AMIGA TRAINING COURSE

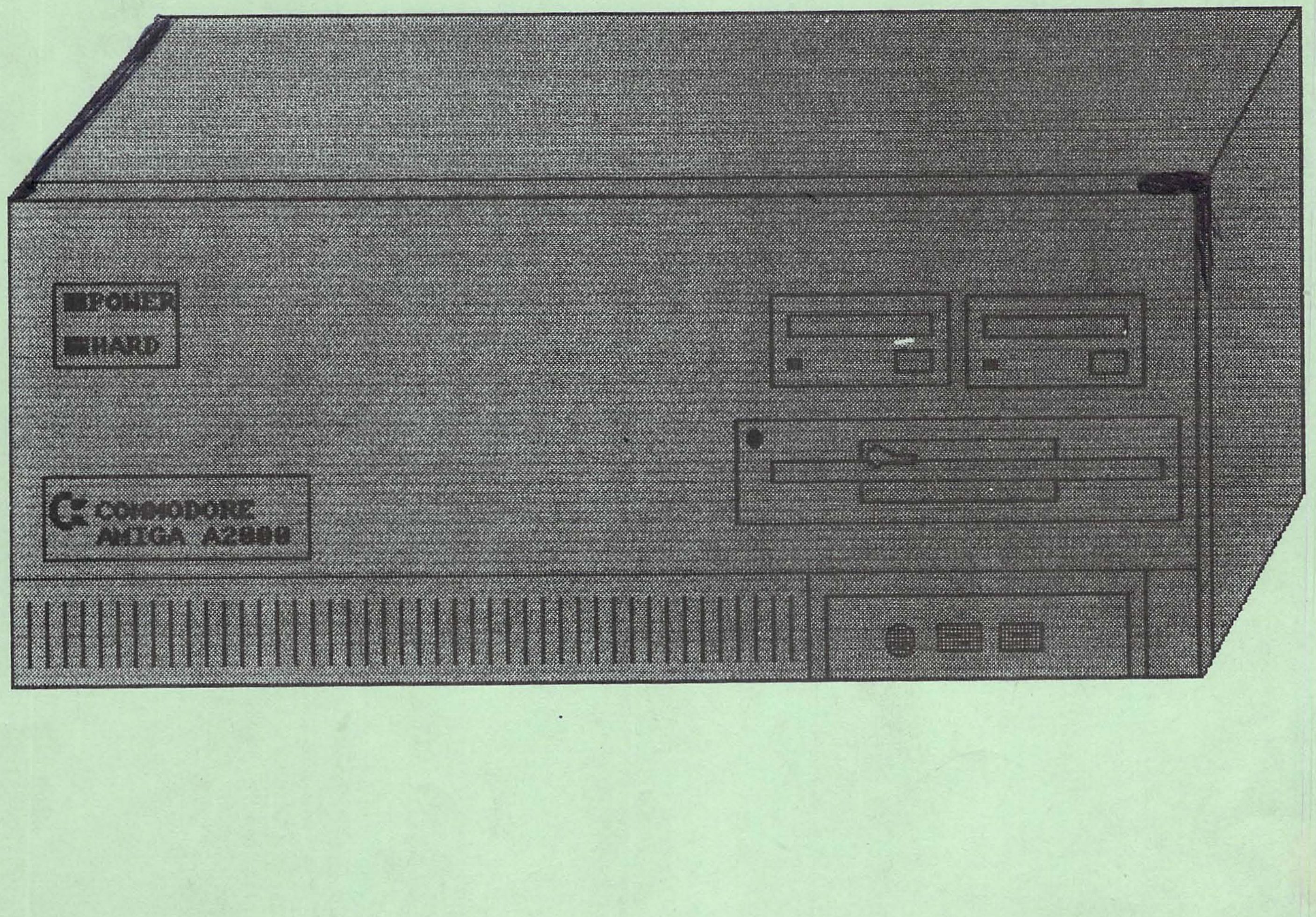


TABLE CF CONTENTS

TABLE OF CONTENTS

SECTION ONE BLOCK DIAGRAMS

A500
BLOCK DIAGRAM

68000 CPU -

The 68000 will handle system boot-up as well as I/O access, which includes the expansion bus and the real time clock.

FULL 68000 BUS -

This line depicts the unbuffered address bus, data bus and the control lines.

REAL TIME CLOCK -

The Real Time Clock is located on the A501 expansion card. The 68000 will be used to read and write to this clock.

EXPANSION BUS -

Expandable up to 8 meg bytes, this bus is used to connect peripherals devices such as the A590 hard drive.

8520's CIA CHIPS -

There are two 8520 chips which handle the Keyboard I/O, Parallel port data and control lines, Disk controls and Serial port controls.

Notice that the 8520's are on the Fast 7mz bus, but their access time is only 1mz. The 68000 generates a line called E Clock with a frequency of 715,000hz. This will amount to ten wait states in the processor during an I/O access to a slow peripheral device.

GARY -

Gary which stands for Gate Array, handles the bus control lines for the CPU. (see Gary Block diagram for more information)

ADDRESS BUS -

This 23 bit bus is generated either by the 68000 or an external bus master.

ROM -

Known as the kick-start Rom, contains the necessary code to bring up the system.

DATA BUS -

The Amiga contains two data buses a fast processor bus (7mz) and a slower display bus (3.58mz). The fast bus connects directly to the Expansion Bus, Rom and the Bi Directional Tri-state latch assembly.

BI DIRECTIONAL TRI STATE LATCH ASSEMBLY -

This array of buffers and latches will enable the processor to read and write to the slow display bus. During a CPU read data from the display bus is latched via a strobe generated by Gary. Next Gary will enable the output buffers placing latched data onto the fast processor bus. However when the processor writes data to the display bus its buffered.

A500
BLOCK DIAGRAM

DISPLAY RAM DATA BUS -

This slow (3.58mz) bus will connect the CPU, Fat Agnus, Display Ram, Denise, and Paula together. The Display bus is used by the custom chips during a DMA cycle and by the CPU.

MULTIPLEXED ADDRESS BUS -

This bus is generated by the 18 bit address generator internal to Fat Agnus. Multiplexed down to 9 bits this bus is used for Ram addressing ether by the CPU or the Agnus during a DMA cycle.

DISPLAY RAM -

Located at $00000 - $7FFFF this 521k block will support the video graphics. An extended 512k block of ram siting at $C00000 - $C7FFFF known as fast ram, is used by the 680000. However with the Fatter Agnus chip installed this ram will map to $080000 - $FFFFF. This one meg block will now be used for chip ram.

FAT AGNUS -

Agnus stands for Register Address Generator. Basically Agnus will generate addresses for all twenty five DMA channels. (see Fat Agnus Block Diagram for more information)

REGISTER ADDRESS BUS -

This bus is driven ether by Agnus or the 68000 and is used to access registers out side Fat Agnus. During a DMA cycle this bus is driven by Fat Agnus. When no DMA is occurring this bus is driven by the lower eight bits of the 68000, thus allowing the 68000 to Read or Write to the Custom Chips.

DENISE -

Denise is the Display enable chip. She will output four digital levels of Red, Green and Blue, for a possible 496,000 colors. Also Denise handles the decoders for the Mouse port. (see the Denise block Diagram for more information)

VIDEO HYBRID -

The video hybrid, an Analog to Digital converter. Here the twelve lines of video information are decoded to Analog signals (approximately one volt levels). Also composite sync is input and added to the decoded video information out putting a Composite Monochrome signal.

PAULA -

Paula stands for Port, Audio, Uart and Disk. Here the disk data and the serial port data signals are processed by Paula, as well are the stereo audio output signals. Paula also handles the Pot Ports. (see the Paula Block Diagram for more information)

FAT AGNUS
BLOCK DIAGRAM

The primary function of Agnus is a RAM Address Generator and register address encoder which produces the addressing for the twenty five DMA channels.

In the block diagram the output side of the DMA controllers contain a number indicating the number of DMA channels driving the register address encoder and the ram address generator.

REGISTER ADDRESS ENCODER -

This is a basic PLA type of structure that produces a predetermined address on the RGA bus whenever one of the DMA channels is active.

RAM ADDRESS GENERATOR -

This contains an eighteen bit pointer register for each of the DMA channels. It also contains pointer restart (backup) registers and jump registers for six of the channels. A full 18-bit adder carries out the pointer increments and adds for jumps.

PRIORITY CONTROL LOGIC -

Here the DMA request will be processed based on their programed priority and sync counter time slot. Blit will then drop low requesting the bus from the processor.

SPRITES -

There are eight individual sprite controllers each operating on its own DMA channel. Sprites are line buffered objects which move very fast because they are controlled by hardware registers and comparators.

Each sprite is sixteen pixels wide and any number of lines high. You can choose from three colors and transparent which can be combined for a verity of colors. Each has a horizontal position register as well as a vertical start position register and a vertical stop position register. This allows for variable vertical size sprites.

The sprite position and image data is fetched automatically from the 512k video ram by the controller.

Sprites can run in DMA mode or under control of the microprocessor.

Sprites can be reused vertically as often as desired, however horizontally they need to be controlled by the microprocessor.

COPPER -

The copper is a co-processor that uses one DMA channel to fetch its instructions. Its three instructions are Move, Wait, and Skip.

Move - data to a register.
Wait - until the electron beam passes a given location.
Skip - past the next instruction if the electron beam is past a given location.

FAT AGNUS
BLOCK DIAGRAM

AUDIO -

There are four audio channels, located outside the DMA controller. Each controller is independent and on its own DMA channel. During blanking time the controller (Paula) will pull the DMAL request line low for DMA access.

BITPLANE -

The bitplane controller uses six DMA channels to handle the six independent bitplanes. Display data is transferred from memory to the display buffer during display time by the controller.

Each bitplane can be a full image or a window of an image that is four times the size of the screen. They can be grouped into two images, each with its own color registers.

F-DISK -

The disk controller uses one DMA channel. The controller uses this time slot for data transfer and can read or write a block of data up to 128k anywhere in the 512k memory.

MEMORY REFRESH -

The refresh controller uses a single DMA channel with its own time slots. It places RAS address on the memory address bus during these slots, in order to refresh dynamic ram. Memory is refreshed on every roster line.

During the DMA no data transfer actually takes place. The register address bus is used to supply video synchronizing codes. At this time, RAS0* and RAS1* are low and CASU* and CASL* are inactive.

BLITTER -

The blitter uses four DMA channels, three sources and one destination. Once the blitter has been started these four DMA channels are synchronized and pipelined to automatically handle the data transfer without further 68000 MPU intervention. The images are manipulated in memory independent of the display (bitplane DMA).

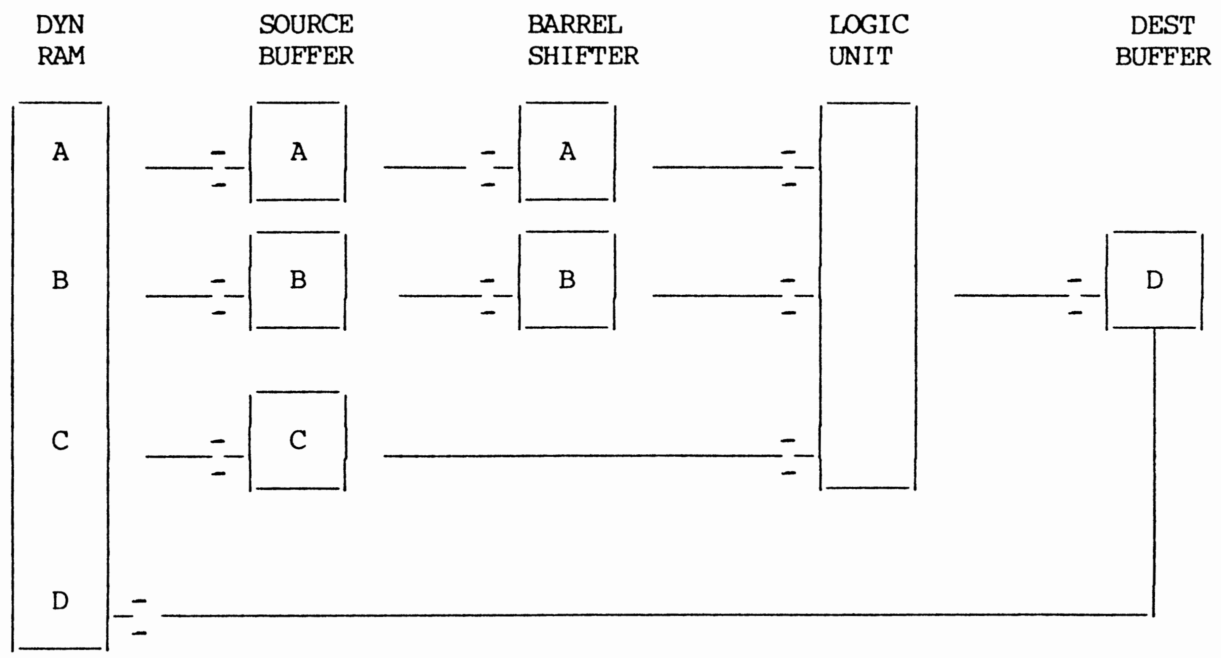
The blitter DMA controller is preloaded with the address and size of three source images (A,B, and C) and one destination (D) in dynamic ram. (see figure 1) These images can be as small as a single character or as large as twice the size of the screen. They can be full images or smaller windows of a large image. After one work of each source image is sequentially loaded into the source buffer (A,B, and C) they are shifted and then combined together in the destination buffer (D) and sent back to ram memory destination address.

