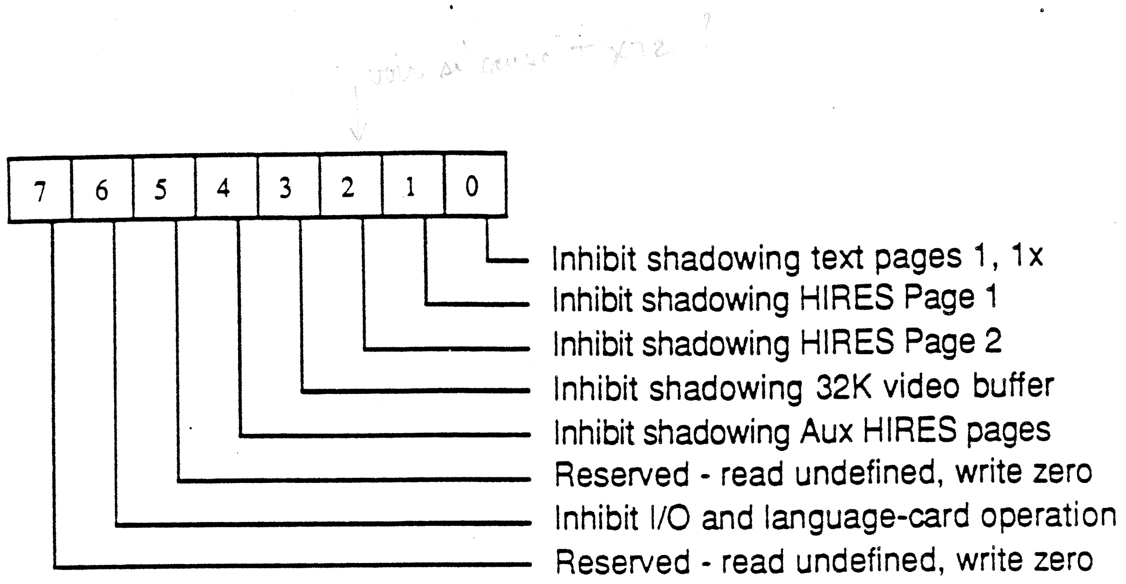

voir si cause + X72 ?
7
6
5
4
3
2
1
0
Inhibit shadowing text pages 1, 1x
Inhibit shadowing HIRES Page 1
Inhibit shadowing HIRES Page 2
Inhibit shadowing 32K video buffer
Inhibit shadowing Aux HIRES pages
Reserved - read undefined, write zero
Inhibit I/O and language-card operation
Reserved - read undefined, write zero

Figure 5. Shadow Register = C∅35 = FPI register

Each bit in the shadow register is active high, meaning that the shadowing of the selected area is inhibited if the appropriate bit is set. The Shadow register is cleared on reset by the FPI so it defaults to shadowing all video areas. Programs that are aware of the Shadow register can turn off shadowing in unused video areas by setting the appropriate bits and therefore reclaiming the memory space in the unused video buffers in Mega II banks $E0-E1.

Within the Shadow register, bits 0 through 3 inhibit shadowing of the display areas indicated in Figure 5. Bit 4 is used with the HIRES shadow bits 1 and 2. If bit 4 is clear, any HIRES display area shadowed will shadow both the main and auxiliary display pages. If bit 4 is set, shadowing the auxiliary HIRES pages is inhibited and only the main HIRES display pages are shadowed when HIRES shadowing is enabled.

> Note: Text page 2 ($800-BFF) is never shadowed. If you need a text display area or a code storage area, use Mega II banks $E0 and E1; however, these banks are limited to 1 MHz operation.

The I/O and language-card inhibit bit (IOLC) controls whether the $Cxxx address range acts as RAM or I/O. If bit IOLC is a zero, I/O is enabled in the $Cxxx space and the 4K of RAM that would normally occupy that space is folded up into a second $Dxxx bank of RAM, forming a language card. If IOLC is a one, the I/O space and language card is inhibited, and contiguous RAM is available from $xx0000 to $xxFFFF.

Shadowed Memory Map

Figure 6 shows banks $0 and $1, or any even-odd pair of banks, if shadowing is enabled in the CYA register.

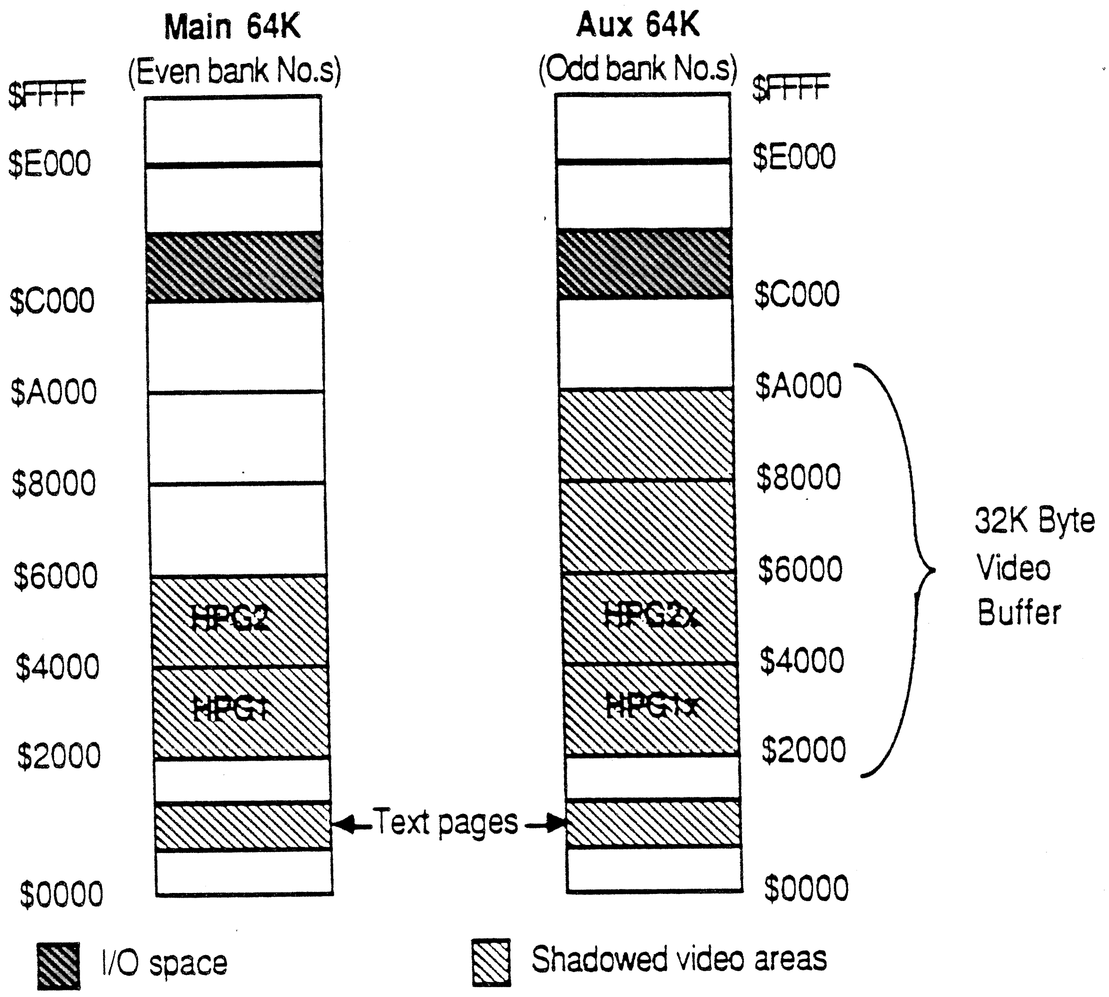


Figure 6. Shadowed Memory Map

The CYA Register where is it!!? ... MOS STATIC RAM ? see the C.PANEL

The Configure Your Apple (CYA) register contains bits that control the speed of operation and shadow enabling in various banks. The CYA register is cleared to zeros on reset or power up (see Figure 5). page 16

(See p19 - chapter 5 hardware designer handbook, end of this) → CYA is Cϕ36 = register FPI

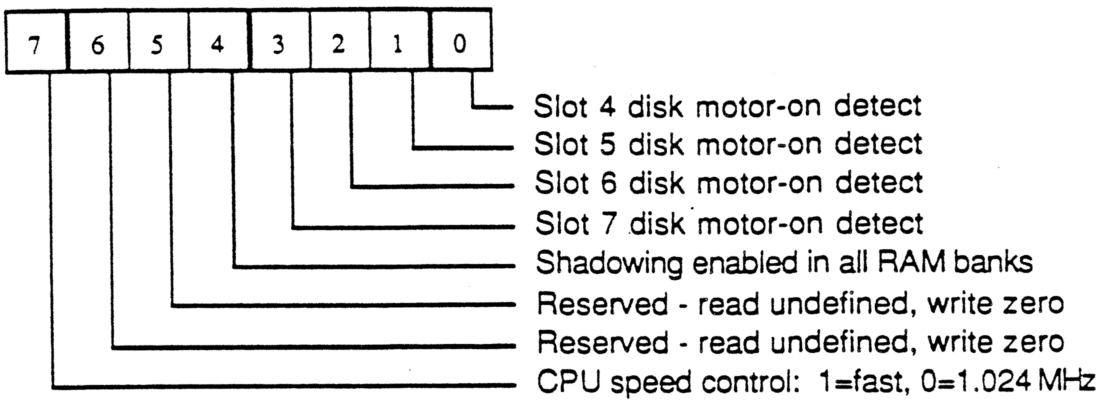

7
6
5
4
3
2
1
0
Slot 4 disk motor-on detect
Slot 5 disk motor-on detect
Slot 6 disk motor-on detect
Slot 7 disk motor-on detect
Shadowing enabled in all RAM banks
Reserved - read undefined, write zero
Reserved - read undefined, write zero
CPU speed control: 1=fast, 0=1.024 MHz

Figure 7. CYA Register = C⌀36

High Speed Operation

The CYA register's bit 7 controls system operating speed. When CYA 7 is set to zero, the system operates at 1.024 MHz (as in an Apple II). When CYA 7 is set to 1, the system operates at 2.8 MHz unless it is slowed by Mega II accesses, distributed RAM refresh cycles, or the disk motor-on detectors.

Disk Motor-On Detectors

Bits 0 through 3 of the CYA register are used to enable the Disk II motor-on/off address detectors in the FPI. Because Disk II, and similar products, require 1.024 MHz operation when accessing the disk, the motor-on detectors slow the system to 1.024 MHz when the disk motor-on address is detected. When the disk motor-off address is accessed, the system speed increases up to 2.8 MHz again. For example, if CYA 1 is set to one, the FPI switches to slow mode (1.024 MHz) when address $C0D9 is accessed, and returns to normal speed (2.8 MHz) following a $C0D8 access.

Note: Drives designed after the Cortland system should use the speed bit (CYA 7) rather than the disk motor-on detectors (CYA 0-2). This allows drives to be installed in slots other than 4-7 and still shift the system speed properly if required. These new designs should also avoid using the motor-on and motor-off addresses unless they are used in a fashion consistant with the drive's CPU speed requirements.

The addresses detected are as follows:

Slot	Motor-on	Motor-off
4	$C0C9	$C0C8
5	$C0D9	$C0D8
6	$C0E8	$C0F9
7	$C0F8	$C0F9

See commentary

RAM Control

The FPI controls dynamic RAM and provides the necessary signals without any external ICs. It supports four 64K x 4 RAMs on the main PC board and up to four rows of RAMs on the extended memory card.

Address Multiplexing

The FPI multiplexes the RAM addresses onto either 8-, 9-, or 10-RAM address lines to provide support for RAM with 64K, 256K or 1M words per RAM. The main-board RAMs (banks 0, 1) are 64K x 4 so the FPI multiplexes the address onto eight lines. The external RAM card can support 256 x 1, 256 x 4, 1M x 1, or 1M x 4 RAMs. The word size of the RAMs on the extended memory card is indicated by the M size output from the card. When M size is strapped low, the FPI provides multiplexing on 9 lines for 256K words per RAM. When M size is strapped high, the address is multiplexed onto 10 lines to support 1M-word RAMs.

later, for 1M Ram chips...

RAM Refreshing

The FPI supports dynamic RAM memory. Dynamic RAMs require periodic refreshing to prevent their internal capacitive memory cells from discharging through internal leakage currents. The FPI provides the refresh cycles necessary to maintain the RAM internal data without requiring any attention from the programmer.

Refresh Requests and Arbitration

!

The FPI RAMs are not transparently refreshed; therefore, the FPI provides periodic refresh cycles to maintain the RAM contents. These refresh cycles occur approximately every 3.5 us. When the FPI is running in fast (2.8 MHz) mode, the refresh cycles reduce the effective speed by approximately 8% for programs running in RAM. Programs in ROM continue to run at full speed because the RAM refresh cycles do not interfere with RAM accesses. When running in slow (1.024 MHz) mode, refresh cycles are executed during an unused part of the processor cycle and do not affect the processor speed.

?!

3.5 %

if (2x) FH → 24% loss

CAS/RAS Refresh

Column Address Strobe (CAS) and Row Address Strobe (RAS) refreshing is used during each refresh cycle (CAS is supplied before RAS). This causes the RAM to use an internal refresh address counter and eliminates the need to supply a refresh address from the FPI. This frees the address bus so that the FPI can execute ROM cycles while RAM refresh cycles are occuring, thus allowing full speed operation in the ROM.

ROM Control

The FPI provides control for 64K bytes of on-board ROM and up to 1M byte of ROM when the memory expansion card is used to hold additional ROM. Apple II series compatibility is provided by supplying 32K bytes of ROM organized as two 16K byte banks that occupy $C100-$FFFF and are enabled in sections as in the Apple II. The Apple II ROM is available only when IOLC shadowing is enabled for the bank being accessed. An additional 32K bytes of ROM is available for tool kits and operating system support in bank $FF.

Note: The Apple II ROM is only available when IOLC shadowing is enabled for the bank being accessed or in the Mega II banks $E0 and E1.

If a 1M-bit ROM is used on the main board, 128K bytes of ROM is available with an additional 64K bytes in bank $FE.

Banks $F8 and $FE-FF

Bank $FF holds the on-board system ROM. The 32K bytes of Apple II ROM are mapped into $FF8000-$FFFFFF. Apple II ROM bank 1 maps into $FFC000-$FFFFFF while ROM bank 2 maps into $FF8000-$FFBFFF. An additional 32K bytes of ROM is mapped into $FF0000-$FF7FFF. This ROM is accessible through bank $FF.

When the 1M-bit ROM is used on the main board, an additional 64K bytes of ROM occupies $FExxxx.

Banks $F0-$FD are reserved for ROM system expansion. The ROM to occupy this area resides on an extended memory card along with additional system RAM.

896 → en FØ/FD
64 → en FE
960K

Extended Memory Card Slot

interface forbidden in "SP#8"!!

The extended memory card slot allows you to add memory cards holding up to 8M bytes of RAM and 896K bytes of ROM memory. It supports additional memory only and is not to be used for any other purpose. One M-byte or 4M-byte RAM cards can be constructed by using 256K x 1 or 1M x 1 RAMs.

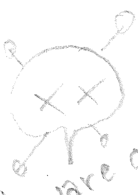
Extended RAM

Up to 4M bytes (64 banks of 64K bytes) of RAM can be included on the extended memory card. However, this requires 1M x 1 or 256K x 4 RAMs. The card supports up to four rows of RAMs with each row holding either 256K bytes or 1M byte.

To control and select individual rows of RAM, the FPI provides /CRAS, /CCAS, CROW0, and CROW1 signals. Signals /CRAS and /CCAS are the basic memory timing signals common to most dynamic RAMs. Signals CROW0 and CROW1 are row selects which, when taken as a pair, indicate the row number to be accessed. Typically, CROW0 and CROW1 are used as the select signals for a dual 4-1 decoder (74F139 or equivalent) that demultiplexes /CRAS and /CCAS into a separate /RAS and /CAS for each row.

just the same as the piggy back board for Rom on the main circuit.

Extended RAM Mapping



beware of the problem!!

Figure 8 depicts a 1M-byte extended RAM card using four rows of 256K bytes per row. The RAM banks above bank $11 are *ghosts* (repeat images) of the RAM in banks $2-F and $10-11. A partially populated card causes holes in the memory map unless there is an option on the card to alter the address decoding. For example, if the card described above were populated with rows 0 and 1 only, there would be 512K bytes of contiguous memory in banks $0-7 and another 128K bytes in banks $10-11. Banks $8-F would be empty.

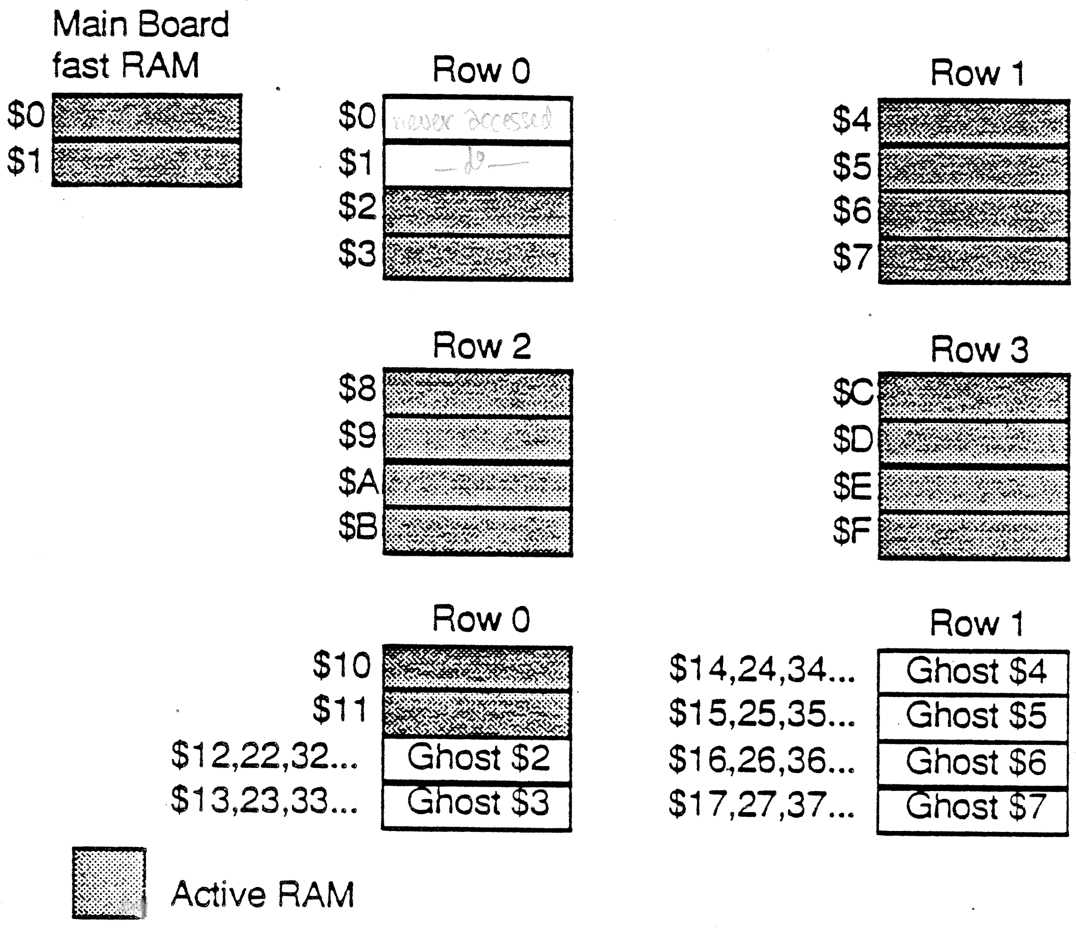


Figure 8. Extended RAM Mapping

MSIZE (= JUMPER)

A special input pin called MSIZE connects to the extended memory slot and is either strapped to ground on the memory card or left open.

If the MSIZE pin is connected to ground (for 256K bit RAMs), the FPI multiplexes 18 address bits onto RA0-8 and generates the CROW0-1 row selects for rows of 256K bytes. If the MSIZE pin is left open (for 1M-bit RAMs), the FPI multiplexes 20 address bits onto RA0-9 and generates the row selects for 1M-byte rows.

Ghosting

The FPI always enables the extended RAM card for accesses in banks $2-$80. Rows of RAMs on the card are selected by CROW0 and CROW1. For a 1M-byte card with 256K byte rows (MSIZE=0), the selected RAM row number is given by the bank number mod 4. The only exception is for banks $0-$1, which are always assumed to be on the main board; therefore, the extended memory card is not accessed. This method of card and row selection causes multiple images or *ghosting* of the RAM areas on the card.

Extended ROM

An extended ROM card can support up to 896K bytes in banks $F0 to $FD. This capability requires an additional bank address latch-decoder on the memory card. The FPI provides a signal (CROMSIL.L) that is a decode of one bank; however, the card must provide the additional decoding to select individual ROMs within the selected space. This can be accomplished, for example, with a 74HC259 addressable latch. The lower three data lines are connected to the 74HC259 address inputs, CROMSEL.L connects to the latch data input, and the PH2 clock is used to enable the latch. This one IC now decodes and demultiplexes the bank address in the range of interest. Each latch output can be used to chip-enable a 27512 ROM or a 64K x 8 equivalent. Similar methods can be used to select other sizes of ROMs.

The extended memory card connector provides a group of signals to support dynamic RAM and additional general purpose signals to support ROM decoding and selection. Table 1 lists these signals.

Table 1. Memory Card Interface Signals

Signal	Description
FRA0-FRA9	10 bits of multiplexed RAM address for RAM cycles – least significant 10 bits of ROM address.
CROW0,1	2 bits select 1 of 4 RAM rows.
CRAS.L	RAM /RAS strobe.
CCAS.L	RAM /RAS strobe.
FR/W	Write enable to RAMs. R/W from CPU or DMA.
D0-D7	8 bits of bidirectional data – CPU data bus.
CDIR.L	Card data buffer direction control. High to read card data.
CROMSEL.L	Card ROM select. Low for accesses to banks $F0-FD.
PH2CLK	CPU clock. Rising edge indicates valid bank address on D0-D7.
MSIZE	Output from card. Indicates RAM row size.
A10-A15	The 6 high-order addresss bits. Used with ROMs.
14M	14 MHz clock signal.
VCC	+5v +/-5%. 600 ma absolute maximum.

To control and select individual rows of RAM, the FPI provides the /CRAS, /CCAS, CROW0, and CROW1 signals. The /CRAS and /CCAS signals are for basic memory timing common to most dynamic RAMs. The CROW0 and CROW1 are row select signals which , when taken as a pair, indicate the row number to be accessed. Typically, CROW0 and CROW1 are select signals for a dual 4-1 decoder (74F139 or equivalent) that demultiplexes /CRAS and /CCAS into a separate /RAS and /CAS for each row. Figure 9 shows a typical circuit for RAM row selection.

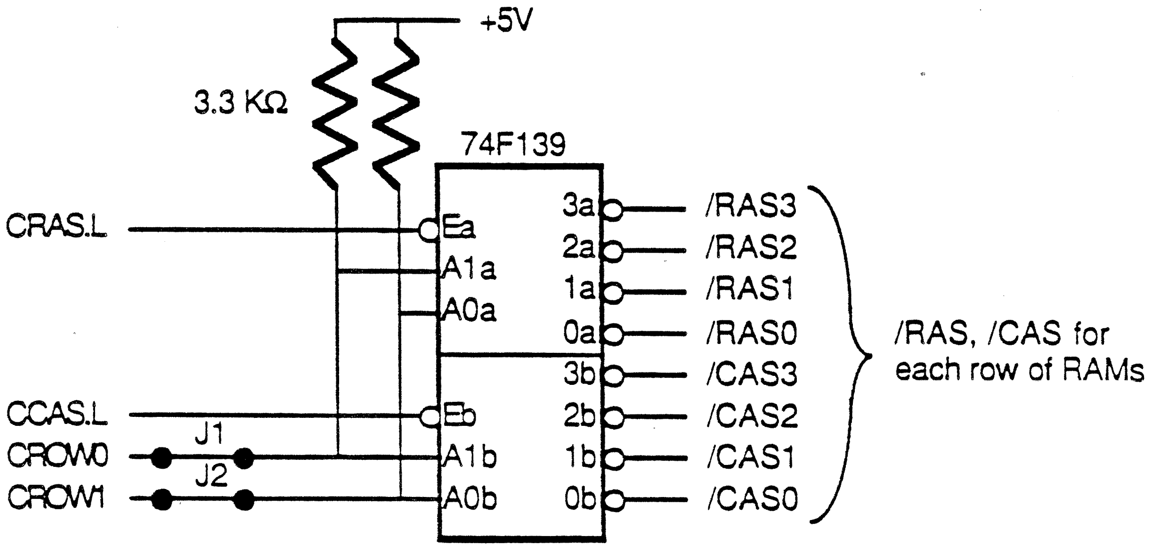


Figure 9. Extended Memory Card RAS/CAS Decoding Example

I/O Processing

Normally, all I/O accesses are in the designated I/O space and are sent to the Mega II as well as the FPI. The only exceptions are when the unique FPI internal registers are accessed, and when the interrupt ROM area is read. The unique FPI registers include the DMA bank register, the CYA register, and the Shadow register.

The FPI registers that are duplicates of Mega II registers include the State register and the Slot ROM register. Write commands to these registers are also sent to the Mega II. Read commands are from the FPI only.

The interrupt ROM segment is always available when both the shadowing and IOLC are enabled in the Shadow register, and is not affected by the ROM bank soft switch.

The Slot ROM Register

The Slot ROM register is used to select either the internal I/O or the I/O slot device for each logical slot. If the enable bit in the Slot register is set to a one, accesses to that slot's ROM space ($Cnxx) are sent to the Mega II and from there to the physical I/O slot. If the enable bit is cleared, $Cnxx read operations from that slot's ROM space are processed by the on-board ROMs that control the built-in I/O devices.

I/O real or phantoms

Slot 3 is a special case. Like slot 4, its hardware addresses are always available. However, its ROM space is controlled by the SLOTC3ROM switch to maintain compatibility with the existing Apple II products.

SP #3 !!

Mega II Interface

The FPI controls the transfer of data to and from the Mega II. When it is necessary for the CPU to communicate (read or write) with the slow side of the system, the FPI holds the CPU PH2 clock high and waits for the beginning of the next Mega II cycle. Shortly after the cycle starts, the FPI asserts the Mega II select line and enables the buffers to drive the CPU address onto the Mega II's address bus (the Mega II R/W line is also driven at this time). When the Mega II's PH0 clock rises, the data-bus drivers are enabled and the data transfer takes place. The cycle is complete when the PH0 falls and the FPI drops the CPU PH2 clock low at that time. In consecutive Mega II cycles (slow mode), the FPI generates one CPU cycle for each Mega II. cycle, thus running the CPU at precisely 1.024 MHz to support time-dependent code.

M-State Counter

To provide correct synchronization with the slow side of the system, the FPI uses an internal counting circuit called the M-State Counter that is kept synchronized with the Mega II. The M-State counter simply counts 14M clocks during PH0 cycle time. The counter outputs are used internally in the FPI to control the timing of the various signals that interface with the Mega II.

Synchronization

Correct synchronization with the Mega II is provided by a signal called STRETCH.L, rom the VGC. Signal STRETCH.L is asserted during Mega II's count $D every 65 Mega ! cycles, and coincides with the stretched CPU cycle common to all Apple II products.

Signal STRETCH.L is held low for two clock cycles and serves two purposes within the FPI. Initially, when the FPI is not yet synchronized after power up, STRETCH.L forces the FPI M-State counter to a $D count to provide the initial synchronization with the Mega II. In normal operation, after the system is synchronized, STRETCH.L stretches the FPI M-State counter cycle by holding it at a $D count when the Mega II cycle is stretched, thus maintaining synchronization throughout the stretched cycles.

Mega II Cycles

The Mega II cycles are CPU or DMA cycles that access the slow side of the system. These include all external and most internal I/O, shadow video write operations, *inhibited* memory or I/O accesses, and all accesses to the Mega II banks.

$E0-$E1

A Mega II cycle typically begins when the FPI recognizes an address that requires access to the slow side of the system. Approximately 90 ns after the CPU PH2 clock falls, the address and bank address from the CPU becomes valid. These are decoded internally in the FPI and the type of cycle to execute is determined before the PH2 clock rises. If the cycle is a Mega II cycle, the FPI generates an internal "slow" signal that holds the PH2 clock high and holds RAS and CAS inactive. The FPI remains in this state until it reaches the end of M-State 1 in the Mega II PH0 timing. On the next rising clock edge, the FPI sets its internal SYNC flag to indicate that it is now synchronized with the Mega II timing.

Mega II Bank Bit

To allow direct access to the Mega II auxiliary bank, the FPI passes the lsb of the bank address to the Mega II during M-States 2 and 3 of each Mega II cycle. If shadowing is enabled or the software is accessing bank $E0-E1, access to an odd-numbered bank will access the Mega II auxiliary memory without requiring use of the soft swtches. For this to work, the programmer must set bit 0 to 1 in the video control register at $C029. Otherwise, the Mega II will ignore the bank bit and the soft switches must be used to access the auxiliary 64K through an even-numbered bank.

DMA

DMA is supported with a full 24-bit address range; this is accomplished using a DMA bank register that can be software-loaded to provide the upper eight bits of the 24-bit address required.

During DMA cycles, when the DMA signal is asserted by a slot device, the address bus is turned off within 30 ns and left off until the CPU PH2 clock goes high and the bank address has been latched. At this time, the address bus is enabled, pointing "in" toward the FPI and 65C816. The FPI decodes the address and stored DMA bank address to determine whether the cycle is to RAM, ROM, or Mega II. If the cycle is a DMA to the Mega II (or slots) , the Mega II select line is asserted by the FPI and the FPI data buffers are turned off if R/W is high. If the cycle is to the fast RAM, the data buffers are enabled while PH0 is high; the RAS and CAS signals are generated at appropriate times.