This operation is repeated until the complete image has been produced. The unit has extensive piplining to allow for shifter and logic unit propagation time, while the next set of words are being fetched.

A controller register determines which of the 256 logical operations is to be preformed as the source images are combined and how far they can be moved (barrel shifted). In addition to the image combining and movement powers the blitter can be programed to do line drawling or area fill between lines.

FAT AGNUS
BLOCK DIAGRAM

FIGURE 1



DENISE
BLOCK DIAGRAM

Denise is stands for Display Encoder. Its primary function is to buffer display data, select the object to be displayed at any instant and encode the object into red, green, and blue color codes.

BIT PLANE SERIAL -

Here the bit plane data is continuously loaded and serialized during display time.

SPRITE SERIALIZE -

Unlike the bit plane data the sprite data is loaded during blanking time.

SPRITE POSITION COMPARE LOGIC -

Sprite data is serialized when ever an equality is detected between the sync counter and the sprite position register.

PRIORITY CONTROL LOGIC -

Here six lines of bit plane serial data and eight pairs of sprite serial data are input. However only one of the sprites or one of the separate bitmap images will be selected, producing the five bit color code.

COLOR SELECT DECODE -

The five bit code will select one of the thirty two color registers, producing the twelve bit RGB, (Red,Green,Blue), video output.

COLLISION DETECT LOGIC -

The six lines of bit plane serial data and the eight pairs of sprite serial data are feed into the collision detect logic which detects Real Time Occurrence between them and sets the appropriate bits in the Collision Storage Register. The 68000 will read this register then clear it.

MOUSE COUNTERS -

The two Mouse/Joystick connectors are controlled by the four mouse counters. The mouse counters count the horizontal and vertical motion of the controllers. These counters are read by the 68000 MPU.

PAULA
BLOCK DIAGRAM

Paula is the Port, Audio and Uart chip. Its main function, in chip area is the four audio channels. It also contains the I/O ports, (Disk and Pots), Serial Port (Uart), and the Interrupt Control and Status Structure.

D TO A CONVERTERS -

The four audio channels each have a DMA pointer register, data register, period, (frequency), register and volume register. Each channel has an on chip D to A (digital to analog) converter on the output. The four channels are grouped into a right and left audio output.

DISK CONTROL -

The disk controller has registers for data read, data write and control. It also contains a Precompensation Output circuit, a Data separator input circuit with a digital phase lock loop.

UART CONTROL -

The serial port uart included in Paula contains Data registers, Control registers, Transmit, (TRN), and receive, registers.

POT CONTROL -

The four pot ports are general purpose I/O ports. They have counters for simple A to D (digital to analog) conversion of an external capacitor charging , which could also be used for analog joystick controllers.

INTERRUPT CONTROL -

The audio, disk and uart controllers all set their own Interrupt Status register bits.

DMA REQUEST LOGIC -

The audio and disk controllers also go to the DMA request logic, (remember: they are DMA users), causing the DMAL signal to request DMA cycles from Agnus.

GARY
BLOCK DIAGRAM

ADDRESS DECODER -

Eight bits of the 68000 address bus will be decoded into the Ram enable (RAME), Register enable (RGAE), and Rom enable (ROME).

The Address Strobe (AS) will validate the address lines input to Gary on the trailing edge.

The Upper and Lower Data Strobes (UDS/LDS) will indicate valid data during write data or enable read data. UDS* will validate data bits 8-15. LDS* will validate data bits 0-7.

On power up Overlay (OVL) will be asserted relocating Rom to zero page ram.

Processor read/write* (R/W) will indicate the direction of the data bus transfer.

Expansion board present (EXPEN) looks for a low input from the A501 ram expansion card. If detected Gary will then configure to 1 megabyte.

Output from Gary is the Real Time Clock Read and Write (NRTC/NWTC), this will enable the data read and write to the real time clock.

Ram Enable (RAME) will output indicating that the 68000 is accessing the display ram bus.

Register Enable (RGAE) output low when the 68000 is accessing the registers inside the custom chips.

Rom Enable (ROME) will drop low selecting the internal Rom.

BUS CONTROL -

The DMA controller will pull the BLIT (Block Image Transfer) line low when requesting the bus from the MPU. Gary will generate DTACK* (Data Transfer Acknowledge) holding the processor off.

External Ready (XRDY) will signal Gary to hold off DTACK*, in order to generate wait states for I/O or slow memory access.

Override (OVR) is used to take over system decoding as well as turn off the DTACK* signal.

Output Enable Latch (OEL) will drop low buffering latched video data to the CPU during a CPU read.

Output Enable Buffer (OEB) will drop low buffering write data from the CPU to the display data bus.

Latch will strobe low latching data from the display bus to the CPU during a CPU read cycle.

(bus control continued)

Data Transfer Acknowledge (DTACK) will drop low allowing the processor to complete read or write cycles. However when DTACK is set high the CPU is put into a wait state.

When the Blitter slow down (BLISS) signal drops low the blitter operation is suspended allowing the processor control of the cycle.

FLOPPY CONTROL LOGIC -

Disk Write Enable (DKWE) is buffered, then output as Buffered Disk Write Enable (DKWEB).
Disk Write Data (DKWD) is inverted, then output as Disk Write Data Buffered (DKWDB).
Motor enable (MTR) is inverted, then output as External Motor enable * (MTRXD). However Select Zero (SEL0) is used to latch the motor enable signal,then output as the Motor Zero (MRT0D).

RESET CONTROL -

The System Reset and warm keyboard reset are buffered through Gary splitting into Reset (RST) and Halt (HLT). The CPU will be placed into a Halt state while reset is low.

EXPANSION BUS
CONTROL SIGNALS

COPCFG -

Coprocessor Configuration is an output from the coprocessor card. The Coprocessor has first priority in the configuration chain. The CONFIG in on each card is determined by the state of the CONFIG OUT of the previous card. When a card is configured its config output will be asserted, setting the config in of the next card. If the next slot is unpopulated on board-logic will pass the state of the last config out to the next config in.

OWN -

Also known as PIC (Plug In Card) because the card owns the current DMA cycle. This line will be asserted by the card when ever it becomes bus master.

INT2 AND INT6 -

These two interrupt control lines are shared with the 8520 chips on the mother board.

BFC 0:2 -

These Buffered Function Control lines indicate the current state of the CPU,eg; memory read/write, I/O read/write.

EINT 7,5,4 -

Expansion Interrupt Request lines from the expansion cards.

BERR -

Indicates a Bus Error on the expansion bus.

VPA -

Valid Peripheral Address - input to the 68000 indicating that the address has selected a slow peripheral device, so the 68000 should sync data to the E Clock.

E -

E Clock is generated by the 68000 and is running at 716.kz. This clock is uses to synchronize data from slow peripheral devices.

VMA -

Valid Memory Address - Generated by the 68000 to indicating a valid address for a slow peripheral device.

RST -

This bidirectional Reset line is used by cards with the ability to reset the system board.

EXPANSION BUS
CONTROL SIGNALS

HLT -

The system Halt line is used by the PICs to halt and tri-state the 68000 at the end of the current bus cycle.

EC3 -

This is a 3.58 MHZ clock synched to the rising edge of the 7.16 MHZ system clock.

ECDAC -

This is a 7.16 MHZ clock that leads the 7.16 MHZ system clock by 90 degrees.

EC1 -

This is a 3.58 MHZ clock synched to the falling edge of the 7.16 MHZ system clock.

OVR -

The Override is used to turn off the on board system decoding of system memory ranges, including those used by the Amiga custom chips. As a result of this, it can also turn off internally generated things like DTACK.

XRDY -

The External Ready provides a means for an external device to delay the mother board generated DTACK, for things like slow memory and I/O boards that need to add wait states.

BA 1:23 -

Buffered Address bus from the 68000 able to address up to 16 Megabyts. However only 8 Meg is available.

BGACK -

When a PIC receives a Bus Grant from the 68000 it will assert the Bus Grant Acknowledge as long as the DMA continues, releasing it once the DMA request is complete.

BD 0:15 -

This is the 68000s Buffered Data bus, used for data word or for byte access.

DTACK -

Input to the 68000, the data Transfer Acknowledge signal will indicate the completion of the data transfer. Amiga logic normal generates this signal for a simple no-wait state cycle (this may be varied by the custom chips).

READ -

Read is a buffered Read/Write* control line from the 68000.

EXPANSION BUS
CONTROL SIGNALS

BLDS -

This is a Buffered Lower Data Strobe. When low it validates the lower byte.

BUDS -

This is a Buffered Upper Data Strobe. When low it validates the upper byte.

BAS -

This is a Buffered Address Strobe. On the falling edge it indicates that the addresses are valid. However the rising edge indicates the completion of a bus cycle.

E7M -

This is the 7.16 MHZ system clock.

DOE -

This Data Output Enable signal is used by an expansion card to enable buffers on the data bus.

BUSRST -

Unidirectional this buffered Bus Reset will be used to reset the PICs.

GBG -

When in 68000 mode this Generic Bus Grant is essentially a buffered bus grant. However when the Coprocessor is in charge GBG becomes a Coprocessor Bus Grant.

INT1 -

Interrupt Request 1 from the expansion bus.

EBG 1:5 -

Bus Grant is generated by the bus arbitrator (Buster) in response to a bus request from any one of the PICs 1 thru 5, needing control of the bus.

EBR 1:5 -

Bus Request is generated by a PIC when its requesting the bus from the processor. This signal is an input to Buster.

SLAVE 1:5 -

Each card has a Slave output signal. This line is asserted when ever a card is responding top a decoded address. Busters collision detect circuitry will input the slave signal generating a bus error if multiple cards respond.

SECTION TWO EXPANSION CARDS

A2052/A2058A RAM EXPANDER OVERVIEW

1. the 2052 Ram Expansion card holds up to 2 megabytes of memory, using 256K byte Ram chips.

2. The 2058 Ram Expansion card holds up to 8 megabytes of memory, using 2 megabyte Ram chips. It is sold with 2 megabytes of Ram and can be populated up to 8 megabytes of memory.

A2020/A2010 INTERNAL DRIVES OVERVIEW

1. These will be DEALER-INSTALLED add-ons for the A2000.

2. The right hand 3.1/2" floppy drive is always configured to be recognized as DF0: by Amiga DOS.

3. The left hand 3.1/2" floppy drive (A2010) can be configured as DF1: or DF2:.

   Note that this drive must be hard set to D1.

4. The A2020 5.1/4" floppy drive can only be connected to the Bridgeboard.

A2090/A2090A HARD DISK CONTROLLER OVERVIEW

1. One SCSI Interface supporting up to 7 SCSI devices daisy-chained together. Two connectors are provided:

    1. 25 pin Macintosh compatible connector.

    2. 50 pin Standard SCSI connector.

    Note that the first SCSI device is Software selected as DH0: However the Hard setting is D0.

2. Two ST-506 interfaces, supporting up to 2 ST-506 compatible devices.

    Note that the First ST-506 device is Software selected as DH0: However the Hard setting is D0.

3. PREP - The program required to identify and prepare a hard disk for usage by Amiga DOS.

4. MOUNT - The program required to make Amiga DOS aware of additional partitions of hard disk and other devices not recognized at boot time.

A2088 BRIDGEBOARD
OVERVIEW

1. Physical configuration:

    1. 8088 Microprocessor running at 4.77MHZ (standard).

    2. Slot for optional 8087 Math Coprocessor.

    3. External floppy disk drive port - for a 1020 (5.1/4") or 1010 (3.1/2"). Floppy drives connected in this way can read MS-DOS formatted disk directly.

    4. Pheonix Bios.

    5. 512k System Ram.

    6. 128k "Special" Ram - 64K for mono (MDA) and color (CGA) displays, and 64k for communication between Amiga DOS and MS-DOS.

2. The PC Mono program _ this emulates the PC monochrome adaptor.

3. The PC Color program - this emulates the PC CGA (color graphics adaptor).

MS-DOS outputs text/prompts to a display adaptor. This is usually a monochrome text display adaptor (MDA). To switch MS-DOS's I/O to a different display adaptor, you must use the Mode command.

1. Mode CO80 <switches to color graphics adaptor>

2. Mode MONO <switches to monochrome text adaptor>

Notes:

1. Since the displays are generated by the Amiga, additional controls such as color (number available and selection), cut and, paste, ect. are available through Intuition menus.