> Note: To increase read/write data timing margins to the fast RAMs, the FPI generates an early CAS for read cycles and a late CAS for write cycles. This provides earlier read data available and less required write data setup time.

Timing Control

The FPI uses two timing speeds in its various functions. These speeds are controlled by two sequencers: the F-State machine and the P-State machine.

F-State Machine

The F-State machine is a five-state sequencer that generates the RAS and CAS timing signals for the fast RAMs. It includes circuitry to hold it in various states so that the timing can be synchronized with the Mega II, and the RAS/CAS timing can be varied for read and write cycles.

P-State Machine

The P-State machine is a four-state sequencer that generates the 65C816 PH2 clock and the internal FPI quadrature Q-clock. The P-State machine is synchronized with the F-State machine in such a way that the PH2 clock is always low for two clock cycles. This always coincides with the minimum two-clock precharge period when both RAS and CAS are inactive. For normal fast RAM cycles, PH2 goes high at the same time RAS goes active (low). However, PH2 can be held high while RAS and CAS are low if, for example, DMA cycles are going on, or during the slow cycles when PH0 is low and a memory refresh cycle is occuring. The result of this synchronization is that the minimum PH2 cycle is five clocks long, two clocks low and three clocks high for a total of 350 ns.

Also, for slow cycles, the PH2 clock is held high until the F-State sequencer comes into synchronization with the Mega II PH0 clock and the slow cycle is completed. These slow cycles can range from a minimum of 14 clocks to a maximum of 29 clocks in cases where there is a maximum synchronization latency that coincides with the Apple II *stretched* cycle. An average cycle time for isolated slow cycles is 1.5 us. Consecutive slow cycles run at 1.024 MHz.

Interfacing the Cortland System

The input and output functions are made possible by the use of pluggable, programmed I/O cards and DMA cards.

Programmed I/O Cards

Most I/O cards used in the Apple II also work in the Cortland system. Specifically, cards that use the IOSEL and DEVSEL bus signals will not be confused by the larger address range in the Cortland system.

The CPU operates with a 24-bit address; however, the I/O slots only get a 16-bit address. Therefore, cards that try to decode a 16-bit address instead of using the DEVSEL and IOSEL signals will not work properly. These cards include the multi-function I/O cards that emulate I/O cards in several slots, and most add-on RAM cards. Fortunately, these cards, in general, are not used for the Cortland system because of the extensive built-in I/O and the fast RAM expansion capabilities.

Cards that use INHIBIT will work properly if a) the system is running slowly, and b) they assert inhibit within 200 ns of the time PH0 falls. However, compatibility with this type of card must be determined on an individual basis because many Monitor calls execute code in bank $FF and the card does not receive bank information. The FPI generally ignores inhibits that occur when the system is running fast, or when it is not in a bank where I/O and language-card operation is enabled. This improves compatibility with existing cards.

DMA Cards

Many DMA cards work in the Cortland system but may require changes in their firmware or associated software to function properly with the DMA bank register. In general, DMA cards that assert/remove the DMA signal within the first 200 ns of the PH0 cycle will probably work properly as this allows sufficient time for the Mega II select line to be activated by the FPI for video and I/O accesses.

DMA

Note: Normally the system should be running slowly when performing DMA, otherwise, DMA to I/O or Mega II video areas will not work properly. However, DMA can be perfomed while the system is running fast as long as the following constraints are observed:

- Fast DMA can only access fast RAM or ROM (access to I/O, video, or the Mega II banks do not work properly).
- Fast DMA may cause a repeated cycle to occur to the location currently being accessed by the processor. This could cause a malfunction if the CPU is accessing I/O when the DMA occurs; however, a repeated access to a RAM location will have no effect. The 65C816 can be stopped indefinitely for DMA and does not require any CPU refresh cycles from a DMA card.

Chapter 3

Cortland Mega II

Introduction

The Mega II is the integration of several enhanced Apple II chips. The following chips comprise the Mega II:

- MMU Custom Chip
- IOU Custom Chip
- Character Generator ROMs (8 languages)
- TMG Timing Generator
- GLU General Logic Unit
- Video Logic

The Mega II, shown in Figure 10, has virtually all the characteristics of an Apple II on a chip; it supports a slotted architecture and has built-in peripherals.

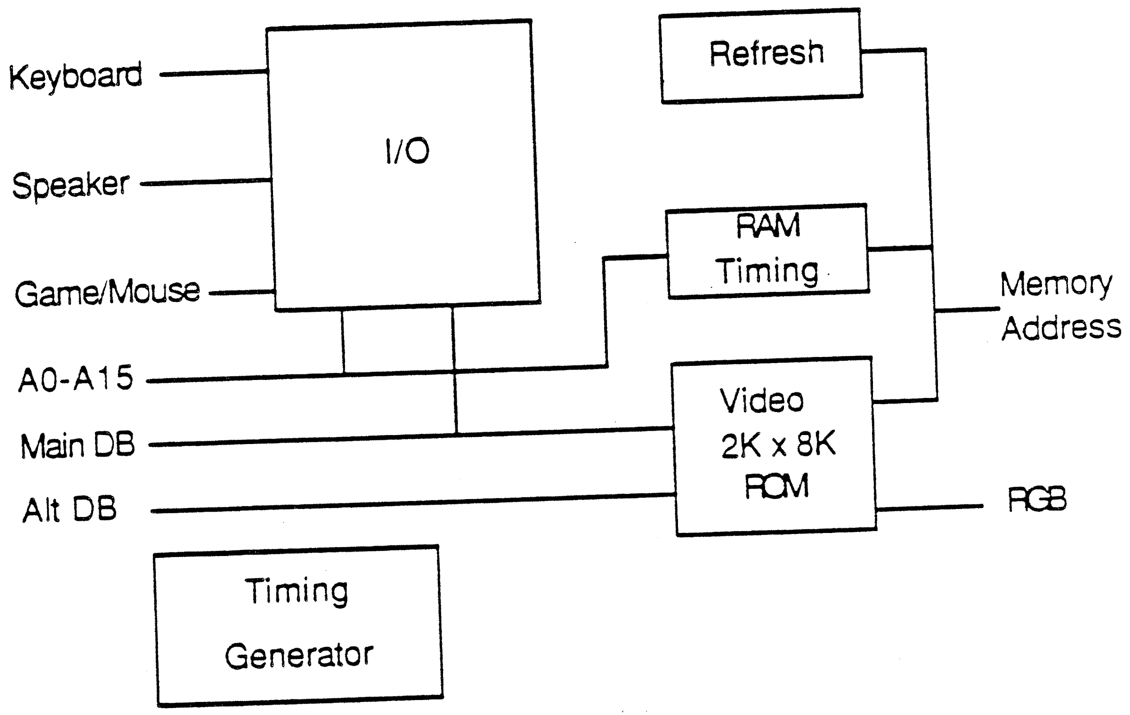


Figure 10. Mega II Block Diagram

Mega II Select Functions

The Mega II supports I/O slots and built-in peripherals. In addition, the 16-pin Game I/O connector is supported by using an external chip referred to as the Slot Maker (see Chapter 5).

Support of Internal Devices

The Mega II supports built-in peripherals and traditional slots. This allows peripherals, such as a serial port, to be built on the main logic board; however, the internal peripherals are disabled if a card is plugged into the corresponding slot. The Mega II disables the internal peripherals using the Slot ROM Select register at location $C02D. Register $C02D functions as follows:

D7	When this bit is 0, the internal ROM is selected. When this bit is 1, the Slot ROM (location $C700 to $C7FF) is enabled and device space $C0F0 to $FF is enabled.
D6	When this bit is 0, the internal ROM is selected. When this bit is 1, the Slot ROM (location $C600 to $C6FF) is enabled and device space $C0E0 to $EF is enabled.
D5	When this bit is 0, the internal ROM is selected. When this bit is 1, the Slot ROM (location $C500 to $C5FF) is enabled and device space $C0D0 to $DF is enabled.
D4	When this bit is 0, the internal ROM is selected. When this bit is 1, the Slot ROM (location $C400 to $C4FF) is enabled.
D3	Not used, therefore it must be set to zero.
D2	When this bit is 0, the internal ROM is selected. When this bit is 1, the Slot ROM (location $C200 to $C2FF) is enabled and device space $C0A0 to $AF is enabled.
D1	When this bit is 0, the internal ROM is selected. When this bit is 1, the Slot ROM (location $C100 to $C1FF) is enabled and device space $C090 to $9F is enabled
D0	Not used, therefore it must be set to zero.
Note:	Device space for slots 3 ($C0C0 to $CF) and 4 ($C0D0 to $DF) are always enabled. On the Mega II, this register is forced to all zeroes.

RAM Refresh

The Mega II performs 256 refresh cycles every 3.25 milliseconds. That is, five refresh cycles are performed every 63.5 microseconds.

Figure 11 depicts the phase 1 (PH1) bus activity. During the unused memory cycles, the RAM address outputs (RA0-RA7) are the last refresh addresses. During this time, RAS and CAS signals are inactive (high).

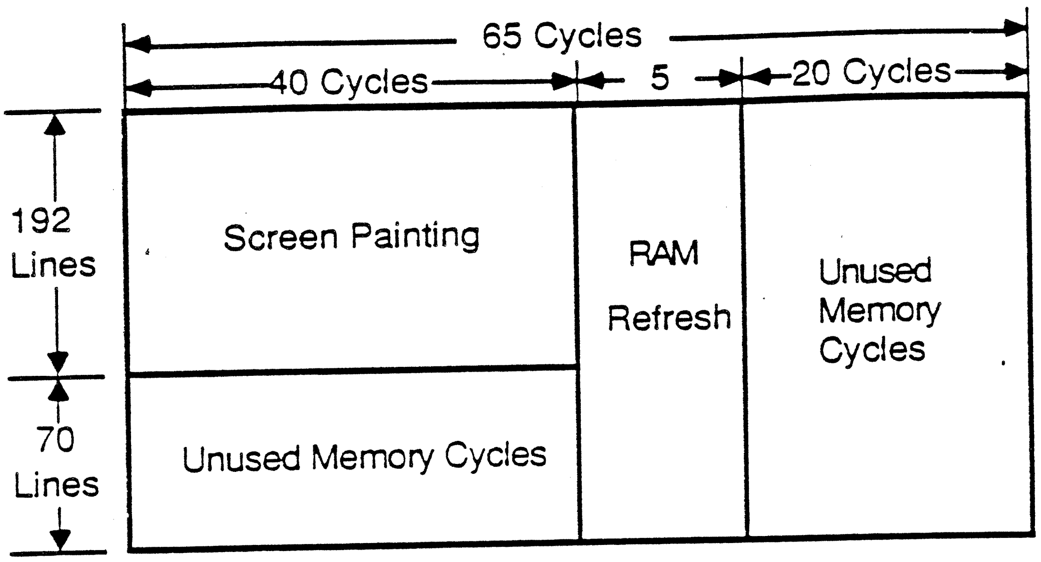


Figure 11. PH1 Bus Activity

Mouse/Game I/O Operation

The mouse interface consists of two eight-bit up/down counters; one for the X, or horizontal, direction and one for the Y, or vertical, direction. These counters unburden the CPU from mouse-movement interrupts and they provide a passive mode (noninterrupt) that is software compatible with the Apple II.

The counter counts up when the mouse movement is down/right, and it counts down when the mouse movement is up/left. The counter is reset after it is read thus providing a delta count to the software. The counters do not overflow in either direction. If the mouse changes direction after a maximum count is reached, it does not count in the opposite direction. If interrupts are not enabled, then at the maximum rate the counter reaches the maximum count in 25 msec, or about 5,000 instructions.

Interrupts

There are three sources of interrupt from the mouse interface that can be masked . These are: mouse movement, VBL, and a pressed button. The mouse movement and button-pressed conditions generate an interrupt on the next VBL after the event occurs. All interrupt-enable information for the mouse is contained in the Mouse-Interrupt-Enable register.

Game I/O

The Mega II supports paddles 0, 2, and 3, and switchs 0, 1, and 2.

Interrupt ROM

The Mega II maintains 15 bytes of ROM to support the interrupt handler. These bytes of ROM are located in $C071 to $C07F. When this space is accessed, the /ROMEN signal goes to an active state.

Video

The Mega II provides a serial video output and an RGB that are similar to the Apple II.

Composite (NTSC)

A serial video stream (SERVID) is multiplexed onto the color gate (CLRGAT) signal. When the serial information is combined with signals CREF and SYNC, Mega II generates the same NTSC signal as the Apple II. The serial video information is valid only during the time the NTSC signal is low.

RGB

The RGB outputs are directly compatible with Apple II's video modes, and Apple's RGB monitor (monitor 100). These outputs are TTL, not analog. A linear-weighted sum of these four signals forms a 16-color RGB video output signal or a gray-scale black and white signal. The full 16 colors are available only in LORES, MERES, and DHIRES graphics modes. The HIRES graphics mode can display only the colors black, purple, medium blue, orange, green, and white (the standard HIRES colors). Table 2 lists the RGB color codes.

Table 2. Color Codes

RGB Output				Color
8	4	2	1	
0	0	0	0	Black
0	0	0	1	Magenta
0	0	1	0	Dark Blue
0	0	1	1	Purple
0	1	0	0	Dark Green
0	1	0	1	Gray 1
0	1	1	0	Medium Blue
0	1	1	1	Light Blue
1	0	0	0	Brown
1	0	0	1	Orange
1	0	1	0	Gray 2
1	0	1	1	Pink
1	1	0	0	Green
1	1	0	1	Yellow
1	1	1	0	Aquamarine
1	1	1	1	White

Text

The Mega II's character-generator ROM contains characters for eight different languages. The languages are selected by writing in the language select register, located at $C02B. Table 3 lists the code for each language.

See p. 32

Table 3. Character Generator Language Codes

D7	D6	D5	Primary Language	Secondary Language
0	0	0	English (USA)	Dvorak
0	0	1	English (UK)	English (USA)
0	1	0	French	English (USA)
0	1	1	Danish	English (USA)
1	0	0	Spanish	English (USA)
1	0	1	Italian	English (USA)
1	1	0	German	English (USA)
1	1	1	Swedish	English (USA)

Bit D4 is the PAL/NTSC bit; the video counters count 262 scan lines when this bit is 0 and 312 scan lines when it is a 1. This option generates the correct frame rate of 50 or 60 HZ for PAL or NTSC monitors. To get the PAL output, a different crystal from the NTSC crystal must be used.

Bit D3 is the language-switch bit that replaces the mechanical switch on the Apple II. When this bit is a 1, the primary language character set is selected.

Bits D2 through D0 are reserved and must be programmed with zeros.

The Mega II returns text in black and white only.

Either of two character sets can be displayed by Mega II: the primary set and an alternate character set (see Table 4). Each character on the screen is stored as one byte of display data. The low-order six bits make up the ASCII code of the character being displayed; the remaining two high-order bits select the format and group within ASCII.

Table 4. Character Sets

Primary Character Set			Alternate Character Set	
Hex Value	Character Type	Format	Character Type	Format
$00-$1F	Uppercase letters	Inverse	Uppercase letters	Inverse
$20-$3F	Special characters	Inverse	Special characters	Inverse
$40-$5F	Uppercase letters	Flashing	MouseText	
$60-$7F	Special characters	Flashing	Lowercase letters	Inverse
$80-$9F	Uppercase letters	Normal	Uppercase letters	Normal
$A0-$BF	Special characters	Normal	Special characters	Normal
$C0-$DF	Uppercase letters	Normal	Uppercase letters	Normal
$E0-$FF	Lowercase letters	Normal	Lowercase letters	Normal

The Mega II character-generator ROM can display 32 uppercase letters, 32 lowercase letters, 32 special characters, 32 MouseText characters, and 10 unique characters for each of the eight languages. See character-generator ROM in Appendix A.

Graphics

All Apple II graphics modes are supported by Mega II including LORES, MERES, HIRES, and DHIRES. Table 5 is a list of the video modes.

Table 5. Mega II Video Modes

C029 D5	AN3	TEXT	HIRES	80COL	Video Mode
X	X	1	X	0	40-column text
X	X	1	X	1	80-column text
X	1	0	0	0	LORES mix with 40-col text
X	1	0	0	1	LORES mix with 80-col text
X	0	0	0	1	MERES
X	1	0	1	0	HIRES mix with 40-col text
X	1	0	1	1	HIRES mix with 80-col text
0	0	0	1	1	140 x 192 16 color
1	0	0	1	1	560 x 192 black and white

In addition, the Mega II has an external controller mode that allows an external controller to use bus video time. This mode, along with the Change-Memory-Map switch, are in register C029. The following describes the external controller mode:

D7 DISABLE VIDEO. When this bit is 0, all existing video Apple II modes are supported. When it is a 1, the memory address bus is tri-stated during PH1, except for refresh. All other timing signals to memory are unaffected. Also, when this bit is 1, the memory map is changed to linear as described in D6.

D6 CHANGE MEMORY MAP. When this bit is 0, the memory map is the same as the normal Apple II. When this bit is 1, the memory map is linear and starts at $2000 and ends at $9FFF in Auxiliary memory.

D5 B&W/COLOR. When this bit is 0, then double HIRES is displayed in color (140 x 192 16 color). When this bit is 1, the double HIRES display (560 x 192) is black and white.

D4 to D1 Reserved. Must be programmed with a zero.

D0 ENABLE BANK LATCH. When this bit is 1, the data on alternate-data-bus bit 1 is latched as bank address bit 0. When this bit is 0, the data is not latched.

When the bank address is latched, the Main/Aux hierarchy is as follows: if the bank address is 1, Aux memory is always addressed; if the bank address is 0, the soft switches determine which bank is accessed.

Memory Translation

To accommodate the linear video map, the Mega II performs a permanent logical to physical translation of some of the memory. Figure 12 shows this translation.

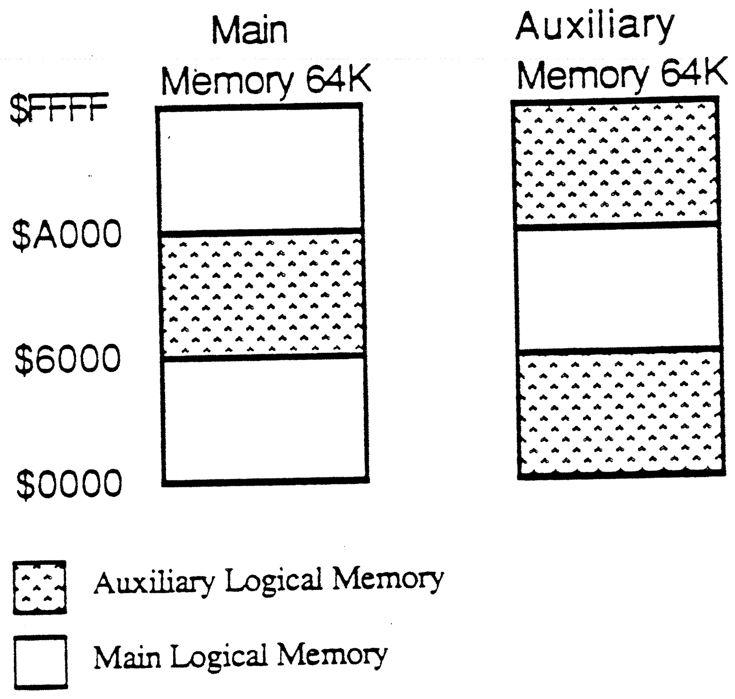


Figure 12. Mega II Physical Memory Map

State Register

See p13 ch 1 = C∅68

The State register contains soft switches that have the same addresses where they were accessed in the Apple II. The State register is a read/write register.

D7:	ALZP
D6:	PAGE2
D5:	RAMRD
D4:	RAMWRT
D3:	RDROM
D2:	BANK2
D1:	ROMBANK
D0:	SLOTCxROM

Select Signal

The select signal, input active low, is used to inhibit PH0 operations in the chip. The PH1 operations, refresh and video, are unaffected by this signal.

Video Registers

The horizontal and vertical video counters can be read through two video count registers. The horizontal count register at location $C02F reads H0 through HPE and VA. The vertical count register at location $C02E reads VB through V5.12

Chapter 4

Video Graphics Controller

Introduction

The Video Graphics Controller (VGC) is a custom integrated circuit that performs the following functions:

- Supports and enhances existing Apple II video
- Implements new video modes
- Interfaces to the Macintosh clock chip
- Provides interrupt handling for three interrupt sources
- Assists in the interface to new disk drives
- Provides test modes for chip and board-level testing

The VGC operates in a computer system with the Mega II. Figure 13 shows the VGC, Mega II, 128K RAM, and signals and buses connecting these elements.

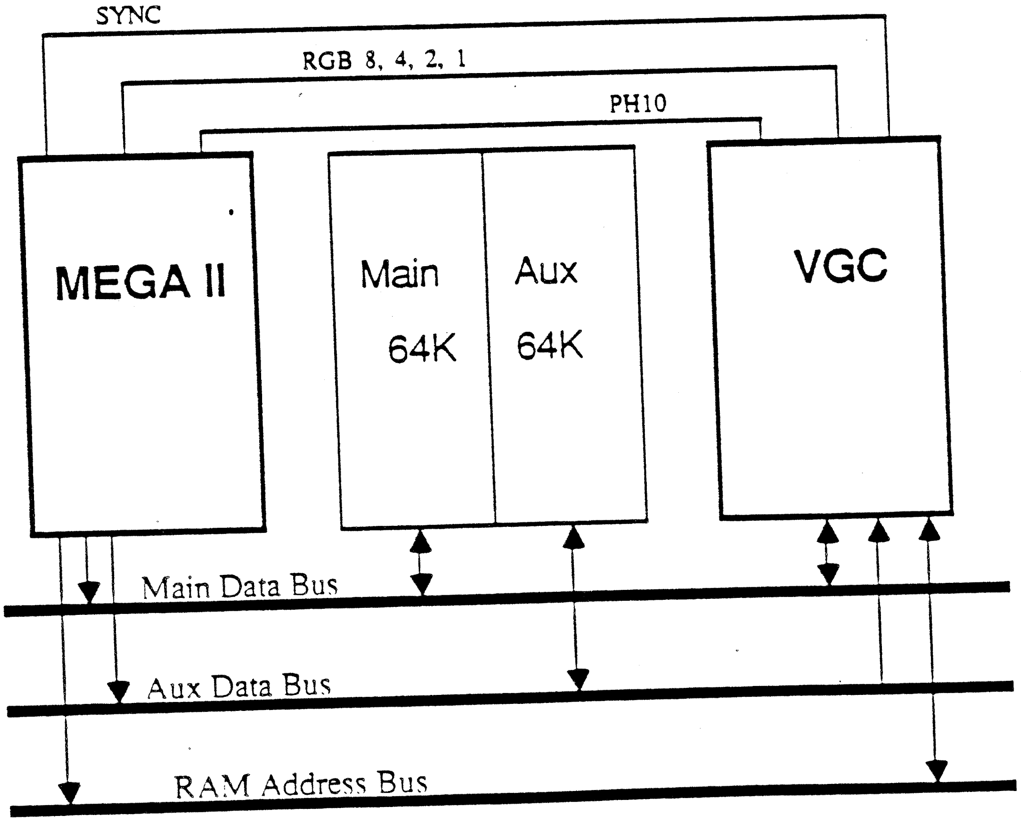


Figure 13. System Signals and Buses

The VGC Supports Apple II Video Modes

The VGC supports and enhances existing Apple II video modes, 40- and 80-column text modes, high resolution, low resolution, medium resolution, double high-resolution graphics, modes and, mixed text/graphics modes. The enhancements include:

- A text-background color register that permits the user to select any of the 16 Apple II colors for text and background.
- A border color register that permits the user to select any of the 16 Apple II colors for the perimeter of the video image.
- A video output that supports the generation of gray-scale video; color video output can be displayed on monochrome monitors in shades of gray.

The following lists the color names and their associated numbers:

$0	black	$8	brown
$1	magenta	$9	orange
$2	dark blue	$A	light gray
$3	purple	$B	pink
$4	dark green	$C	green
$5	dark gray	$D	yellow
$6	medium blue	$E	aqua marine
$7	light blue	$F	white

Text Background Register

The most significant four bits of the text-background register determine the text color; the least significant four bits determine the background color. On reset, the default is to white text on a black background. This is a read/write register.

Border Color Register $\longrightarrow$ $ Φ33 see p. 44

depending upon static CMOS RAM chip

This register serves a dual purpose. The least significant four bits determine the border color. On reset, the default color is black. The most significant three bits are the control bits for the clock chip interface logic. The remaining bit is reserved; it sets to zero for write commands and returns zero for read commands.

Mono/Color Register

The mono/color register uses only bit seven; the remaining bits are used for writes to this location. If the computer is used with a monochrome monitor, this switch should be set. It is a write-only soft switch.

If the Mono/color soft switch is set, then the color information is removed from the composite video signal. This cleans up higher frequency artifacts from monochrome composite monitors that results in a better looking display. This switch affects only the composite video outputs, the NTSC, and PAL outputs; it does not affect the analog RGB video output of the computer.

The VGC also removes the color information from the composite video signal when the computer is running in any of the existing Apple II text modes (except mixed text/graphics modes). This is done so that text displayed on a color composite monitor is not subject to color fringing and, therefore, is more readable. Composite color monitors cannot display colored text, colored background, or colored borders if the computer is running in text modes. Instead, the text, background, and border are displayed in shades of gray. Again, this applies to the NTSC or PAL composite video output of the computer and has no effect on the analog RGB video.

When the computer is running in mixed text/graphics modes, the four lines of text at the bottom of the display exhibits color fringing on composite color monitors. This is unavoidable because the color detection circuitry of color monitors cannot respond fast enough to the switching on and off of the chroma information.

New Video Modes

The VGC implements new video modes for the Apple II. The following lists the features of these new modes:

- 320 or 640 horizontal resolution
- 200 line vertical resolution
- 12-bit color resolution allows 4096 colors to be displayed
- 16 colors for each of the 200 lines--up to 256 colors per frame
- Polygon fill mode
- Scan line interrupts
- All new video mode features are programmable for each scan line
- Linear display buffer
- Pixels contained within byte boundaries

The video modes are turned on by writing to a soft switch/register. This is one of several Mega II soft switches that are *shadowed* in the VGC.

The new video modes are facilitated by the way the Mega II refreshes the dynamic RAMs. The Mega II refreshes the RAMs using five of 65 cycles available for each video scan line; this scheme frees the remaining video cycles to be used by the VGC when it is running in the new video modes. During the time the video display is in horizontal blanking, the VGC reads data from the pointer and color palette area of the display buffer that determines the options and colors for the next scan line. Then, during the time that the VGC is displaying the line, data is read from the pixel area of the display buffer; this information determines the color of the individual pixels from among the 216 colors loaded for that line.

The Display Buffer

The display buffer for the new video modes is shown in Figure 14.

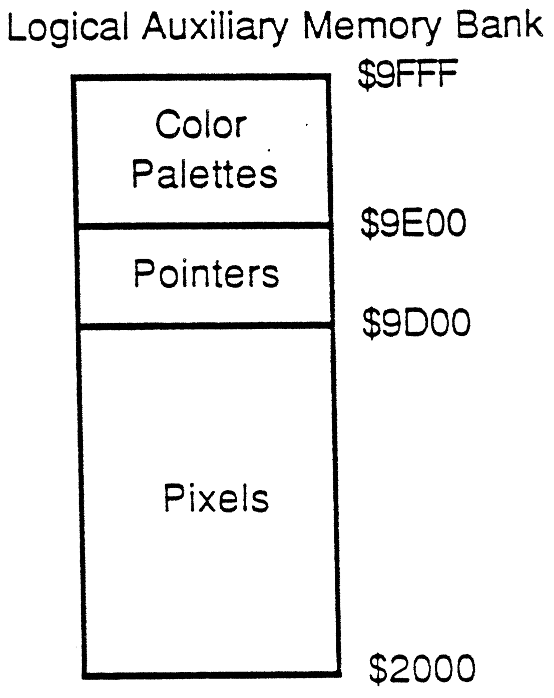
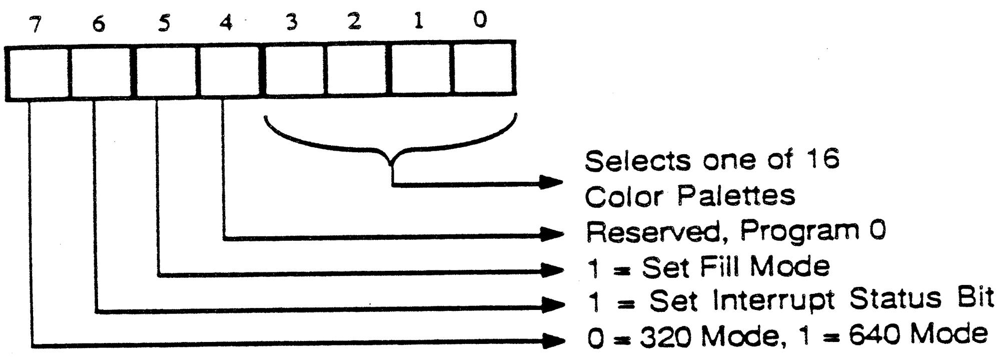


Figure 14. Display Buffer

The display buffer resides in the logical auxiliary 64K bank of the Apple II RAM from $2000 through $9FFF. It contains three types of data for the new video modes: pixel data, pointer bytes, and color palettes.