2. Display modes such as Hercules , EGA, PGA, ect. are not emulated, but are available by simply installing the appropriate adaptor in one of the PC card slots.

3. The Bridgeboard has an 8088 which is the microprocessor used in an IBM PC or PC/XT.

4. The 2 leftmost PC slots on the A2000 are XT style slots. The rightmost PC slots have the additional AT style connector as well. The left two slots can also accept AT cards if the additional connector is added to the board. The signals and holes are already present, but connectors were not supplied so to allow full depth XT style cards to fit.

A2286 BRIDGEBOARD
OVERVIEW

1. Physical configuration:

   1. 80286 Microprocessor running at 8MHZ (standard).

   2. Socket for optional 8087 Math Coprocessor.

   4. 32K Rom Bios.

   5. 1024k System Ram.

   6. 128k "Special" Ram - 64K for mono (MDA) and color (CGA) displays, and 64k for communication between Amiga DOS and MS-DOS.

2. The PC Mono program _ this emulates the PC monochrome adaptor.

3. The PC Color program - this emulates the PC CGA (color graphics adaptor).

MS-DOS outputs text/prompts to a display adaptor. This is usually a monochrome text display adaptor (MDA). To switch MS-DOS's I/O to a different display adaptor, you must use the Mode command.

1. Mode CO80 <switches to color graphics adaptor>

2. Mode MONO <switches to monochrome text adaptor>

Notes:

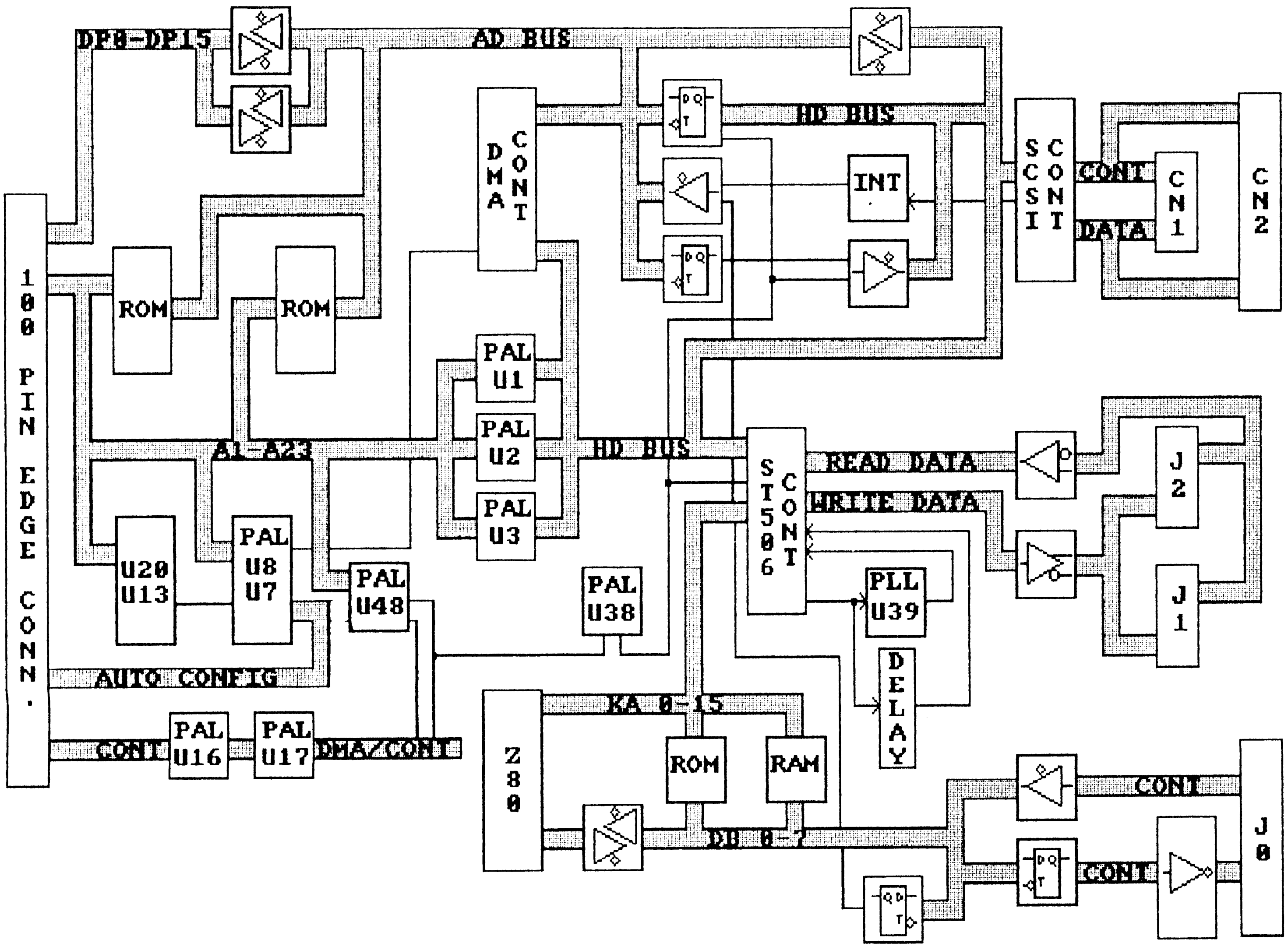
1. Since the displays are generated by the Amiga, additional controls such as color (number available and selection), cut and, paste, ect. are available through Intuition menus.

2. Display modes such as Hercules , EGA, PGA, ect. are not emulated, but are available by simply installing the appropriate adaptor in one of the PC card slots.

3. The Bridgeboard has an 80286 which is the microprocessor used in an IBM PC or PC/AT.

4. The 2 leftmost PC slots on the A2000 are XT style slots. The rightmost PC slots have the additional AT style connector as well. The left two slots can also accept AT cards if the additional connector is added to the board. The signals and holes are already present, but connectors were not supplied so to allow full depth XT style cards to fit.

A2090A HARD DISK CONTROLLER



A2090A
BLOCK DIAGRAM

1. ROMS -

These Roms allow the 2090A controller to boot the Amiga from DH0: provided there isn't a boot disk in drive DF0:.

The Auto-Boot Roms will only configure in a system with 1.3 kick-start and above. If the system contains 1.2 kick-start then the Aouto-Boot Roms need to be removed, otherwise the system will fail during configuration.

2. U20 - DLATCH, U13 - 8 BIT COMPARITOR -

During the Auto-Configuration process these two chips will identify which slot the card is located in.

3. PAL U8 -

This PAL will handle control lines necessary for Auto-Configuration.

4. PAL U7 -

This PAL will handle the control lines for the Auto-Configuration hardware.

5. DMA CONTROLLER -

The DMA controller is a custom LSI chip made by Commodore with byte to word funnelling and a built in 64 byte FIFO (First In First Out). The internal 64 byte FIFO permits real time data transfer to and from the host without holding the bus for a entire sector transfer.

The 68000 or the Z80 will use control lines PCSS and PCSD to interface with the DMA controller. PCSS is strobed first in order to strobe in the state of the HD bus. This state will determine the function to be preformed on the next PCSD strobe. However not all valid states require a PCSD strobe to be read.

The DMA controller will use the Bus Request line to request the bus from the 68000, in turn the 68000 will set the Bus Acknowledge line Low indicating that the bus has been granted. PAL chip U16 will translate these control lines to Amiga DMA protocol.

6. PAL U16 -

This PAL is used to interface the DMA controller with the 68000 bus.

7. PAL U17 -

This PAL handles control lines that interface the DMA controller with the 68000 and the on board Z80.

8. PAL U1,U2 and U3 -

These PALs are configures as 74LS461 Octal Counters. Written in three the DMA controller will pre-set the starting address into these counters before each transfer. The MSB (corresponding to A24) will set the direction of the Read Write control line on the host during a DMA transfer.

9. U9 DLATCH -

Write Command Block Pointer (WCBP) will strobe this DLatch, latching in data from the command block. The Z80 will read this data by pulling AI/O1 Low.

10. U11 BUFFER -

By controlling RDSTST1 (read status) and RDSTST0 (read status 0) The 68000 can read the condition of control lines such as Interrupt Pending and Controller Command Block Pointer (CCBP).

11. U12 DLATCH -

The 68000 will generate both SCSI and INTERRUPT control lines which are latched in when WRSTAT0 (write status 0) strobes Low.

12. U22 BUFFER -

This chip will buffer control signals such as INTEN (interrupt enable) and SSEL (SCSI select). The Z80 will read these control lines by pulling AI/O2 Low.

13. INTERRUPT -

Made up of U5 (AND), U6 (inverter) and U10 (buffer) this network will enable Interrupt 2 to the 68000 bus.

14. SCSI CONTROLLER -

This hard disk controller made by Western Digital supports full SCSI protocol. The WD33C93 is supported with a flexible architecture allowing either the 68000 or the Z80 to control its operations.

This controller is capable of handling up to 7 SCSI hard drives.

15. PAL U48 -

This PAL will handle control lines necessary to interface the SCSI controller.

16. U47 BIDIRECTIONAL BUFFER -

Controlled by the 68000 this buffer will connect the SCSI data bus to the host bus thus allowing the 68000 to control the SCSI controller.

17. U14 and U15 BIDIRECTIONAL BUFFERS -

Controlled by PAL U16 these buffers will connect the DMA controller to the 68000 data bus (D0-D15).

18. Z80 -

This processor will support system power-up and the ST506 controller. The Z80 will generate the control lines necessary to operate 2 ST506 devices. Also this processor will handle system power-on diagnostics. (see service manual for complete information on the diagnostics and error codes).

19. ROM -

This 8K Rom will support the Z80 in system power-up and in the ST506 protocol.

20. RAM -

This 2K Ram will support the operations of the Z80.

21. U31 BUFFER -

This buffer will support the Z80s bidirectional buffered data bus. The Read/Write line will control the data direction. The output is enable by the DMA line when a DMA cycle is not in operation.

22. ST506 CONTROLLER -

This controller will support the read write protocol for the ST506 interface. The DJC can operate in MFM (modified frequency modification) or NRZ (non-return to zero). Decoding for NRZ can be handled internal to the chip, however MFM decodes need to be handled externally.

23. U39 PLL -

This chip will work with the DJC to convert the MFM data input to an NRZ output during a read data mode.

24. U41 DELAY

During an output mode this chip will delay the write data by 12ns or 24ns preventing over crowding of bits as the heads move towards the center of the platters.

25. U27 COMPARITOR AMPLIFIER -

This chip is configured as a pulse peak detector. Data written to the disk is broken down into One Bits with varying pulse widths. The time (pulse width) between One Bit transitions will indicate the existence of a zero bit(s). The comparitor will input this data, out putting a high when ever a One Bit has been detected.

26. U26 DIFFERENTIAL AMPLIFIER -

Serial data output by the DJC is split into two different current paths through the recording heads. This process will create the pattern of One Bits on the disk medium with proper spacing representing Zero Bit(s).

27. U23 BUFFER -

This chip will buffer the control lines to the Z80 data bus incoming from the ST506 connector.