Pointer Bytes

The pointer bytes determine the features of each scan line; 320 or 640 horizontal resolution, scan line interrupt, polygon fill mode, and the colors for the line. There are 200 pointer bytes in the memory page--one pointer for each of the 200 scan lines. The location of the pointer byte for a scan line is $9DXX, where XX is the hexadecimal value of the line. For example, the pointer byte for the first scan line (line 0) is located in memory location $9D00; the pointer byte for the second scan line (line 1) is in location $9D01, and so forth. The first 200 of the 256 bytes in the memory page, beginning at $9D00, are used as pointers and the remaining 56 bytes are reserved and must be filled with zeros. The format for pointer bytes is shown is Figure 15.



7 6 5 4 3 2 1 0
Selects one of 16 Color Palettes
Reserved, Program 0
1 = Set Fill Mode
1 = Set Interrupt Status Bit
0 = 320 Mode, 1 = 640 Mode

Figure 15. Pointer Byte $9DXX

Color Palettes

The area of the video buffer starting at $9E00 up to $9FFF is used to store color palettes. Sixteen color palettes reside in this space. Each color palette contains 16 colors (totaling 256 colors). Each color within a palette requires two bytes of memory. The color format for these bytes is shown in Figure 16.

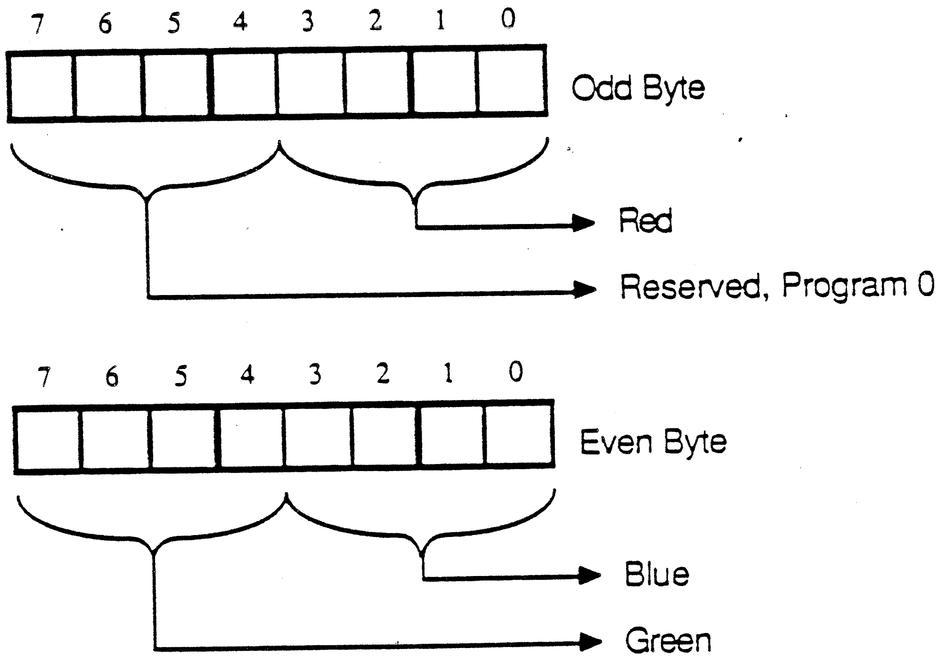


Figure 16. Color Format

The starting addresses of the color palettes are listed in Table 6.

Table 6. Palette Starting Addresses

Palette Number	Starting Address
$0	$9E00
$1	$9E20
$2	$9E40
$3	$9E60
$4	$9E80
$5	$9EA0
$6	$9EC0
$7	$9EE0
$8	$9F00
$9	$9F20
$A	$9F40
$B	$9F60
$C	$9F80
$D	$9FA0
$E	$9FC0
$F	$9FE0

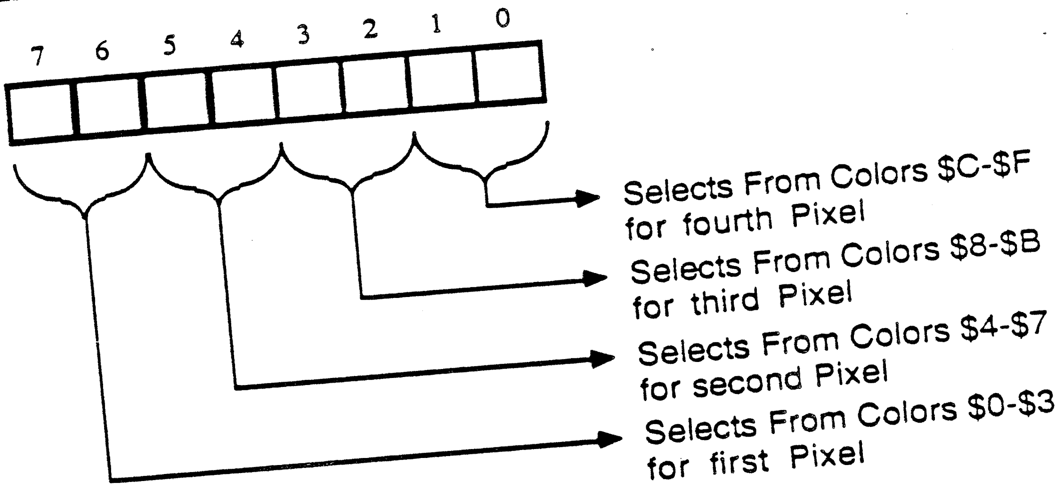
The 16 colors within a palette are assigned numbers $0 through $F. The addresses for palette $0 are listed in Table 7.

Table 7. Color Addresses

Color Number	Addresses
$0	$9E00 - $9E01
$1	$9E02 - $9E03
$2	$9E04 - $9E05
$3	$9E06 - $9E07
$4	$9E08 - $9E09
$5	$9E0A - $9E0B
$6	$9E0C - $9E0D
$7	$9E0E - $9E0F
$8	$9E10 - $9E11
$9	$9E12 - $9E13
$A	$9E14 - $9E15
$B	$9E16 - $9E17
$C	$9E18 - $9E19
$D	$9E1A - $9E1B
$E	$9E1C - $9E1D
$F	$9E1E - $9E1F

Pixels

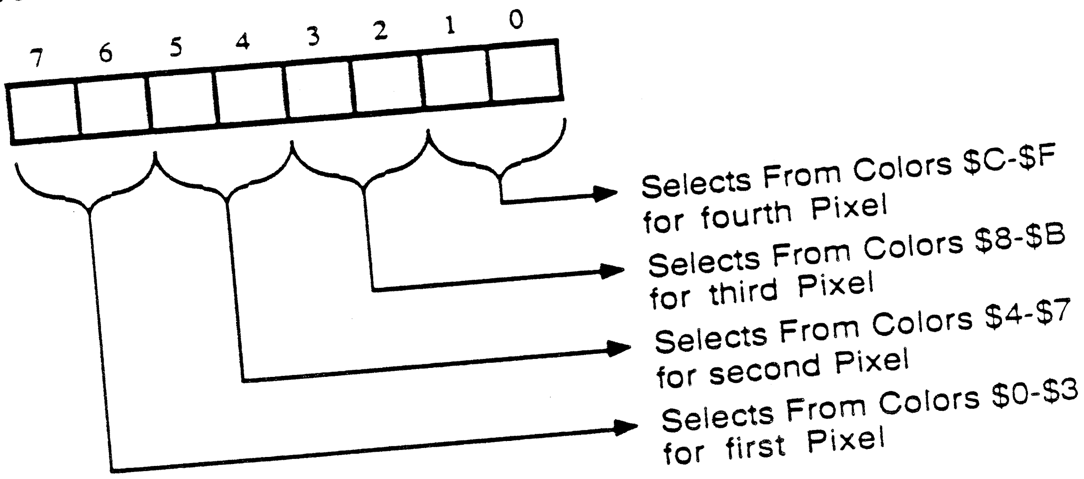
In the 320 mode, information for two pixels is contained in one byte of the pixel area of the display buffer. Each pixel uses 4 bits so that a pixel can be any of the 16 colors loaded for the scan line (see Figure 17.).



7 6 5 4 3 2 1 0
Selects From Colors $C-$F for fourth Pixel
Selects From Colors $8-$B for third Pixel
Selects From Colors $4-$7 for second Pixel
Selects From Colors $0-$3 for first Pixel

Figure 17. Pixel Data Byte Format, 320 Mode

In the 640 mode, information for four pixels is contained in one byte of the pixel area of the display buffer. Two bits can select from among four possible colors, however, all 16 colors loaded for the scan line can be displayed on the line. This is accomplished by having the first pair of bits in the byte (bits 6 and 7) select from among colors $0, $1, $2, and $3. The second pair of bits selects from the third set of four colors, and the fourth pair of bits selects from the fourth set of four colors (see Figure 18).



7 6 5 4 3 2 1 0
Selects From Colors $C-$F for fourth Pixel
Selects From Colors $8-$B for third Pixel
Selects From Colors $4-$7 for second Pixel
Selects From Colors $0-$3 for first Pixel

Figure 18. Pixel Data Byte Format, 640 Mode

The pixel data is mapped into the display buffer in a linear and contiguous manner from $2000 through $9CFF, with $2000 at the upper-left corner of the display, and $9CFF at the lower-right corner. Each scan line uses 160 ($A0) bytes.

Polygon Fill Mode

The Polygon Fill mode is used to rapidly fill a large area of the video display with a single color, and works only in the 320 mode. The fill mode redefines the pixel value of $0; it causes the color of the previous pixel to be displayed again instead of selecting color $0. This means that 15 colors are available for each scan line rather than 16.

For example, assume that A, B, and C represent three diffferent pixel values or colors (four bits per color), and that they are not $0. The desired colors for a series of pixels on a line is:

AAAAAAAAABBBBBBBBCCCCCCCC

The pixel values shown below would produce the color pattern in fill mode:

A00000000B0000000C0000000

The program needs only to fill the pixel area of the display with 0, then write a color value into those locations where a color should start and where it should change or stop.

The only provision is that the first pixel value on a scan line must not be 0; else an undetermined color results.

Clock Chip Interface

The VGC communicates with the clock chip through two read/write registers, control and data. The control register also serves as a border color register (see Figure 19).

see p.38

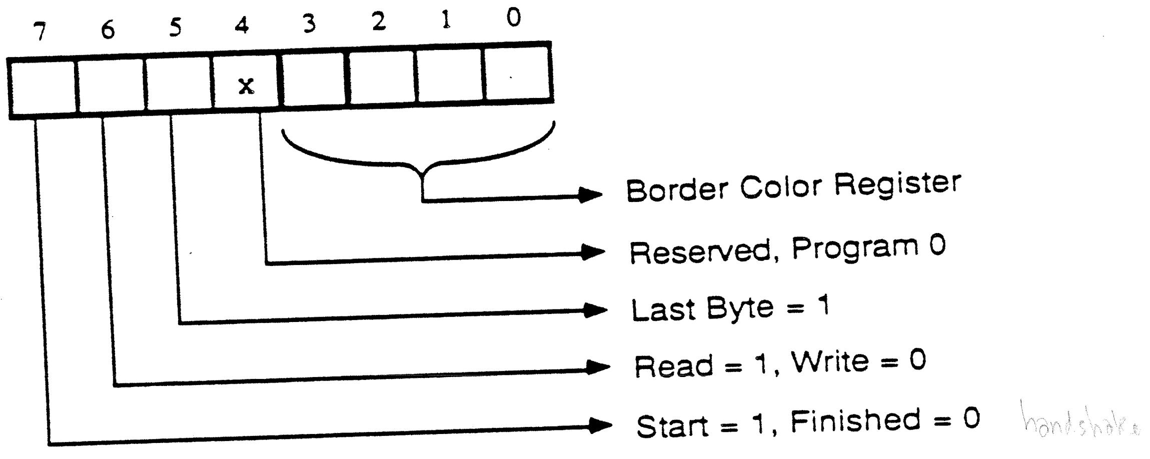


Figure 19. Clock Control and Border Color Register

Data transfers between the VGC and the clock chip are processed one byte at a time under software control. To write to the clock chip, the program must first write the data to be transferred into the data register ($033), then set the appropriate bits in the control register. To read from the clock chip, the program must set up the control register, then read the data register.

or a read or write, the transaction begins when bit 7 of the control byte is set to 1. The program can determine that the exchange is complete by polling bit 7 for a 0.

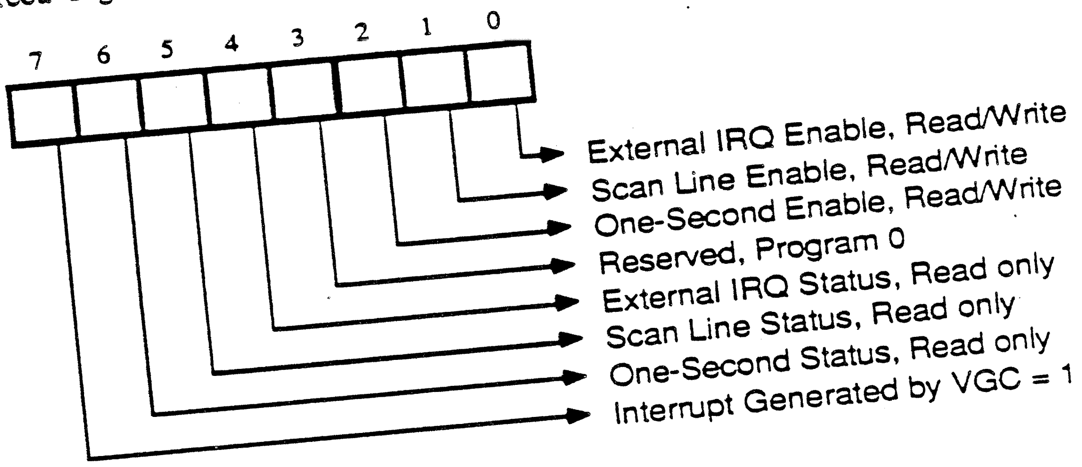
Bit 6 is the read/write bit, from the perspective of the VGC. A read is a data transfer from the clock chip to the VGC; a write transfers data from the VGC to the clock chip.

Bit 5 is the *last byte* control bit. A data transfer typically involves an exchange of two to three bytes. Bit 5 must be set to 1 for transferring the last byte and must be set to 0 for the other bytes.

VGC Interrupt Registers

There are three interrupt sources processed by the VGC: One Second, Scan Line, and External. The interrupt register contains a status bit and an enable bit for each of these three sources; another bit is set if any of the three VGC interrupt sources generates an interrupt.

An interrupt occurs if both the status and enable bits for any one of the interrupt sources are set. The enable bits are set or cleared under software control by writing into the appropriate positions in the Interrupt register; the status bits are set by the interrupt source itself. Only the enable bits in the Interrupt register can be written; writing into the other bit positions has no effect. Figure 20 shows the External Interrupt register.



7
6
5
4
3
2
1
0
External IRQ Enable, Read/Write
Scan Line Enable, Read/Write
One-Second Enable, Read/Write
Reserved, Program 0
External IRQ Status, Read only
Scan Line Status, Read only
One-Second Status, Read only
Interrupt Generated by VGC = 1

Figure 20. External Interrupt Register

The one-second status bit and the scan-line interrupt status bit is cleared by the software by writing zeros into the appropriate bit positions. Writing a one into these positions or writing into the other bit positions has no effect. The scan-line status bit is cleared by reading from the video counters in the Mega II (see Figure 21).

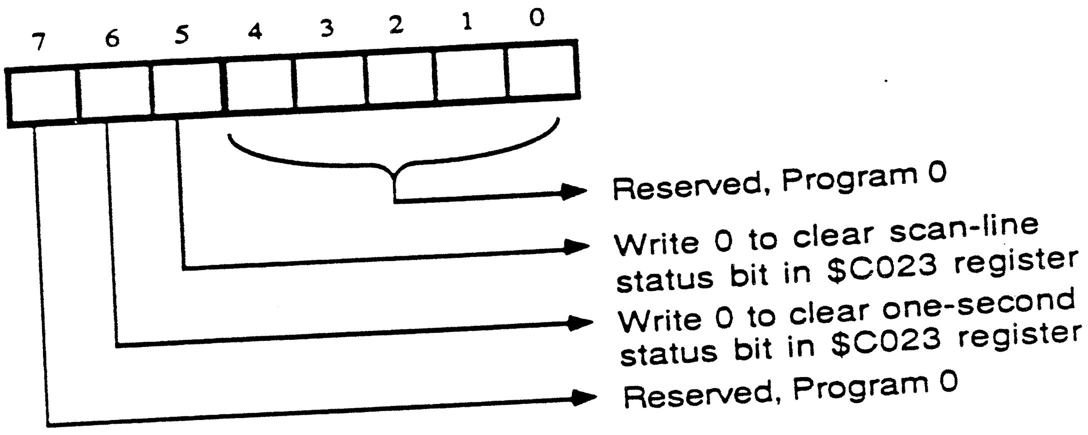

7
6
5
4
3
2
1
0
Reserved, Program 0
Write 0 to clear scan-line status bit in $C023 register
Write 0 to clear one-second status bit in $C023 register
Reserved, Program 0

Figure 21. One-Second and Scan-Line Interrupt Register

The one-second status bit is set by a 1 Hz input signal to the VGC from the clock chip.

The scan-line status bit is set by setting the interrupt bit in one of the pointer bytes. Scan-line interrupts can only be generated when the computer is operating in the new video modes.

The external status bit is set by driving one of the VGC input pins to a logic 0 level. This status bit is level sensitive; to clear it, software must control the device that is driving that input.

VGC Disk Register

The Disk register is the hardware interface to the disk drives. Two bits are used; bit 7 is the head select bit (HDSEL), and bit 6 is the select for the 3.5-inch disk drives (3.5DISK). Both of these bits are cleared on reset, and when the Disk register is read, zeros are returned in the unused bit positions.

VGC Test Modes

The VGC can be put into the test mode by pulling the TEXT input pin to a logic 0 level. When this pin is low, an internal dot-clock signal is multiplexed out onto the 3.5DISK output pin. This signal is used inside the chip to clock out video data; it can be used for signature analysis testing of the video outputs.

When the VGC is in the test mode, writing a 1 in bit 0 of the Disk register "breaks up" the internal video counters and address counters and causes them to count in a different sequence. This is done to speed chip testing. When the counters have been broken, only a power-on clear can restore them to their normal operating condition.

Chapter 5

Slot Maker

General Description

The Slot Maker is an integrated circuit that generates control signals for the Apple II computer I/O bus. A six-bit encoded value is used to determine the output to be asserted. Additionally, the Slot Maker buffers clock signals that are distributed to the slots. The annunciator outputs that are available on the Apple II's game connector are also generated by the Slot Maker.

Chapter 6

Key Glu

Introduction

The Key Glu works with a microprocessor to form an intelligent keyboard and Input Device Bus interface. The Key Glu, using two independent data buses, serves as a communications interface with multiple internal read/write registers to store keyboard data, key modifiers, mouse X/Y-coordinates, command data and status information. All registers, except the status registers, have a status flag which is set when the register is written to and cleared when the register is read. The keyboard data, mouse, and data registers also have interrupt bits that generate system interrupts when enabled. The Key Glu includes a one-of-ten decoder for driving the keyboard matrix scanner Y-axis.

Key Glu Read-Only Status Register

0	1 = any key down
1	Always 0
2	Always 0
3	Always 0
4	1 = keyboard strobe set
5	1 = data register full
6	1 = command register full
7	1 = mouse X-axis register full

System Status Register

0	1 = command register full, read only
1	0 = mouse X-axis register available, 1 = mouse Y-axis register available, read only
2	1 = key data interrupt enable, read/write
3	1 = key data full, read only
4	1 = data interrupt enable, read/write
5	1 = data register full, read only
6	1 = mouse interrupt enable, read/write
7	1 = mouse register full, read only

Status Flags

The Keyboard Strobe flag is set by writing to the keyboard data register; cleared when system asserts the following combination: KS2=0, KS=0, KS0=1, and Phase 0=1.

The Command Full flag is set when the system writes to the command register; cleared when the key micro reads the command register.

The Mouse Register Full flag is set when the key micro writes to the mouse X-axis register; it is cleared when the system reads the system status register followed by a read from the mouse register twice. The first mouse-register read returns the X-coordinate; the second read returns the Y-coordinate.

The Data Register Full flag is set when the key micro writes to the data register; it is cleared when the system reads the system status register followed by a read from the data register.

The Keyboard Data Full flag is set when the keyboard data register is written into; it is cleared when the system reads the system status register followed by a read from the keyboard data register.

Chapter 7

Sound Glu

Introduction

The Sound Glu interfaces the Ensoniq sound chip and the Mega II's 64K by 8K dynamic RAM; this allows the Ensoniq chip to run asynchronous of the Mega II. The Sound Glu contains a 16-bit auto-incrementing address pointer and a bidirectional data holding latch.

Write Operation

To write to the Ensoniq chip or RAM, set the control register to point to either the RAM or the Ensoniq chip and enable or disable auto-address incrementing. Then, load the address pointer to the beginning location in which data is to be written. Data can then be written into the data register. If the auto increment feature is used, the address register increments the pointer to the next location after each write into the data register. A minimum of 1.12 microseconds delay is experienced when accessing the Sound Glu chip.

Read Operation

The read operation is the same as the write operation with one exception--the data read back lags by one read cycle. For example, if you want to read 10 bytes from the RAM, set the control register to point to the RAM and enable auto incrementing. Then set the address pointer to the starting address, and read the data register eleven times, discarding the first byte read. Table 8 lists the registers and their types; Table 9 lists the control register bits and their functions.

Table 8. Operation Registers

Register	Type
Control register	R/W
Data register	R/W
Address pointer, low byte	R/W
Address pointer, high byte	R/W

Table 9. Control Register

Bit	Function
Bit 0	0 = Access DOC, 1 = Access RAM
Bit 1	0 = Disable address pointer auto incrementing 1 = Enable address pointer auto incrementing
Bit 2-7	Ignored, read back as zeroes

Chapter 8

Front Desk Bus

Introduction

The Front Desk Bus is a method and protocol for interconnecting computers with input devices. This document describes the physical layer, datalink, and the network layers of the Front Desk Bus. In this discussion, the computer is referred to as the host, the peripherals connected to the bus are referred to as devices.

The host controls the flow of data by issuing commands (only the host can issue commands). The Talk command is used for data transaction from a device to the host; the Listen command is used for a data transaction from the host to a device.

Physical Layer

All devices communicate with the host by way of a 3.5 mm mini-phono jack.

Modulation

There are three forms of modulation on the bus: Normal modulation that transmits commands and data, High Speed modulation that transmits data, and Signals that broadcast global messages, such as Service Request and Reset.

Normal Modulation

An RZ code for modulation has been adopted for the Front Desk Bus. Each bit-cell boundary is signified by a falling edge on the bus.

High Speed Modulation

High speed modulation is used for data and not for commands.

Signals

Certain transactions are neither commands nor data transactions. These are special transactions that globally broadcast status to devices on the bus. There are four special transactions in this group: Attention, Sync, Reset, and Service Request. The following paragraphs describe these transactions.

Attention and Sync

The start of a command is signaled by a long attention pulse. This is followed by a sync pulse to begin the initial bus timing. The falling edge of the sync pulse is used as a timing reference for the first bit of the command.

Reset

Reset issues a break on the bus.

Service Request

Service request is a transaction that devices can use to signal the host that they require service, such as have data to send. Following any command transaction, a requesting device can signal by holding the bus low during the low portion of the stop bit of the command transaction. The requesting device holds the bus low beyond the bit-cell boundary to signal.

When a device has requested service, it signals Request Service repeatedly until served. When the requesting device is addressed to Talk, it is considered served and it does not send a Request Service signal again until it needs to be served. The ability for a device to send Request Service can be enabled and disabled by the host.

Transactions

Transactions are initiated by commands. The format of a command is: an attention signal, followed by a sync signal, followed by eight data bits. To synchronize the stopping of the transaction, a stop bit is transmitted following the imaginary bit-cell boundary. After the stop bit, the transaction is complete and the host releases its active drive of the bus.

Data Link

Each device on the bus has an address. There is only one active talker on the bus at a time; this may be the host or an addressed device. A device addressed to Talk, with data to send, *untalks* itself after it sends its data. If a device has no data to send, it *untalks* itself immediately and allows the bus to time out. The host may also send data after a command.

Front Desk Bus Peripherals

Each peripheral device has a four-bit command address that identifies its device-type. A device always responds to its address in case of a power-on or a reset signal.

Registers

All devices have, at most, four locations to receive data. The following describe these locations:

Resister 0, Talk:	Data register, device specific.
Register 0, Listen:	Data register, device specific.
Register 1, Talk:	Data register, device specific.
Register 1, Listen:	Data register, device specific.
Register 2, Talk:	Data register, device specific.
Register 2, Listen:	Soft addressed devices: Device specific.
Register 3, Talk:	Status information, that is, device address handler.
Register 3, Listen:	Status information, that is, device address handler.

Commands

Commands can be sent only by the host. There are four commands: Talk, Listen, SendReset, and Flush. A command is an eight-bit word with the following syntax(see Table 11):

- A four-bit device address field that specifies the address of the desired device. The most significant nibble is the address that ranges from 0-15 (A3-A0).
- A two-bit command
- A two-bit register address field (RB, RA). This field, which is optional, allows a specific register, R0 to R3, within an addressed device to be specified. For example, to differentiate a data register (in a keyboard, keystroke) from a status or configuration register (a response that signifies the model of the keyboard).

Table 11. Command Syntax

7654	32	10	Command
XXXX	00	00	Send Reset*
A3-A0	00	01	Flush
XXXX	00	10	Reserved
XXXX	00	11	Reserved
XXXX	01	XX	Reserved
A3-A0	10	RB RA	Listen
A3-A0	11	RB RA	Talk

*Forces RESET signal on FDB

To allow for future expansion of the command structure, a group of *place holder* instructions has been defined. These instructions are treated as no-ops.

Talk

All devices on the bus must support Talk and Listen commands. When a device is addressed to Talk, it must respond before being timed out by the host.

The selected device, if it does not time out, becomes active on the bus. It performs its data transaction, *untalks* itself, and is inactive on the bus.

Listen

When a device is addressed to Listen, it is enabled to receive the data bits that are placed on the bus by the host. The host performs its data transaction. After the data bits are received, the transaction is complete and the device *unlistens* itself. If a device is addressed to Listen and it receives another command on the bus before it receives any data, the transaction is immediately considered complete and the device *unlistens* itself.

SendReset

The SendReset command sends a Reset signal to the bus. The Reset signal resets all pending Service Requests; turns the service request mode of all devices to enable; and puts the devices in a mode in which they will accept commands.

Flush

The Flush command is defined by the device. It is used to clear a fifo and resets all keys on a keyboard so that they can be sent again.

Collision Detection

All devices detect collisions of data. If a device is trying to output a 1 and the data line is or goes to zero, it has lost a collision to another device. If another device sends data before the device is able to assert its start bit, it has lost a collision. The losing device should immediatedly *untalk* itself and preserve the data that was being sent for retransmission. The device sets an internal flag if it looses a collision.

Note: Devices using internal clocks that operate within +/- 1% should attempt to assert their start bit at a random time within the limits of the line turn-around time.

Error Conditions

If the data line hangs low, all devices reset themselves and output a one. If a command transaction is incomplete by staying high beyond the maximum bit-cell time, all devices ignore the command and seek another attention signal.

Network Layer (FDB Types)

The Network Layer accommodates Normal Devices and Extended Address Devices.

Normal Devices

A normal device optionally has a device, called the *activator*, on it to indicate activity. The activator can be a special key on a keyboard or a mouse button. To aid in collision detection,.the address portion of the address field of Register 3 is replaced with a random number in response to a Talk R3 command. Normal devices will change their Register 3 to the data received when they receive a Listen R3 command, no collision detected, and activator inactive is true.

At the systems level, a host can change the address of normal devices by forcing the collision of devices sharing the same address. By issuing a Talk R3 command and following it with a Listen R3 command, with a new address in bits 8 to 11 of the data, all devices that detect collisions are moved to the new address. This process can be repeated at new addresses until the response to the Talk R3 command is a time-out. This can be used to identify and relocate multiple devices of the same type after initialization of the system.

At the applications level, addresses can be changed by displaying a message requesting a user to use the activator. The host then issues a Listen R3 command to a new address and all devices, except the one with the activator being used, are moved. This method can be used to identify and locate individual devices in multi-user applications (see Figure 22).

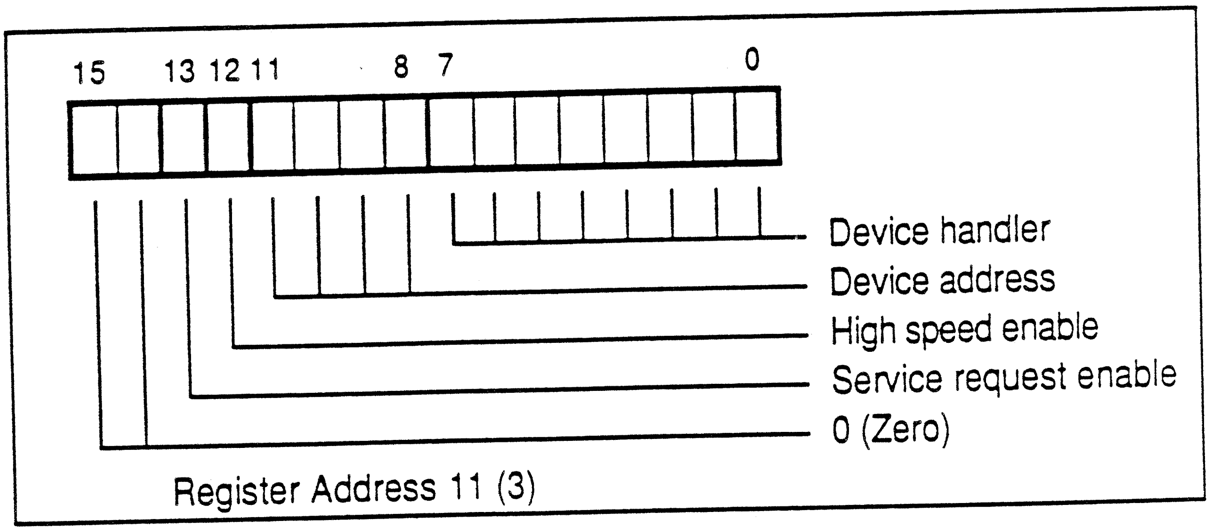


Figure 22. Register Address 11

Extended Address Devices

Extended-address devices have the same command address and a unique 16-bit extended address that is stored in the device. Their command address cannot be changed. On powerup or after RESET, they only accept the Listen R2 command in which the data match their stored address. When enabled, they respond to all commands addressed to them. These devices become disabled after receiving a Listen R2 command in which the data do not match their stored address.