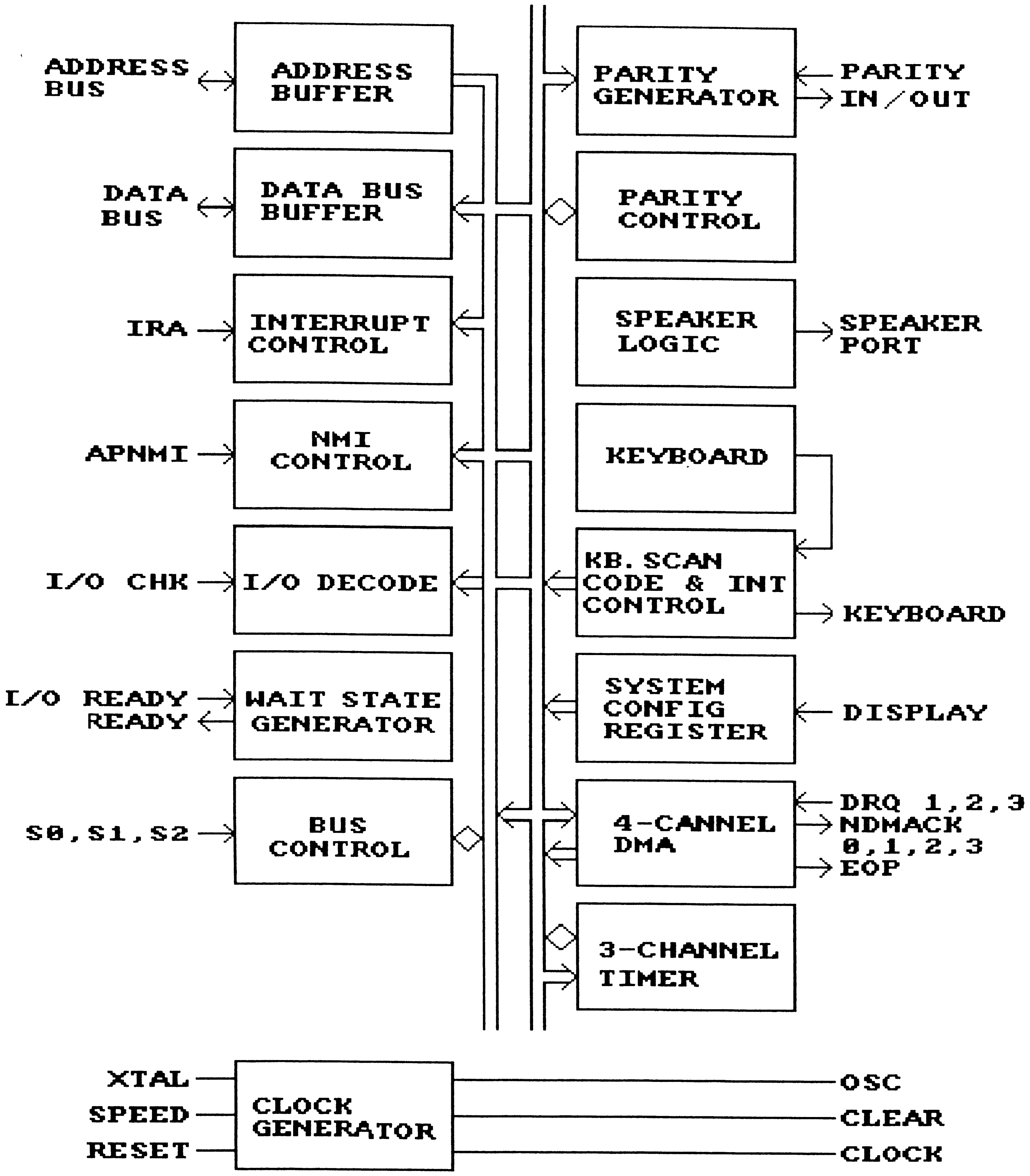
28. U24 DLATCH -

This chip will latch control lines from the Z80 data bus to the ST506 connector.

29. U25 PAL -

This PAL chip will latch control lines necessary to interface the ST506 controller to the DMA controller.

BLOCK DIAGRAM
FE2010A



A2088 EMULATOR
FE2010A

1. INTERRUPT CONTROLLER

The interrupt controller is equivalent to an 8259A programmable interrupt controller. This will control which I/O device is being serviced by the processor.

2. WAIT STATE GENERATOR

This will generate the necessary wait states fro CPU, I/O, and DMA operations. The number of wait states will vary with the processor clock speed.
It also synchronizes the external ready which can be used to generate wait states.

3. BUS CONTROL

This is equivalent to an 8288, which generates controls for CPU operation. Memory decodes including the generation of RAS and CAS are provided by this block.

4. CLOCK GENERATOR

This is equivalent to an 8284A, and will generate the system clocks.

5. TIMER

Equivalent to an 8253 programmable interval timer, this will handle the time of day clock, the ram refresh, and the speaker data.

6. DMA

Equivalent to an 8237 DMA controller, handling all DMA operations. Note that DMA channel Zero is internal for ram refresh.

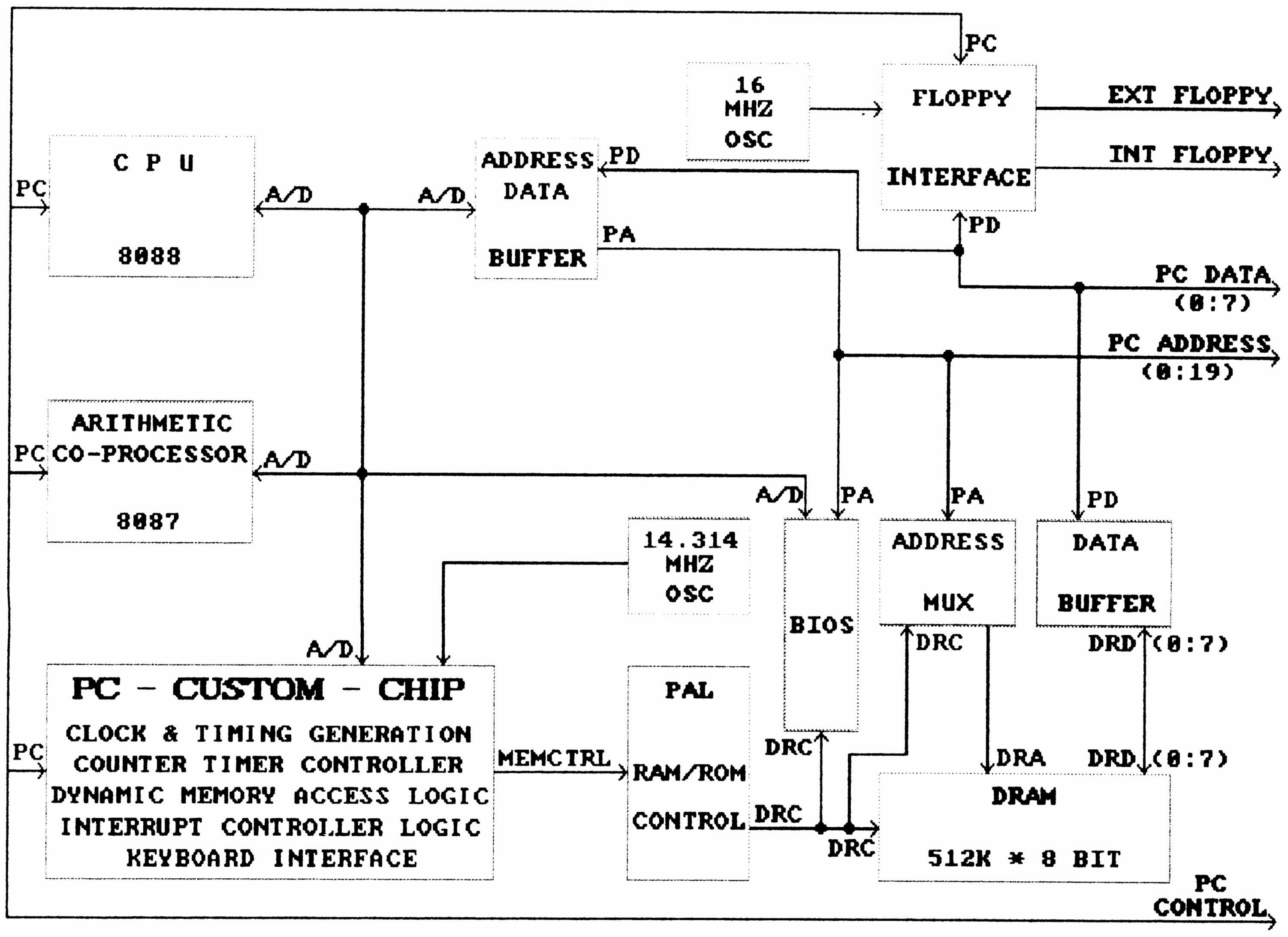
7. SPEED SELECT

This generate the Three CPU clock speeds. 4.77, 7.15, and 9.54 MZ.

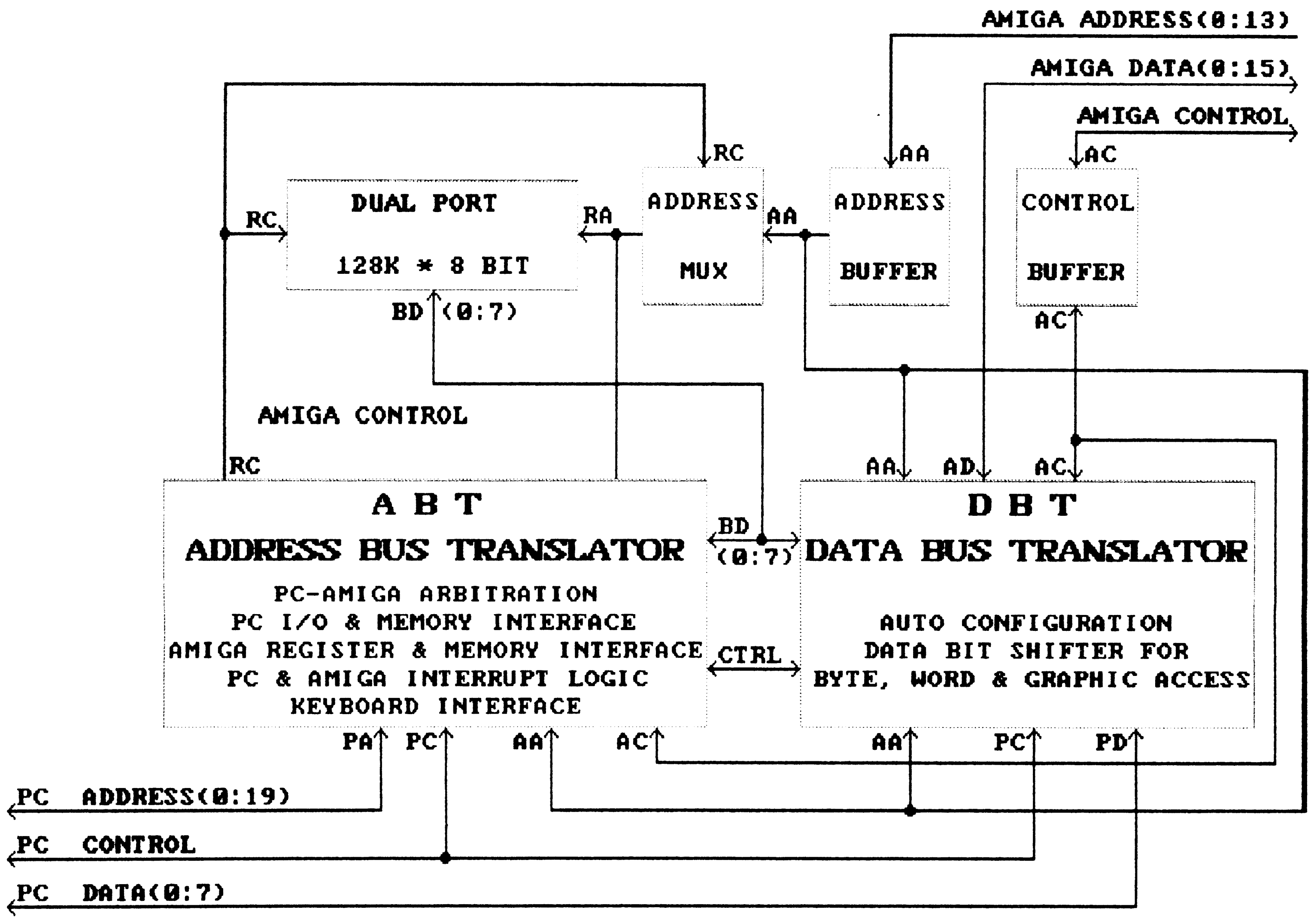
8. PIO

Equivalent to an 8255 PIO (programmable peripheral interface) this will handle speaker data, keyboard ports, and enable error checks.

A2088 PC/XT EMULATOR



A2088 PC/XT EMULATOR INTERFACE



2088 XT EMULATER
BLOCK DIAGRAM

1. 8088 PROCESSOR

This is an 8 bit microprocessor supporting system start-up. The 8088 has two bus request lines, RQ/GT0 and RQ/GT1. The first bus request is from the bus controller (FE2010) which has high priority. The second bus request is from the 8087 math coprocessor.

2. ADDRESS BUS

This unlatched bus will connects the 8088, 8087 and the FE2010 together.

3. DATA BUS

This unbuffered data bus connects the 8088, 8087 and the FE2010A together.

4. ADDRESS LATCH

This block is made up of three 74LS373 latches. Because address bits A0-A7 are shared with data bits D0-D7 the latches must only be enabled during a vailed address cycle. ALE (address latch enable) generated by the FE2010 will strobe high whenever the address is valid latching the current address to the system address bus.

5. DATA BUFFER

This block is made up of a 74LS245 bidirectional buffer. The output control as well as the direction are controlled by the FE2010.

6. LATCHED ADDRESS BUS

This 20 bit address bus (A0:A19) supports addressing to the Bios Rom, Ram and the Expansion Bus connector.

7. BUFFERED DATA BUS

This 8 bit bus supports the Bios Rom, Ram and the Expansion Bus connector.

8. ROM

This is a Phoenix Bios Rom supporting system power-up and operation.

9. FE2010

Made by Faraday this bus controller will support Ram decoding, DMA channels, IRQs, Memory and I/O controls and the system clock 4.77 MHZ. see FE2010 block for more information.

10. RAM

On board system Ram is 512k, expandable through the expansion bus.

11. FDC FLOPPY DISK CONTROLLER

This chip will handle the MFM data transfer to and from the disk medium. The FDC will support the protocol necessary to connect two floppy drives to the system with a storage capacity of 360k bytes (5 1/4") or 720k bytes (3 1/2"). Precompensation circuitry internal to the FDC will prevent over writing of bits as the head moves towards the collet hub. Data transfer to and from the controller is handled via DMA channel 2.