Register 3

The function of a device and the use of its data by the host are controlled by a handler that is stored by the device in Register 3. The host changes the handler with a Listen R3 command. If the receiving device is able to function with the new handler, it is stored and sent in response to a Talk R3 command.

The handler "FF" hex is reserved for the self test mode for all devices. The handler "00" hex in response to a Talk is reserved to indicate a failed self test. The handler "00" hex sent with a Listen is reserved to indicate that the device is only to change the address portion of R3. The following are examples (see Tables 23 and 24) of how the handlers for keyboards (coded device) and mouse devices (relative device) could interpret data received from a Talk R0 command.

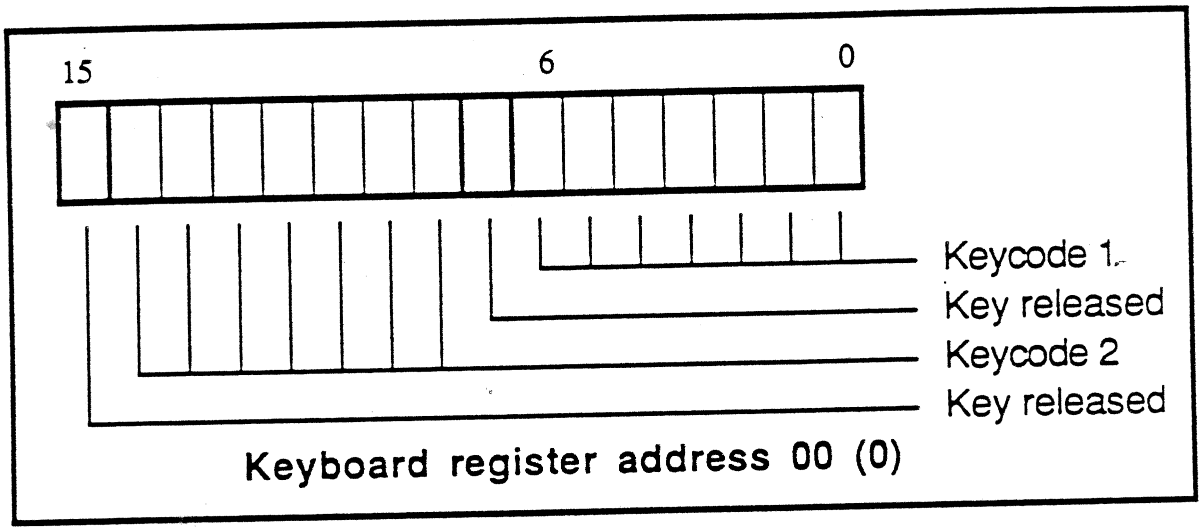


Figure 23. Keyboard Register

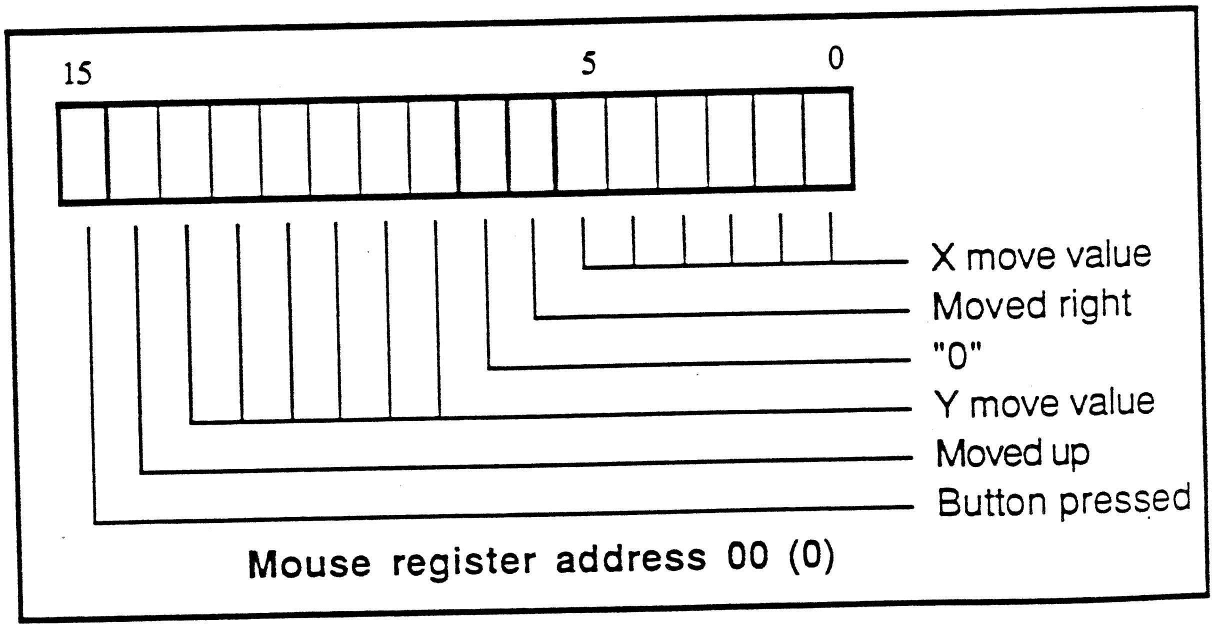


Figure 24. Mouse Register

Service Request and High Speed Enabling

The Listen R3 command is also used to enable and disable Service Request and High Speed modulation. They are enabled by setting the appropriate bit in R3 to a one and disabled by setting the appropriate bit to a zero.

Service Request is enabled on the bus by setting the R3 enable bit to one; it is disabled by setting the bit to zero. This is useful in systems where the Service Request response time in a polled system is longer than desired. When only specific devices are required for an application, the others can be disabled.

Chapter 9

Integrated WOZ Machine

Introduction

The Integrated WOZ Machine (IWM) allows a microprocessor to read and write serial GCR (group code) encoded data. The IWM has a status register, mode register, and multiple modes of operation. Also, the IWM provides an asynchronous mode which relaxes the precise software/hardware timing required in synchronous mode, a fast mode, and an optional 1-second one-shot timer to hold the enable outputs low.

The IWM is controlled by the following registers:

- Mode register
- Status register
- Write-handshake register
- Read data register

The IWM is controlled by setting state bits and reading or writing registers. The modes selected by the mode register include synchronous, asynchronous, slow, or fast. The format is an 8-bit nibble with the msb set. The msb of the data nibble is shifted in or written out first. A bit is transferred every bit-cell time.

Control Registers

The IWM is controlled by five internal registers. The following paragraphs describe these registers and their functions.

State Register

The state register is an 8-bit write-only pseudoregister. The bits in this register are individually addressed by A3, A2, and A1. The data on A0 is latched into the addressed state bit by /DEV being low. All eight state bits are reset to 0 by /RESET being low.

State bits L6, L7, and motor-on select the register and determine if the operation is to be a read or write. If the state of one of these bits changes, the new state selects the register to be accessed during the operation, and determines if the operation is to be a read or write.

The following is a list of the bits and their functions:

Address	Name	Function
0		If this bit =1, PHASE0 is set to a high state.
1		If this bit = 1, PHASE1 is set to a high state.
2		If this bit = 1, PHASE2 is set to a high state.
3		If this bit = 1, PHASE3 is set to a high state.
4	LMotor-on	If this bit = 1, the selected enable is set to a low state.
5	Drive-Sel	If this bit = 1, /ENABL2 is selected. If this bit = 0, /ENABL1 is selected.
6	L6	See Table 10.
7	L7	See Table 10.

The state bits, L6 , L7, and Motor-on select the register to be accessed for a read or write operation. Registers not selected by L6, L7, and Motor-on, are read during any operation in which A0 is a zero. A write operation is performed on a register when L6 and L7 are set to 1, and A0 is set to 1. A combination of L7, Motor-on, and /underrun enables /WRREQ (low). The following is a list of the register control bits, their selected functions, and their state:

L7	L6	Motor-On	Register Operation Selected	State
0	0	0	Read all ones	
0	0	1	Read data register	Read
0	1	x	Read status register	Write-protect sense
1	0	x	Read write-handshake register	Write
1	1	0	Write mode register	Mode set
1	1	1	Write data register	Write load

Mode Register (Write-Only)

All eight mode bits are reset to 0 by /RESET. The following is a list of the register bits and their functions:

	Bit	Function
LSB	0	1 = Latch mode. Must be set in asynchronous mode
	1	0 = Synchronous handshake protocol: 1 = asynchronous
	2	0 = 1-second on-board timer enable: 1 = timer disable
	3	0 = Slow mode: 1 = fast mode (2 us bit-cell timing)
	4	0 = 7 MHz: 1 = 8 MHz (7 or 8 MHz clock descriptor)
	5	1 = Test mode: 0 = Normal operation
	6	1 = MZ reset
MSB	7	Reserved for future expansion

Status Register (Read-Only)

The status register is read-only. The following is a list of the bits and functions of the status register:

Bit	Function
0-4	Same as mode register
5	1 = Either /ENBL1 or /ENBL2 is currently active (low)
6	1 = MZ. Reset to 0 by /RESET and MZ-reset
7	1 = SENSE input high: 0 = SENSE input low

Handshake Register (Read-Only)

The following is a list of the Handshake register bits and functions:

Bit	Function
0-5	Reserved for future use (read as ones)
6	1 = Write state (cleared to 0 if a write underrun occurs)
7	1 = Write data buffer register ready for data

Data Register

The operation of the data register depends on the setting of state bits L6 and L7 and on the synchronous mode bit. With L6 and L7 clear, the data register operates as a read-data register. With L7 set, the data register operates in the write state as a write-data buffer.

Appendix A

Mega II Character Generator ROM

Uppercase Characters

@ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [\ ] ' _

Lowercase Characters

' a b c d e f g h i j k l m n o p q r s t u v w x y z { | } ~

Special Characters

To be supplied

International Characters

English (USA)
English (UK)
French
Danish — To be supplied
Spanish
Italian
German
Swedish

Appendix B

Cortland Input/Output Timing

The Cortland computer has seven I/O slots that are almost identical to the slots in the Apple II, the only exception being /M2SEL which replaces µPSYNC on pin 39. The slots behave like their counterparts in the Apple II with only a few differences, the most important one being the behavior of the address bus. Since the Cortland computer can operate at 2.8 MHz and has a 24-bit address, the address bus to the slots is not always valid as it was in the Apple II. The signal /M2SEL indicates when a valid address for banks 224/225 (hex $E0/$E1) is present on the address bus and so should be used to qualify any address decoding that does not use one of the I/O enable lines. Figure 25 shows a pinout diagram of the I/O.

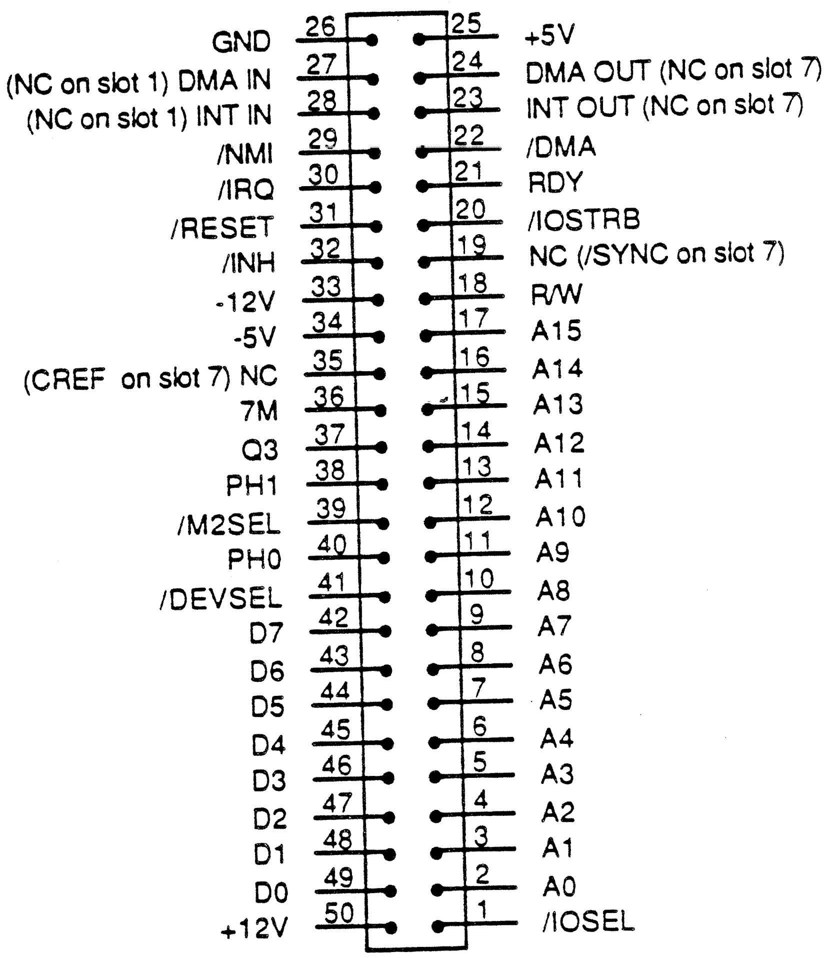


Figure 25. Input/Output Pinout Diagram

The total current available for the 7 slots is 500 mA at +5V, 250 mA at +12V, 200 mA at -5V, and 200 mA at -12V.

The support circuitry for the slots is designed to handle a DC load of 2 LS TTL loads and an AC load of no more than 15 pF per pin per slot.

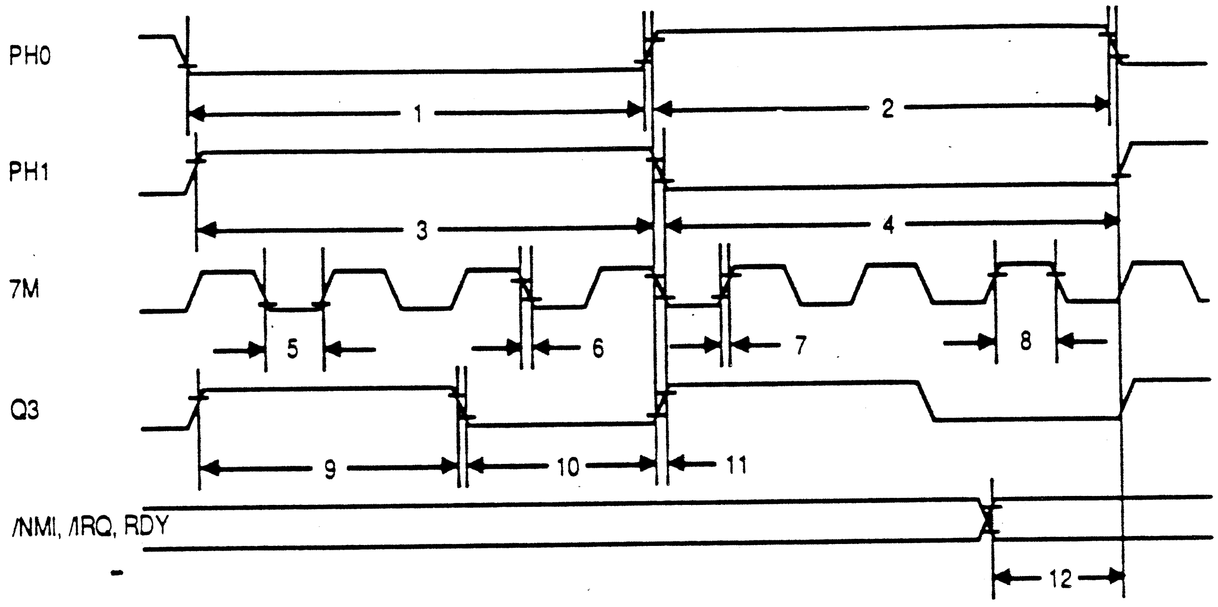


Figure 26. Input/Output Clock and Control Signal Timing

Number	Description	Min(ns)	Max(ns)
1	PH0 low time	480	
2	PH0 high time	480	
3	PH1 high time	480	
4	PH1 low time	480	
5	7M low time	60	
6	Fall time, all clocks		10
7	Rise time, all clocks		10
8	7M high time	60	
9	Q3 high time	270	
10	Q3 low time	200	
11	Skew, PH0 to other clock signals	-10	10
12	Control signal setup time	140	

All clock signals present on the I/O slots are buffered by a custom IC called the *Slot Maker*. These clock signals are delayed somewhat from the corresponding signals on the main board because they are buffered through the Slot Maker. All timing parameters given here (Figure 26) and with the other diagrams (Figures 27-29) have been adjusted to account for this delay.

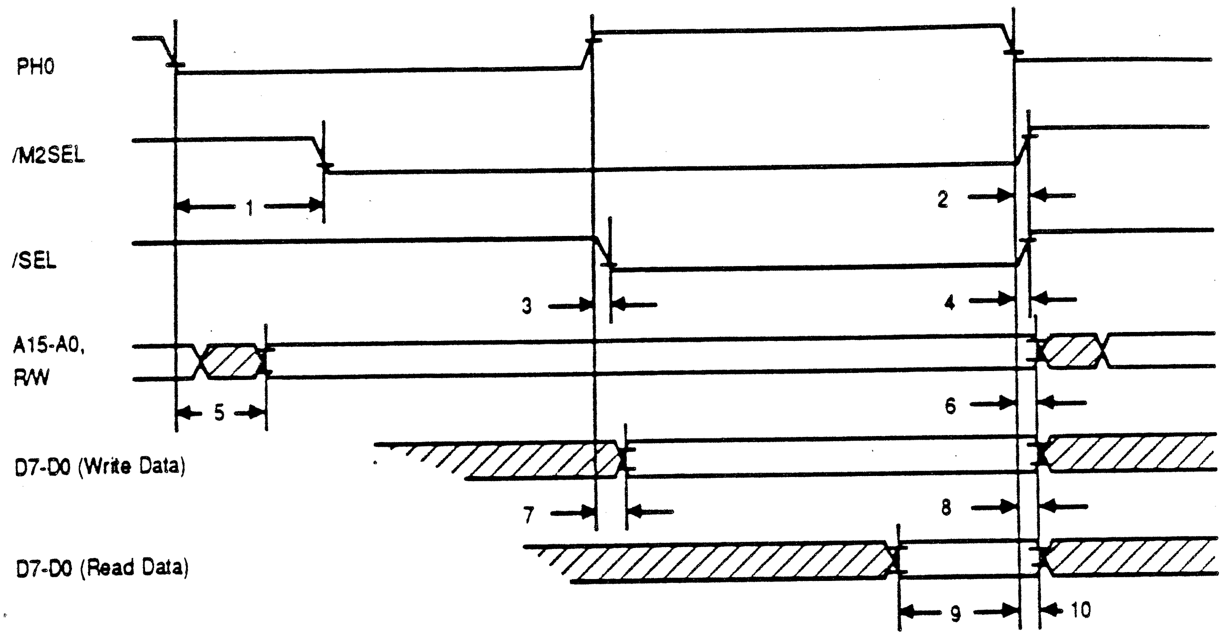


Figure 27. Slot I/O Read and Write Timing

Number	Description	Min(ns)	Max(ns)
1	/M2SEL low from PH0 low		160
2	/M2SEL hold time	-10	
3	I/O enable low from PH0 high (DEVn,/IOSELn,/IOSTRB)		15
4	I/O enable high from PH0 low (DEVn,/IOSELn,/IOSTRB)	10	
5	Address and R/W valid from PH0 low		100
6	Address and R/W hold time	15	
7	Write data valid delay		30
8	Write data hold time	30	
9	Read data setup time to PH0	140	
10	Read data hold time	10	

The standard Cortland slot I/O timing is shown in Figure 27. When the Cortland computer is running in its high-speed mode, the address bus to the I/O slots is not valid during the entire PH0 cycle, and therefore, cannot be used to perform unqualified address decoding. Signal /M2SEL, which replaces signal µSYNC, indicates when a slow, synchronized memory cycle is taking place and when the value on the address bus will remain valid during the current PH0 cycle. This means that cards that use the Apple II technique of *phantom slotting* to put multiple I/O devices on one card must use /M2SEL to qualify their address decoding circuitry.

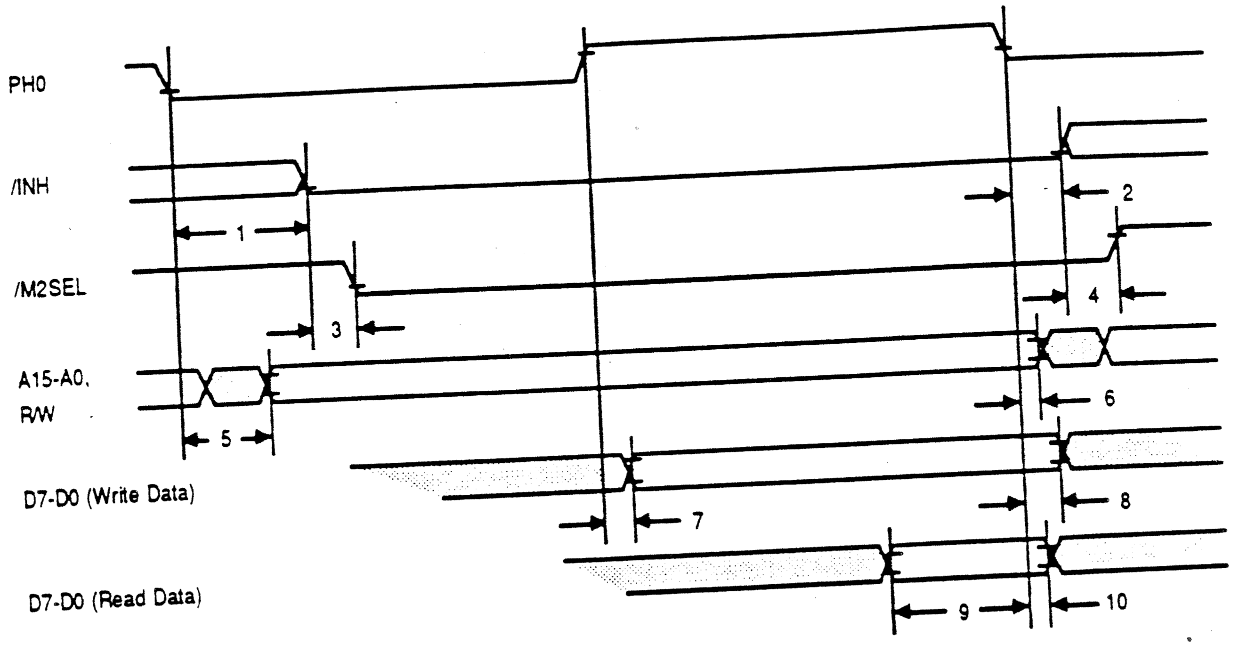


Figure 28. I/O Read and Write Timing With /INH Active

Number	Description	Min(ns)	Max(ns)
1	/INH valid after PH0 low		175
2	/INH hold time	15	
3	/INH low to /M2SEL low delay		30
4	/INH high to /M2SEL high delay		30
5	Address and R/W valid from PH0 low		100
6	Address and R/W hold time	15	
7	Write data valid delay		30
8	Write data hold time	30	
9	Read data setup time to PH0	140	
10	Read data hold time	10	

Read and write cycles that are directed to the I/O slots by /INH have the same timing parameters as normal I/O read and write cycles. When /INH is asserted, the computer responds as if a Mega II memory cycle were being performed.

Cards that use the /INH signal will function properly only if the computer is running in its slow mode (1 MHz). If the computer is running in its high-speed mode, the addresses that are seen by cards in the I/O slots are not guaranteed to be valid during an entire PH0 cycle. Also, because the upper eight bits of the memory address are not available to cards, the usefulness of /INH is greatly reduced in this machine.

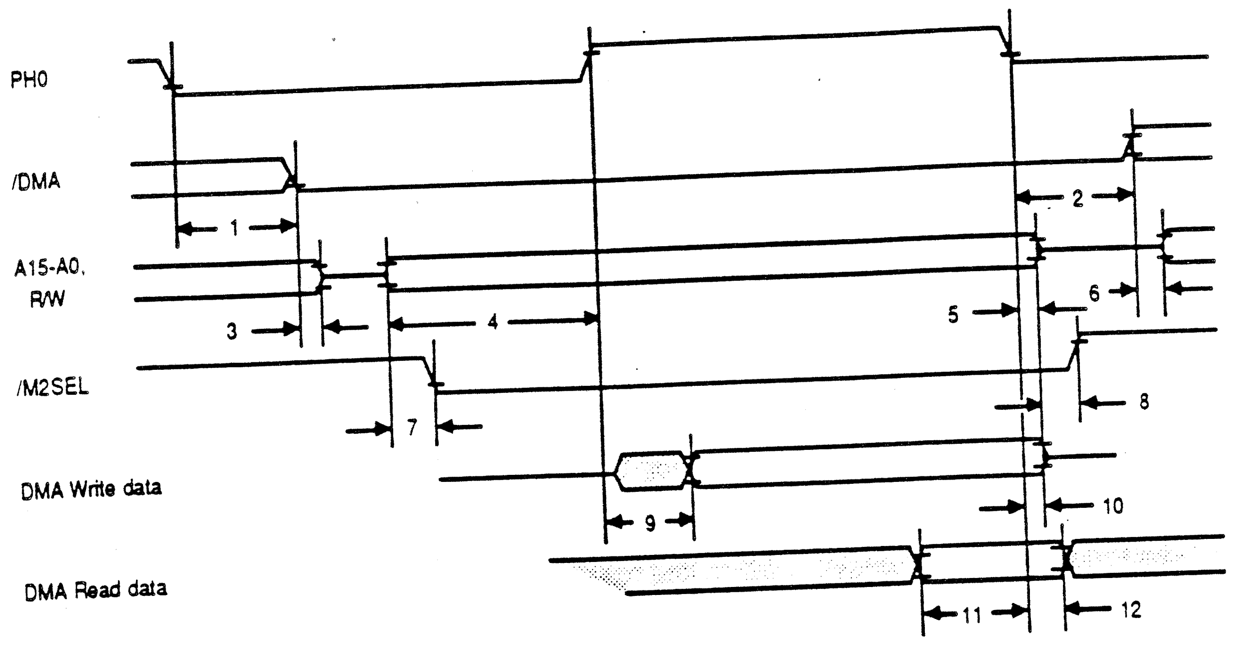


Figure 29. /DMA Read and Write Timing

Number	Description	Min(ns)	Max(ns)
1	/DMA low from PH0 low		120
2	/DMA high from PH0 low		120
3	A15-A0 and R/W float from /DMA		30
4	DMA address and R/W valid before PH0 goes high	300	
5	DMA address and R/W hold time	10	
6	/DMA high to A15-A0 and R/W active		30
7	DMA address valid to /M2SEL low		30
8	DMA address float to /M2SEL high		30
9	PH0 high to write data valid		100
10	DMA write data hold time	10	
11	DMA read data setup time	125	
12	DMA read data hold time	30	

DMA devices will work if the Cortland computer is running slow, at 1 MHz. If the computer is running in its high-speed mode, at 2.8 MHz, DMA will work to the high-speed memory (banks 0 - 127) and will not work with the slow part of the system (all I/O and video memory). DMA can be performed to or from any part of the Cortland memory map, provided that the DMA bank register in the FPI is first set to the correct bank.

Appendix C

Signal Descriptions

A15 - A0	Address input from CPU
MD7 = MD0	Bidirectional Main Data Bus
AD7 - AD0	Bidirectional Alternate Data Bus
RA7 - RA0	Multiplexed RAM address bus
14M	14.31818 MHz master clock input
7M	7 MHz output
PH0	65C02 clock phase zero
Q3	2 MHz asymetrical cloc.
CREF	3.58 MHz video-color reference signal
/SYNC	Video horizontal and vertical sync signal
SERVIDEO	Serial video multiplexed with color-burst gating signal
/WINDOW	Video nonblank window
/RAS	Multiplexed RAM row-address strobe
/RAMWE	Read/write signal to RAM
/ROMEN	Enable signal for the monitor ROM
/ROMBNK	Select for the upper 16K bytes of the monitor ROM
/CASM	Multiplexed RAM column address strobe for main memory
/CASA	Multiplexed RAM column address strobe for alt. memory
RGB1	Least significant bit of the 4-bit RGB output
RGB2	One of the four RGB outputs
RGB4	One of the four RGB outputs
RGB8	Most significant bit of the 4-bit RGB output
S4	One of 6 encoded signals used to generate the slot signals

/RESET	Reset input
/IRQ	Interrupt request output to CPU (open drain)
R/W	Read/write signal from CPU
/INH	Inhibit input from slots
/DMA	DMA input from slots
/KSEL1	Encoded select signal used by the keyboard interface
/KSEL2	Encoded select signal used by the keyboard interface
SPKR	Speaker output signal
/PDLTRIG	Output to clear paddle timer
/C060-67	Address decode for $C060 to $C067
/MSW	Mouse switch input
X1 - X0	Mouse X-direction quadrature signals
Y1 - Y0	Mouse Y-direction quadrature signals
/SEL	Signal that enables the Mega II for PH0 activity