12. 128K SPECIAL RAM

Split into two 64k banks this Ram will interface the video, I/O and disk information with the Amiga bus. The lower 64k serves as disk buffer ram. The disk buffer is used to transfer sectors between two machines.

The upper 64K is broken down into 32 bytes for PC color memory, 8K is for monochrome and 16K is for parameter ram. This parameter ram serves two functions. First the two computers can set what is known as a lock byte when they are about to use the general purpose ram to transfer disk data, they will release this byte when the transfer is complete.

The second function is to store the parameter information the computers set when transferring a block of data. In other words, this parameter ram contains the controls necessary to transfer data from one computer to another.

The last 8K byte is used for I/O page ram. This ram is used to interface the I/O address of the PC to register addresses of the Amiga.

ABT ADDRESS BUS TRANSLATOR

This chip will handle control lines necessary to interface the PC bus to the Amiga bus. Below is a brief summery of these controls:

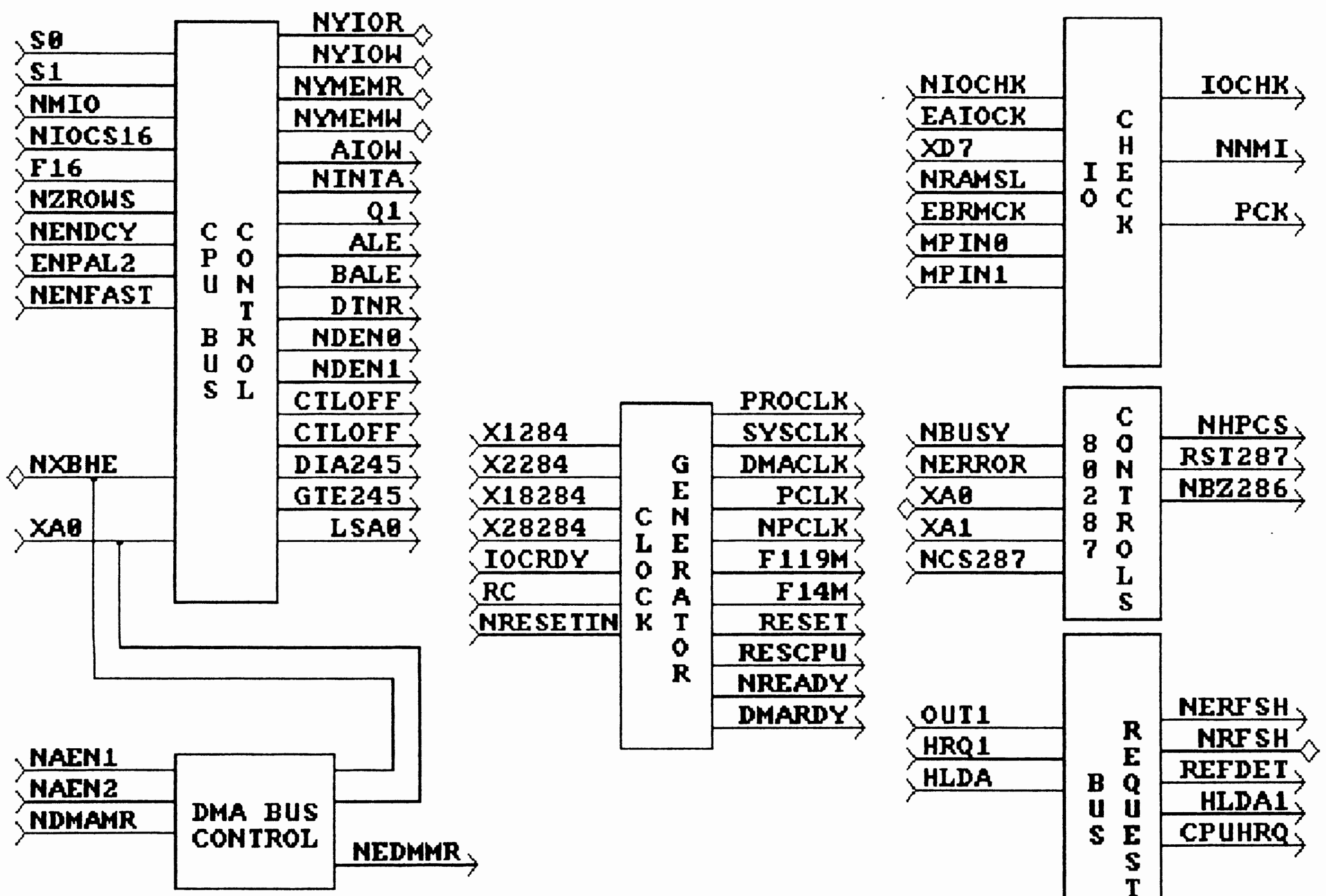
PC - Amiga Arbitration
PC I/O memory Interface
Amiga Register and Memory Interface
PC and Amiga Interrupt Logic
Keyboard Interface

DBT DATA BUS TRANSLATOR

These chip will handle the Auto-configuration controls. On power-up the 68000 will search the Expansion bus for cards. This chip will identify the size and abilities of the 2088 bridge card. Below is a brief summery of the DBTs functions:

Auto-Configuration
Data Shifter for Byte
Word Graphic Access

FE3000
CPU CONTROLLER



2286 EMULATOR
FE3000A

1. CPU BUS CONTROL

This block generates the bus controls for CPU, DMA, and refresh cycles. This includes data buffer controls, adddress latch controles, BHE and address 0 generation, I/O read and write signals, and memory read and write signals.

Controls are provided for Data bits 0-7 (CTLOFF, NDEN0, DTR), data bits 8-15 (NEDN1, DTR), and the byte swap buffer (CTLOFF, DIR245).

2. DMA BUS CONTROL

The Bus control block will enable the bus to do a DMA Memory Read. The DMA controller will assert the DMA Memory Read (NDMAMR) line, at the same time one of the two DMA Channel Enable (NAEN1 OR NAEN2) lines must also be asserted indicating the current DMA cycle.

3. CLOCK GENERATOR

This block contains two clock generators. The first generates clocks for the CPU, DMA, and 8042 keyboard controller. The clock generator runs at 6, 8, or 12 mhz. However the DMA clock is half the CPU clock i.e. 3, 4, or 6 mhz.

The second clock generator provides a 14.31818 mhz. clock for the PC bus and clock for the 8254 timer contained in the FE3010.

This block contains circuitry used to synchronize the CPU reset, CPU ready, DMA ready, and system reset.

A wait state generator is used to maintain compatibility with the PC and AT bus. The internal wait state generator may be disabled by means of a jumper (ENPAL2). An external wait state generator may be added by using the Q1 signal to start a wait state generator and sending a NENDCY to the FE3000A to end the cycle.

4. I/O CHECK

This circuit test for parity errors from on board ram or errors the PC bus. If an error is detected then an MNI is generated and sent to the 80286.

On board parity is checked by sending the two parity bits from the ram parity generator to MDPIN0 and MDPIN1. The EBRMCK and NRAMSL are used to enable the parity checking.

The IOCK signal indicates an error from the bus. It is generated by the EAIOCK signal.

5. 80287 BUS CONTROLS

Handles the I/O decodes for the 80287 coprocessor as well as the BUSY and ERROR signals. If an error is detected, an interrupt can be generated by using the NIRQ13 signal.

6. BUS REQUEST

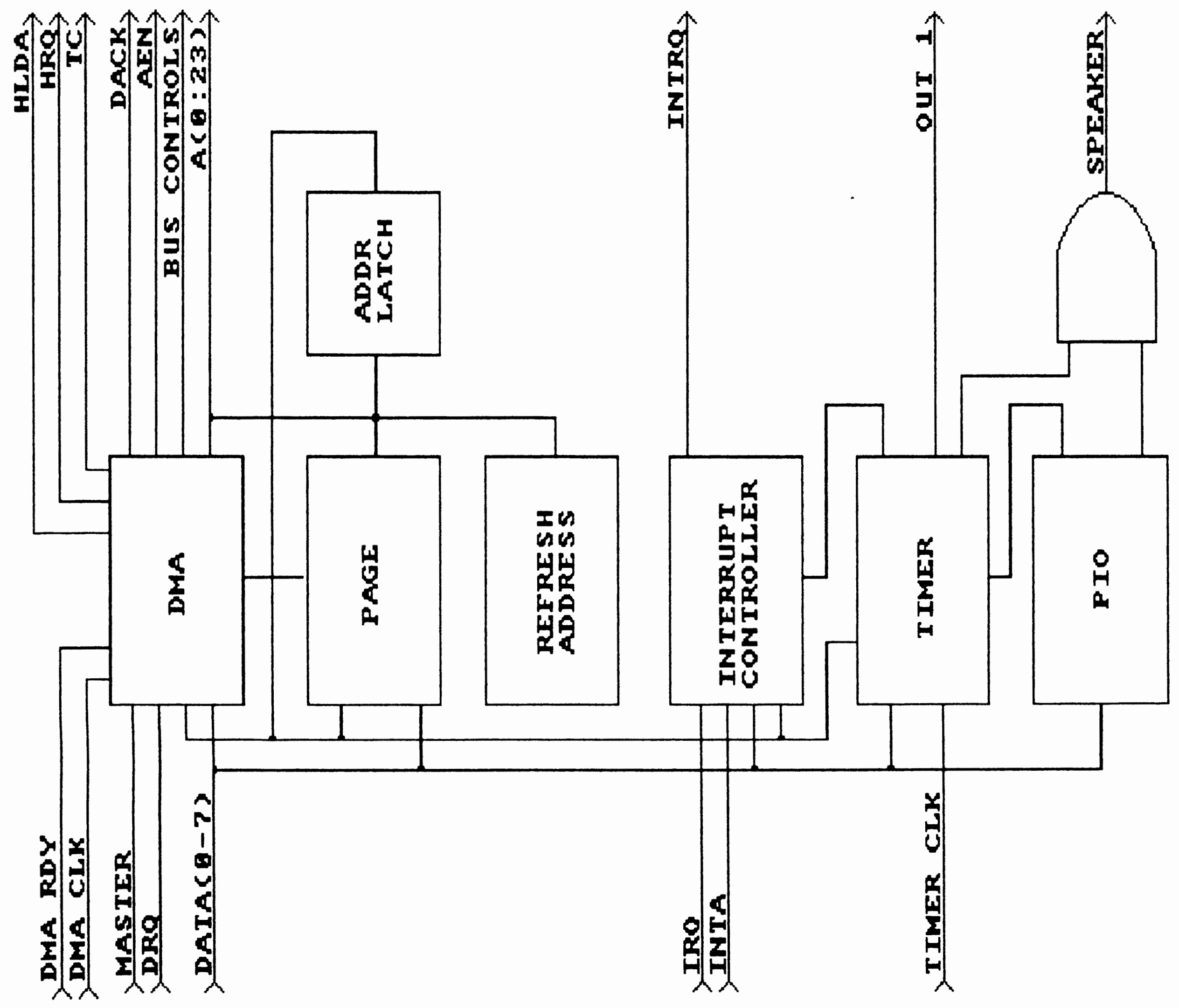
Handles the arbitration between a DMA request and a refresh request.

When a refresh or DMA request is made, a CPUHRQ will be sent to the 80286 and a HLDA will be received in reply.

The DMA controller (FE3010) will set HRQ1 high in order to request the bus from the 80286. HALDA is then set high indicating to the DMA controller (FE3010) that the bus is granted.

For a refresh request OUT1 is set high from timer 1 of the FE3010. NERFSH is then set low enabling the refresh address to the bus. NYMEMR is set low generating a memory refresh cycle.

FE3010
PERIPHERAL CONTROL



1. DMA

The internal DMA controller is equivalent to Two 8237 DMA controllers in cascade mode. This controller handles all seven independent DMA channels. DMA Acknowledge lines DMA0-2 will output a three bit code signaling the requesting device that their DMA has been granted. see figure 1

When in idle mode the CPU can read or write to internal register controllers with the use of the I/O Read, I/O Write lines and address bits 0-9. However in DMA mode these lines are outputs.

2. PAGE

This is an 8-bit by 16-byte dual port ram. When in 8 bit DMA mode address lines 16-23 are generated. However in 16 bit DMA mode and refresh cycle, address lines 17-23 will be generated.

One of the ports is a read only ram for DMA and Refresh cycles. The other port however is a read / write ram for the 80286 CPU.

3. INTERRUPT CONTROLLER

This is equivalent to two 8259 interrupt controllers supporting up to 15 interrupt control lines.

4. TIMER

This is equivalent to an 8254 timer. One channel will generate the ram Refresh Request, another will generate the sound and the third is tied to interrupt 0.

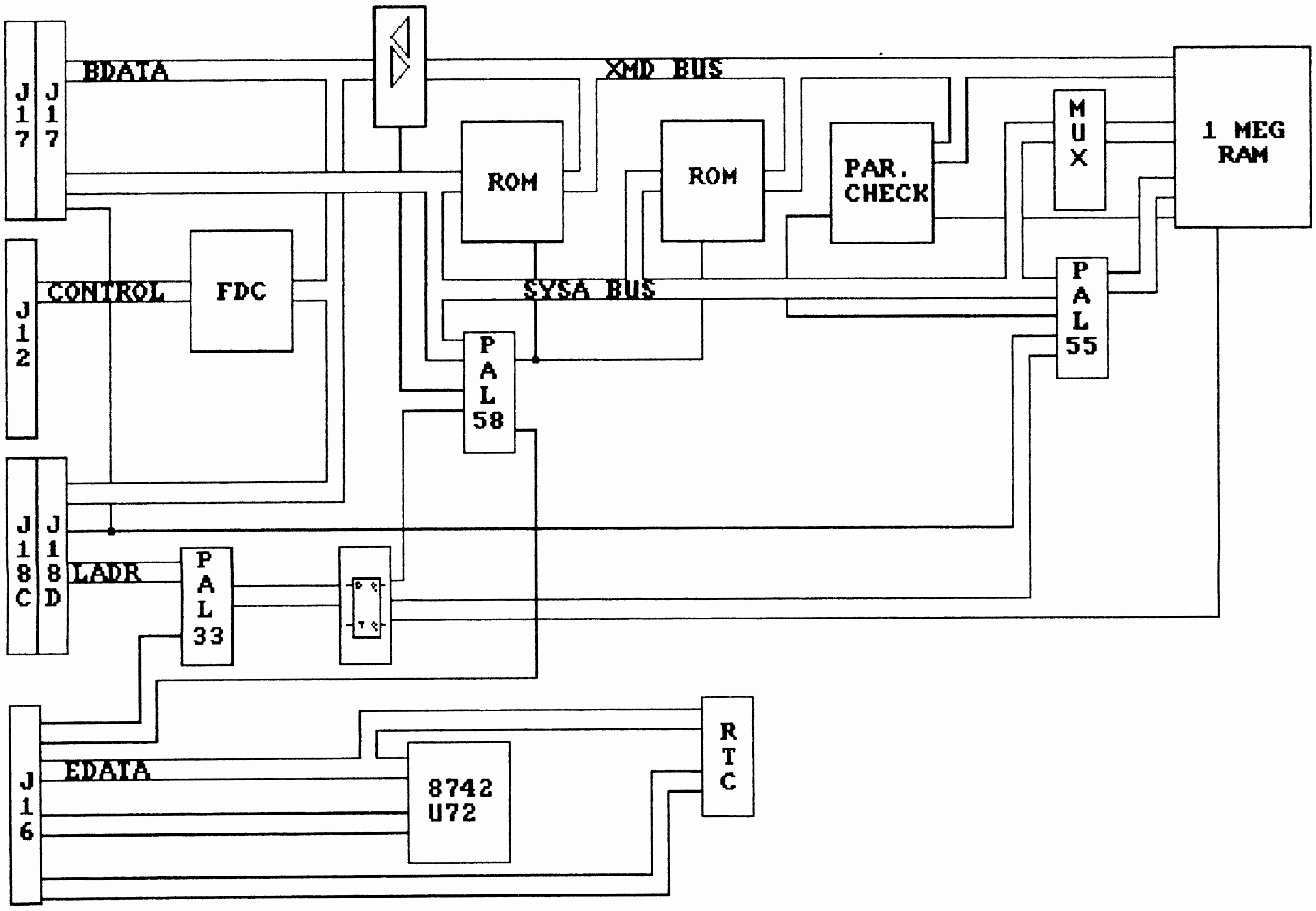
5. PIO

This control port will control the Speaker and Timer Channel. It also contains circuitry to detect if the Ram Refresh is running.

6. REFRESH ADDRESS

This block contains a 9 bit counter used to generate the Refresh Address.

A2286 PC/AT EMULATOR DAUGHTER BOARD



A2286 AT EMULATOR BLOCK DIAGRAM

1. 80286 MICROPROCESSOR -

This 16 bit processor will support system start-up.

Bus request are made by pulling the Hold Request line LOW, the processor will complete its current cycle then drop its Hold Acknowledge line LOW indicating that the bus has been granted.

2. ADDRESS BUS -

This unidirectional bus will interface the 68020 with the FE3010A Peripheral Controller for operations such as ran decoding. This bus will also interface with the Address Buffer block were it will interface with the System Address bus and the Latched Address bus.

3. ADDRESS BUFFER BLOCK -

This block generates the System Address Bus (SYS 0:19), the buffered LDAR 17:23 Address Bus and the Latched Address Bus (LA17:23). The System address bus is partially generated by U12 and U13 (74ALS646 Octal Transceivers). These chips will give address lines A1:A15 a bi-directional attitude. When the CPU is writing a valid address to the Bus, PAL chip U10 will generate Negated Address Latch Enable (EN). When EN is strobes High data is stored in internal registers of the octal transceivers. Master Not will control the direction of the address bus. A High state would indicate that the CPU is in charge. However when Low a DMA controller is in control of the address bus. In other words the buffer reads from B to A.

Address bits A16:A19 on the other hand are unidirectional. When the CPU is in charge and on the leading edge of EN (address enable) D-Latch U14 (74ALS573) will latch bits A16:A19 onto the System Address bus.

Buffer U15 (74LS254) will generate the LADR A17:A23 address bus. The direction of this bus is controlled by Negated Master. When the CPU is in charge Master Not is High and the bus reads A to B. However during a DMA cycle Master Not is Low and the buffer will read B to A. This Bus connects to the Daughter Board where its used for Ram decoding (RAS0 and RAS1) and Prom select.

The LA 17:19 is generated by D-Latch U16 (74ALS573) when the system is in CPU mode. This latch is strobed by BLAE (bus address latch enable) which is generated by the FE3000A by ORing ALE (address latch enable) and HLDA (hold acknowledge). This bus is connected to the Daughter Board but at present is not used.

4. DATA BUFFER BLOCK -

This block generates the Buffered Data Bus D0:D15 and the EData Bus 0:7. The control lines for this block are generated by the FE3000A. The low order byte (BD0:BD7) is buffered by Octal transceiver U19. Data direction is controlled by DTR (direction), when High the direction is A to B. However when Low the direction is from B to A. Control line D646EN (data 74ALS646 enable) will gate the output enable of octal transceiver U19. On the leading edge of DGATECTL (data gate control) data from the B bus is stored into internal registers of U19. When high, address bit A0 will transfer stored data from the B bus to the A bus. However when low the buffer will transfer data from A to B in Real Time.

Buffer U20 (74LS245) will buffer the high order byte to BD8:BD15. The data direction is controlled by DTR (direction). When DTR is high the buffer reads from A to B and when low the buffer reads from B to A. The output enable is controlled by C245EN (74LS245 enable).

Bi-directional buffer U24 will generate the 8 bit EData bus. When in CPU mode AEL address enable will enable this buffer for I/O access. Direction control is generated by PAL chip U21.

The Amiga and the PC can read and write to one another via Word Access. However when the PC writes a word, it writes the word as low byte /high byte. When the Amiga writes a word it writes high byte / low byte. In order to handle this problem Byte Swapping hardware has been added to correct this problem. The circuit is made up of U22 (74LS245) and the FE3000A. The FE3000A will generate F245EN (buffer enable) and F245DIR (buffer direction) necessary to control this Byte Swap buffer (U22).

5. PAL U5 -

This PAL has three functions outlined below:

1. DMA ADDRESS ENABLE -

Address enable (AEN) and DMACK 2 Low will generate Address enable 1 (AEN1). AEN and DMACK 2 High will generate AEN2. Input to the FE3000A AEN1 will enable DMA channels 0-3 and AEN2 will enable DMA channels 5-7.

2. ENABLE ADDRESS BIT 20 -

The 8742 keyboard controller will generate address bit 20 gate (A20GT). During a CPU cycle A20GT will enable Address bit 20, allowing the system to access over one mega byte.

3. DMACK 0-7 -

DMACK 0-2 will decode into DMACK 0-3 and DMACK 5-7, DMACK 4 is not used.

6. PAL U10 -

This PAL has six functions listed below :

1. ADDRESS LATCH ENABLE -

Address latch enable (AEL) is negative NAND with the synchronized address strobe outputing address enable (EN). During a CPU cycle EN Not will store the current address into internal registers of the address
buffer latches. Also EN Not is used to enable the output of address latch U14.

2. RESET

Reset is inverted to output a Reset Not.

The system needs to be able to communicate with the keyboard to preform task such as turning on and off LEDs. For example when the CAPS lock key is depressed the keyboard controller will transmit the key codes to the mother board. However the keyboard controller will not light the CAPS lock LED on its own. The mother board will transmit a control code to the keyboard telling the controller to turn the LED on. This will keep the system in sync with the keyboard operations, also it allows programs to control things like CAPS lock without depressing any keys.

Line number 3 and 4 explain the hardware necessary to give the data and clock lines a bi-directional attitude.

3. KEYBOARD DATA ENABLE (KBDATA) -

This line will define the state of the keyboard data line. Generated by the keyboard controller P27 will enable this buffer when ever the CPU is going to write to the keyboard. The input of this buffer is tied Low, whenever the output is enabled the CPU is writing a zero. However when P27 is High the output is High impedance. Depending on the current operation ether the CPU is writing a ONE bit (do to an external pull up resister) or the keyboard is writing to the bus.

4. KEYBOARD CLOCK ENABLE (KBCLCK) -

This line operates in the same manner as the KBDATA discussed above. The only difference is that this buffer is controlled by P26.

5. ENABLE I/O CHECK (ENIOCK) -

The value of EDATA3 is NAND with NRESET to output I/O Check. Preset to zero after a reset, ENIOCK will enable an internal gate of the FE3000A allowing it to process the I/O Check signal from the expansion bus.

6. ENABLE RAM PARITY CHECK (ENRAMPCK) -

The value of EDATA2 is NAND with NRESET to output Enable Ram Parity Check. Preset to zero after reset, the ENRAMPCK will enable the Ram Parity signals (MDATAP0 + MDATAP1) to be processed by the FE3000A.

7. PAL U6 -

This PAL serves as an address decoder for I/O functions. These functions are listed below:

1. ENABLE 74LS138 - (ENABLE 138) -

In CPU mode Enable 138 Not will enable address decoder U11 (74LS138). This decoder is used for Floppy access.

2. BUFFERED DATA BIT 7 (BD7) -

The CPU will open the I/O channel to check the disk drives Change Disk line (DCHG). During this cycle the PAL will output the DCHG as Buffered Data Bit 7.

3. KEYBOARD PROCESSOR READ (N8042R) -

With I/O read vailed, the CPU will generate N8042R enabling a read to the Output Data Buffers or the Status Register of the keyboard processor.

4. KEYBOARD PROCESSOR WRITE (N8042W) -

With I/O write vailed the CPU will generate N8042W enabling a write to the internal Input Data Buffer of the keyboard processor.

5. MATH COPROCESSOR CHIP SELECT (NCS287) -

The CPU will generate a vailed chip select for the coprocessor from $0E0 to $0FF.

6. NMI CHIP SELECT (NNMICS) -

With I/O write vailed the CPU will generate NNMI enable to the FE3000A.

7. NEGATED GATE PROGRAMMABLE PERIPHERAL INTERFACE (NGTPIO)

This enable signal is input to PAL U21 gating the Parity Check and the I/O Check signals.

8. NEGATED SELECT (NSELE) -

During an I/O access the CPU will generate this line to enable the direction control for the EData bus.

9. NEGATED SELECT CLOCK (NSELCHK) -

This line will clock EData bits 2 and 3 through U10 during an I/O write.

8. PAL U21

This PAL has 5 functions listed below:

1. DIRECTION (DIR) -

This line is driven by INTACK, SELE, and NPCS. Interrupt Acknowledge (INTACK) will be asserted while the CPU is servicing an Interrupt. This Low state will set the Direction control High allowing the CPU to read to the EData Bus. The CPU will set the state of the Select (SELE) line. When this line is Low the CPU is writing to the EData bus.

However when SELE is High then the CPU is reading the EData bus. Negated Coprocessor Chip Select (NPCS) will drop Low setting the Direction control line Low allowing the Co processor to write to the EData bus.

2. MASTER / ADDRESS ENABLE -

Negated Master is internally inverted to generate Master.
Master is negative NORED with Hold acknowledge to give the state of Address Enable (AEN). Wile in CPU mode AEN Low will enable an I/O access. However during a DMA cycle AEN is High, disabling the Edata bus.

3. REAL TIME CLOCK READ -

The product of I/O read (IOR) and Real Time Clock Chip Select (RTCCS) will output Real Time Clock Read (RTCR).

4. REAL TIME CLOCK WRITE -

The product of I/O Write (IOW) and Real Time Clock Chip Select (RTCCS) will output Real Time Clock Write (RTCW).

5. EDATA BITS 6 AND 7 -

A buffered I/O Check (IOCK) is output as data bit 6 when NGTPIO is Low. This will indicate an error on the expansion bus.

A buffered Parity Check (PCK) is output as data bit 7 when NGTPIO is Low. This will indicate an error in the Ran bank.

9. PAL U23 -

This PAL has 4 functions listed below.

1. DUAL PORT RAM ENABLE -

The product of Ram Write Enable (RWE) and Ram Enable (RAMEN) will output a Write Enable Signal. This signal will enable a write to the 128K of Dual Port Ram.

2. DELAYED ADDRESS BIT A0 -

Address bit 0 is synchronized with the system clock outputing Delayed A0 (DLYA0).

3. DELAYED I/O WRITE COMMAND (DLYWR) -

The product of Address Enable (AEN) and Enable Ram Refresh (ERFSH) will control the direction of the I/OW line. When Low the CPU is in charge so the FE3000A will input the Delayed I/O Write signal (YIOW) to pin 2 of the PAL. This will output on pin 18 as Delayed I/O Write (DLYWR). However when High a DMA cycle is in progress so the FE3010A will input DLYWR on pin 18 outputing it as YIOW on pin 19.

4. DEGLITCHED NMI (NMI) -

The NMI NOT signal is input at pin 6. Internally this signal will be synchronized with the system clock. Once inverted it will output on pin 15.

10. 128K SPECIAL RAM -

Split into two 64K banks this Ram will interface the Video, I/O and Disk information with the Amiga Bus. The lower 64K serves as disk buffer ram and is used to transfer sectors between the two machines.

The upper 64K is broken down into 32K bytes for PC color memory, 8K is for monochrome and 16K is for Parameter ram. This parameter ram serves two functions. First the two computers can set what is known as a lock byte, when they are about to use the general purpose ram to transfer disk data, they will release this byte when the transfer is complete.

The second function is to store the parameter information the computers set when transferring a block of data. In other words, this parameter ram contains the controls necessary to transfer data from one computer to another.

The last 8K byte is used for I/O page ram. This ram is used to interface the I/O addresses of the PC to register addresses of the Amiga.

11. ADDRESS BUS TRANSLATOR -

This chip will handle control lines necessary to interface the PC bus to the Amiga bus. Below is a brief summery of these controls:

PC - Amiga Arbitration
PC I/O Memory Interface
Amiga Register and Memory Interface
PC and Amiga Interrupt Logic
Keyboard Interface

12. DATA BUS TRANSLATOR -

This chip will handle the Auto-Configuration controls. On power-up the 68000 search the Expansion bus for cards. This chip will identify the size and abilities of the 2286 bridge card. Below is a brief summery of the DBTs functions:

Auto Configuration
Data Shifter for Byte
Word Graphic Access

A2286 AT EMULATOR
DAUGHTER BLOCK DIAGRAM

1. ROMS -

These Roms will handle system power-up as well as operations.

2. FDC (FLOPPY DISK CONTROLLER)

This is a Western Digital (WD37C65) floppy controller. The controller will Read and Write MFM (Modified Frequency Modulated) data to and from the disk medium. Connected to the IData bus, data transfers are handled via DMA. DMA Request 2 (DRQ2) will input to the FE3010 requesting the bus, DMA Acknowledge 2 (NDACK2) is generated by the FE3010 indicating that the bus is ready for the DMA transfer.
In non-DMA mode the controller will generate an interrupt to the processor when ever a data byte is available for transfer.

3. PAL 58 -

This PAL has four functions outlined below:

1. NEPROMSEL -

During a memory read in CPU mode NEP-OE will output low enabling the Bios Roms.

2. MEMDIR -

Memory Direction (MEMDIR) controls the direction of the XMD (memory data) bus. During a CPU Read this line is High, and Low when the CPU is writing to the XMD bus.

3. F16 -

While in AT mode F16 will go active High enabling 16 bit memory access.

4. RAS ENABLE -

RAS enable is a buffered output of Memory Read / Memory Write (RDWR). RAS enable gate both RAS0 and RAS1 during a Ram access.

4. PAL 33 -

This PAL is configured as an Address Decoder. Here LADR 17:23 are decoded into control lines such as RAS Enable 0, RAS Enable 1, CAS Enable, PROM select and XMA8. Buffered Address Latch Enable (BALE) will clock these lines (except XMA8) through DLatach U69 (74ALS573).

5. PARITY CHECK -

Made up of two Parity Generator/Checkers (74F280) and four Parity Rams this block will test the Ram bank for an uneven parity. If an error is detected then the system will compute the location and display it on the screen.

6. MUX -

This block will multiplex system address bits 1:7 and 9:15 in order to do high and low order ram access.

7. PAL 55 -

This PAL has three functions outlined below:

1. RAM PARITY -
During an active memory read cycle the ODD and EVEN Parity bits input from Ram are enabled to the Ram Parity Generator/Checkers.

2. CAS UPPER/CAS LOWER -
System Address bit 0 (SYSA0) will enable the output of CAS0. However Buffer High Enable (NBHE) will enable the output of CAS1.

3. XMA7 -
Multiplexed Address bit 7 is the product of ether System Address bit 8 or 16 if NBHE is low. When NREFRESH is low then XMA7 is outputing a valid Ram Refresh address. However when NREFERESH is high XMA7 is a valid Ram address.

8. RAM -

The 1meg ram bank is made up of eight 256K X 4 rams. They are then split into 4 banks, 1,2,3, and 4. Banks 1 and 2 handle the lower 512K. Bank 3 and 4 are split into four 128K blocks. The first 128K is the expanded ram (640K) mapping from 000000 to 09FFFF. Address bit A20 is Low. The upper 384K is accessed when address bit A20 is High. Decoded address bits A17 and A18 are now used to access the upper 384K in banks 3 and 4.

9. 8242 KEYBOARD CONTROLLER -

This programmable micro-controller will handle the keyboard scan codes. Even though the 2286 AT emulator uses the Amiga keyboard this chip will process incoming as if it were generated by a PC keyboard.

The 8242 will also generate A20GT which is used to gate address bit 20 when accessing 1MEG and above.

10. 6818 REAL TIME CLOCK -

This chip will maintain the time of day as well as the current date.

16. BATTERY -

This rechargeable back up battery will maintain power to the Real Time Clock and to the CMOS memory, where the setup information is stored.

AMIGA PARTITIONS
(1st time set-up with 2090-A card)

Here's a sample entry for a 20MB ST506 harddrive divided into three partitions. Load 1.3 Workbench disk and then insert the 2090-A install disk into drive. Double click on HDinstall icon. When window opens double click on install icon. After install copies the files onto your Workbench, it will automatically go into Prep. Prep is the program the Amiga uses to initialize the first two cylinders of your hard disk (cylinders 0 and 1). These two cylinders are reserved for Amiga software use. The rest of the drive will be available for data storage. The third question during the prep will ask you to enter the last cylinder to be used for the first partition. The first partition only needs to contain the boot-up sequence, therefore does not have to be to large in size. The last cylinder should be set at 40 (that will make the partition about 1.3 MB). When prep is finished, reboot the Amiga with Workbench disk and open up window and go into the shell icon**(see note on page 3) Now you must format the first partition. The syntax you type is as follows:

FORMAT DRIVE DH0: NAME "DISKNAME"

"DISKNAME" is any name you want to call the new partition. Next you have to copy your Workbench over to your hard disk. The syntax to do so is as follows:

COPY DF0: TO DH0: ALL

This is assuming Workbench is in drive df0:. Once the disk is finished coping remove the disk and reboot the system. You do not need the Workbench disk to boot-up any longer.

To make the second partition you must modify your mountlist file. You can divide the hard disk into as many partitions as needed. This is a sample of just two partitions (not including the 1st partition, which has the Workbench on it). This partition will use cylinders 41 through 305. To get into the mountlist go into the shell and type the following:

ED DH0:DEVS/MOUNTLIST

Once in the mountlist, go down the file until you get to an example mountlist called "FAST". Make a copy of this mountlist by doing the following:

1. **Put the cursor on the letter "F" in the word fast.**
2. **Push the Esc key, then type the letters "BS" (Block Start) and hit return.**
3. **Then put the cursor on the # sign at the end of the mountlist and push the Esc key, then type "BE" (Block end) and hit return.**
4. **Leaving the cursor on the # sign, push the Esc key once again and type "IB" (insert block) and hit return. (DO THIS TWICE)**

AMIGA PARTITIONS
(1st time set-up with 2090-A card)

There is now three copies of the "FAST" mountlist. Using any one of the mountlists, change the name from FAST to FH0:. Now go down and change the low cylinder to 41 and the high cylinder to 305. Your mountlist should look like this:

```
FH0: Device = hddisk.device
     FileSystem = 1:FastFileSystem
     Unit = 1
     Flags = 0
     Surfaces = 4
     BlocksPerTrack = 17
     Reserved = 2
     Interleave = 0
     LowCyl = 41; HighCyl = 305
     Buffers = 20
     GlobeVec = -1
     BufMemType = 0
     Mount = 1
     DosType = 0x444f5301
     StackSize = 4000
```

Go to one of the other Mountlists called Fast: and change the name to fh1:. This is for the third partition. Be sure the LowCyl equals 306 and the HighCyl equals 611.

The surfaces should be changed to the amount of heads that are on the hard drive. Now push the Esc key and type the letter "X" and hit return (Esc X saves the new mountlist to disk). You must now mount the new partitions. To mount go into the startup-sequence by typing the following in the shell:

```
ED DH0:S/STARTUP-SEQUENCE
```

Once in the startup-sequence go down the list until you get to BindDrivers. Put the cursor at the end of the word and hit return. In the blank space type the following:

```
MOUNT FH0:  <RETURN>
MOUNT FH1:
```

Push the Esc key and type the letter "X" and hit return. Reboot the system. Now you must format the partitions separately by typing the following:

```
FORMAT DRIVE FH0: NAME "ANYNAME" FFS
FORMAT DRIVE FH1: NAME "ANYNAME" FFS
```

FFS stands for Fast File System. Fast File System can be used on all partitions except the first partition. "anyname" is the name you want to call the partition.

A500 / A2000 SYSTEM TEST

The system test may also be customized. Type the word SYSTEST then select any of the numeric alpha characters listed below.

1 _ LOOP - Enable endless loop. This will test the real time clock and the blit only once. Then cycle through any of the selected test.

2 _ BLIT - This will run BOXER as a background task for loading.

3 _ ETERNAL (FAST) RAM - This will test ram at $C00,000 and $200,000 to $9FF,FFF if found. If Fast ram is not detected program will pass but say FAST RAM NOT FOUND.

4 _ EXTERNAL (CHIP) RAM - This is the same as FAST RAM test except that it test from $0 to $800,000.

5 _ EXTENDED RAM TEST - This is an expanded version of Fast Ram. Along with unique address and refresh various bit patterns are used. These are All Zeros, All Ones, All A's and All 5's.

6 _ DISK - In order to test the disk I/O. Tracks 2 and 8 will first be formatted. upon successes $6DB6 will be written and verified from tracks 2 and 8.

7 _ EXTENDED DISK - For Extended Disk all ONES, ALL ZEROS, $AAAA and $5555 are used.

8 _ EXTRA DISK SEEK - During PCB and Burnin testing this will seek only at a minimum. However during Aging test it will use maximum seeking.

9 _ DISK CYCLE - This will test disk every 8 cycles if Loop test is also selected.

a _ NO AUDIO - Disable Audio test.

b _ NO SPRITES - Disable Sprite test.

c _ CLOCK CHIP - This will test the real time clock chip.

d _ DISPLAY GRAPHICS CLOCK - Displaying the graphics clock will test bliter indicating a bad agnus chip if display is corrupt.

f _ BLITER LINE DRAW - This will test for extraneous bit set. Previously visual inspection of the graphics clock was used to test bliter. However now the line draw feature of agnus is tested through the program.

AMIGA PARTITIONS
(1st time set-up with 2090-A card)

** Workbench 1.3 uses "SHELL" instead of "CLI" (for more information on SHELL look in your 1.3 Enhancer Software Manual). If you are getting errors when you try to go into the shell, just click on cancel and it will default to the cli window. Put your Workbench disk in df0: and type the following:

```
COPY DF0:S/STARTUP-SEQUENCE TO DH0:S
```

This will correct the errors so you can get into the shell.

SUBDIVIDING THE WORK PARTITION FOR THE A2000HD & A2500

1. Modify the Work partition's Mountlist file (located in the Boot partition's devs directory). To do this go into shell and type the following:

   ```
   ED BOOT:DEVS/MOUNTLIST.HD
   ```

   A new window appears with the MountList.hd file. Cursor down until you reach the FH1: entry. change the HighCyl to 471. The low cylinder does not need to be changed. This cuts the partition in half.

2. Create a new MountList Entry for the next partition. Move down to a line below the # sign of the FH1: entry. Type the following information as shown below:

   ```
   FH2:    Device = hddisk.device
           Unit = 1
           Flags = 0
           Surfaces = 6
           BlocksPerTrack = 17
           FileSystem = 1:FastFileSystem
           Reserved = 2
           LowCyl = 472 ; HighCyl = 869
           Buffers = 50
           GlobVec = -1
           Mount = 1
           BufMemType = 0
           DosType = 0x444F5301
   ```

3. Exit ED by hitting the ESC key then the letter X. This saves the mountlist to disk.

4. Now you must mount fh2: in the startup-sequence. To do that type the following:

   ```
   ED BOOT:S/STARTUP-SEQUENCE
   ```

   Go down to the line that mounts fh1: and put the cursor at the end of the line. Hit return. In the empty space type the following:

   ```
   MOUNT FH2: FROM DEVS/MOUNTLIST.HD
   ```

   Save the changes to the file, by hitting the ESC key and typing the letter X.

5. Reboot the computer.

SUBDIVIDING THE WORK PARTITION FOR THE A2000HD & A2500

6. Open SHELL window, and format the new partitions by typing the following:

```
FORMAT DRIVE FH1: NAME "ANY NAME" QUICK
FORMAT DRIVE FH2: NAME "ANY NAME" QUICK
```

   Since the drive has been previously formatted, you can specify the QUICK option after the format commands. This speeds up the formatting process considerably. The name of the partitions can be any name you like them to be.

   NOTE: After subdividing the Work partition, be careful when running FormatHD or FormatWork. If you use these programs, they will reformat your Work partition as one large partition (as the drive was shipped).

SETTING UP AUTOBOOT PSEUDO HARDDRIVE

If your AutoBoot file will reside on an ffs (Fast File System) partition, you may need to add the following line to the FFS partition's mountlist entry:

MASK = 0

To do this type the following:

ED DH0:DEVS/MOUNTLIST

Go down to the mountlist that you are using and insert the "MASK" towards the bottom of the list. Since this will slow down all file access to the partition it is recommended that you create a separate partition for the AutoBoot file.
(See other write-up on Amiga partitions)

You set up an AutoBoot volume as follows:

1. Edit your startup-sequence to include the following line after binddrivers:

   RUN >NIL: SYS:PC/PCDISK

   AutoBoot uses PCDisk to access the Amiga file; therefore, PCDisk must be running when the Bridgeboard PC begins its boot procedure.

   To get into startup-sequence type the following in shell:

   ED DH0:S/STARTUP-SEQUENCE

   To exit startup-sequence hit the ESC key and the letter X. This saves the new startup-sequence.

2. To create the AutoBoot file on the Amiga, use the makeab command included with the PCInstall Disk. With the PCInstall disk in DF0:, change drives by using (cd DF0:). Decide where you want the file to be located and issue the command:

   MAKEAB DRIVE:DIRECTORY/FILENAME

   For example, to set up a PC pseudo hard drive on an Amiga hard drive you would type the following command:

   MAKEAB FH2:FAKEC

   Where FH1: is the partition you are using, (this is just an example, you can use any partition you like), FakeC is the name of the file. Makeab will now ask for the number of heads, sectors/track, and number of cylinders to emulate. Makeab will now print the size of the proposed hard disk file and ask you to accept it.

Make sure the "total file size" is no larger then the partition you are putting it on. If it is larger, answer no (when asked to go on) and redo makeab command and lower the cylinder size. When parameters are correct, answer yes to go on and makeab will generate the file in the proper format for AutoBoot. Makeab creates the file "full-sized" so it may take some time to create large volumes.

3. Once the file is created, you must tell the system where to find it. The system looks for the file ABOOT.CTRL in the SYS.PC/System directory. This file must contain the full path-filename of the file you created with makeab. For the example given above you create the file by typing the following:

   ED SYS:PC/SYSTEM/ABOOT.CTRL

   The screen will display a "creating new file". In the file type the path-filename plus a carriage return by typing the following:

   FH1:FAKEC

   This is actually the name of the file just created. Save this file by hitting the ESC key and the letter X. This step completes work on the Amiga side.

4. Reboot. Boot-up the PC window with the MS-DOS System disk. Then you must execute the FDISK and FORMAT commands. To do so type the following:

   FDISK

   Choose 1 (create DOS partition)
   Then choose 1 again to create primary DOS partition.
   Finally choose "y" to do maximum size and make active.

   The system will then reboot. Next format the drive by typing the following:

   FORMAT C:/S

   Format will be very quick. After the hard drive is formatted, remove the floppy MS-DOS disk and reboot the PC (Ctrl Alt Del keys). The PC will boot from the AutoBoot volume. You can copy files on to it and use it just as you would a normal PC hard drive.

INSTALLING VIRTUAL DRIVES ON AMIGA HARDDRIVES

Make sure there is a JDISK.SYS and JLINK.COM on your MSDOS disk. Put a copy of MSDOS disk in drive A and reboot whole system. Open PC drawer and double click on PCdisk icon. Now open PC window. Add the following to your Config.sys file:

Device=Jdisk.sys

(For more information on edlin, refer to your MS-DOS users guide)
To get into edlin, type the following at your A prompt:

edlin config.sys   <return>

At the asterisk type the letter "i" (insert line) and hit return. on the first line type:

Device=Jdisk.sys   <return>

On the next line hold the CTRL key down and hit the letter Z (Z gets you off the line so you can exit) then hit return. At the asterisk type E (ends editing & saves to disk) and hit return. Reboot by pressing CTRL/ALT/DEL keys simultaneously.

To link a virtual drive to an AMiga file, use the Jlink command. the syntax for that command is as follows:

Jlink n: filename /sw

Where n: is the virtual drive to use, filename is a standard AMiga path specifying a file, and /sw is one of the following switches:

/n	**- opens an already created file.**
/c:n	**- creates that volume on AMiga side - n is the size in kilobytes.**
/u	**- unlinks a file already opened.**
/r	**- links read only (all write accesses will fail)**

Create an autoexec.bat file so the file you created boots up every time you go into the PC environment.

Type the following:

```
A> edlin autoexec.bat   <return>
{new file}
*i   <return>
     1:*jlink C: dh0:filename /n    <return>
     2:*CTRL Z                      <return>
*e   <return>
```

REBOOT THE SYSTEM.

INSTALLING VIRTUAL DRIVES ON AMIGA HARDDRIVES

To create a file type the following:

jlink C: dh0:vd /c:1000

```
Where:     C: = virtual drive to be used.
         dh0: = harddrive being used.
          vd  = any file name you want to give it.
          /c: = create that volume on AMiga side.
        1000  = the size in kilobytes, in this case 1000
                kilobytes or 1 megabyte.
```

Any time you want to use the virtual drive you must always open PCdisk before going to the PC environment, and must close the link if you are going to use AREAD or AWRITE.

Here's an example on how to use Jlink:
This is assuming you have added Jdisk to the config.sys file.

```
A> jlink C: dh0:vd /c:1000    <return> (creates file)
A> copy command.com d:        <return> (this is copying the
                                        file, command.com
                                        from drive A to
                                        virtual drive d)
A> C:                         <return> (changes drive
                                        directories)
D> dir                        <return> (this shows that the
                                        file has been copied)
```

The command.com file should be in the virtual drive file. This is the first virtual drive file. You can do the same with virtual drives d,e,and f.

To unlink, go back to drive A and type:

jlink C: dh0:vd /u

Where /u = unlink.

AMIGA PARTITION ON PC HARDDRIVE

Before starting be sure you have DEBUG, FDISK AND ADISK on MSdos disk. With backup copy of workbench, boot up the system and do a PC install with the 2088 disk. When the install program gets to the, "files to install" section, go down to the line with the startup-sequence on it and click on the check mark (We do not want to copy this over to the work-bench disk). Then click on the "ok" box to continue. When finished the install program, reboot the system with the PC install workbench.

Then go into PC window (make sure you have your copy of MSdos disk in the 5 1/4" drive) and type debug at the "A" prompt. Type the following syntax (at the dash) to do a low level format:

g=c800:5 <return>

Answer the questions according to the type of harddrive and hardcard you have, leaving the interleave set at 3. When formatting is finished, reboot the PC side by holding the CTRL, ALT and DEL keys down. When the "A" prompt returns type the following to set up the partition:

fdisk <return>

Select option 1 (create dos partition). Answer "no" on question if you want to do whole disk. Then select the number of cylinders you want to make the size of your partition. Then enter starting cylinder at 0 and hit return. Press the Esc key to go back to the fdisk options menu and select option 2 (change active partition). Select number 1 as your active partition and hit return. Hit the Esc key to go back to the menu. Reboot the PC side (CTRL, ALT and DEL) and format the partition using the following syntax:

format c:/s

When the "A" prompt returns type the following to start your AMiga partition:

adisk <return>

Select option 3 (create an AMiga partition). Enter the size of the partition you want to make and hit return. Then enter the starting cylinder (which should be one after the last cylinder on your PC partition) and hit return. Reboot whole system (CTRL, AMIGA, AMIGA keys). Open workbench and go into shell (or cli if you have 1.2 workbench) to mount and format AMiga partition. The syntax for mounting is as follows:

djmount <return>

If errors occur, just cancel them all until you get a prompt (If you don't get a prompt, the harddrive may not be compatible). At the prompt type the following to format the AMiga partition:

```
DPFORMAT DRIVE JH0: NAME "ANYNAME"    <return>
```

There is no fast file system for the AMiga partition (on PC harddrive).

Next you must edit your startup-sequence so you can access the new partition on boot-up. The syntax to edit your start-sequence is as follows:

```
ED DF0:S/STARTUP-SEQUENCE    <return>
```

Once you are in the startup-sequence, go down to the line that reads; cd c: and put a space below it by going to the end of the line and hit return. In the space type - BindDrivers <return>. On the next line type - wait 30 <return> (wait 30 allows BindDrivers to complete its routines). In the last space type - djmount (this mounts the drive every time you boot-up). before saving, go down the list and find the BindDrivers that was originally in the startup-sequence. Put the cursor on the same line. Hold down the CTRL key and hit the B key. This will delete the line. The first six lines should look like the following:

```
Addbuffers df0: 10
c:setpatch >NIL: patch system functions
cd c:
BindDrivers
Wait 30
djmount
```

Next press the Esc key then the X key and hit return (this saves the edits to disks). Reboot the system and use the partition like a normal Amiga Partition.

SECTION FOUR TROUBLESHOOTING

A500 / A2000 SYSTEM TEST

System Test is an auto-boot diskette. Once booted its own cli window will open. Type the commands listed below in order to run each test.

KEYTEST
- Full keyboard test
- press all keys to confirm proper operation
- click left mouse button in upper left corner to EXIT

KB
- Keyboard matrix test
- pressing the keys indicated by the program will
- test the matrix and seven dedicated key functions

SCREENTEST
- Screentest is low resolution graphic screen made up of 6 bit planes. Used to test RGB linearity and HALF BRIGHTS.
- screentest will display a RGB COLOR BAR scale and a 16 level gray scale in both normal and half bright mode. Normal scales are on top.
- Press space bar to EXIT screen.

SHOW.HIRES.TEST
- Hires.Test is a 4 bit plane high resolution 640 by 400 screen that displays NTSC color bars, 8 level gray scales, horizontal resolution lines, and an interlace test.
- Press space bar to EXIT screen.

SHOW BALLOON
- Balloon is a 6 bit plane hold and modify picture used to verify proper HAM decoding.
- This will test for a bad Agnus or Denise chip.
- Press space bar to EXIT screen.

CUBEROTE
- Cuberote is an animated cube whos speed and direction is controlled by the mouse X-Y position. The cube is blue with shading of three visible sides. Cuberote uses double buffering for a smooth animation. Direct manipulation of the Copper List permits the display of 16 shades blue while only using 2 bit planes.
- Click left mouse button in upper left corner to EXIT screen.

SYSTEST
- Systest will run the following test Realtime clock, Blit, Chip/Fast Ram Sprites/Bliter, and Disk I/O.
- Reboot computer to EXIT program.

Type the following for customizing systest.

SYSTEST 246d	; A500 NO A501	PCB
SYSTEST 2436cd	; A500 WITH A501 OR A2000	PCB
SYSTEST 1245679d	; A500 NO A501	BURN-IN
SYSTEST 12345679cd	; A500 WITH A501 OR A2000	BURN-IN
SYSTEST 12345678d	; A500 NO A501	AGING
SYSTEST 12345678cd	; A500 WITH A501 OR A2000	AGING

AMIGA 1.3 STARTUP SEQUENCE - OVERVIEW

---

1 - DELAY 1/3 SECOND (LET HARDWARE SETTLE)

2 - JUMP TO ROM CODE IN DIAG CART (IF FOUND)

3 - DISABLE/CLEAR ALL INTERUPTS & DMA'S

4 - TURN ON SCREEN

5 - DISPLAY DARK GRAY SCREEN -HARDWARE OK

6 - CHECKSUM ROM - IF BAD DISPLAY RED SCREEN

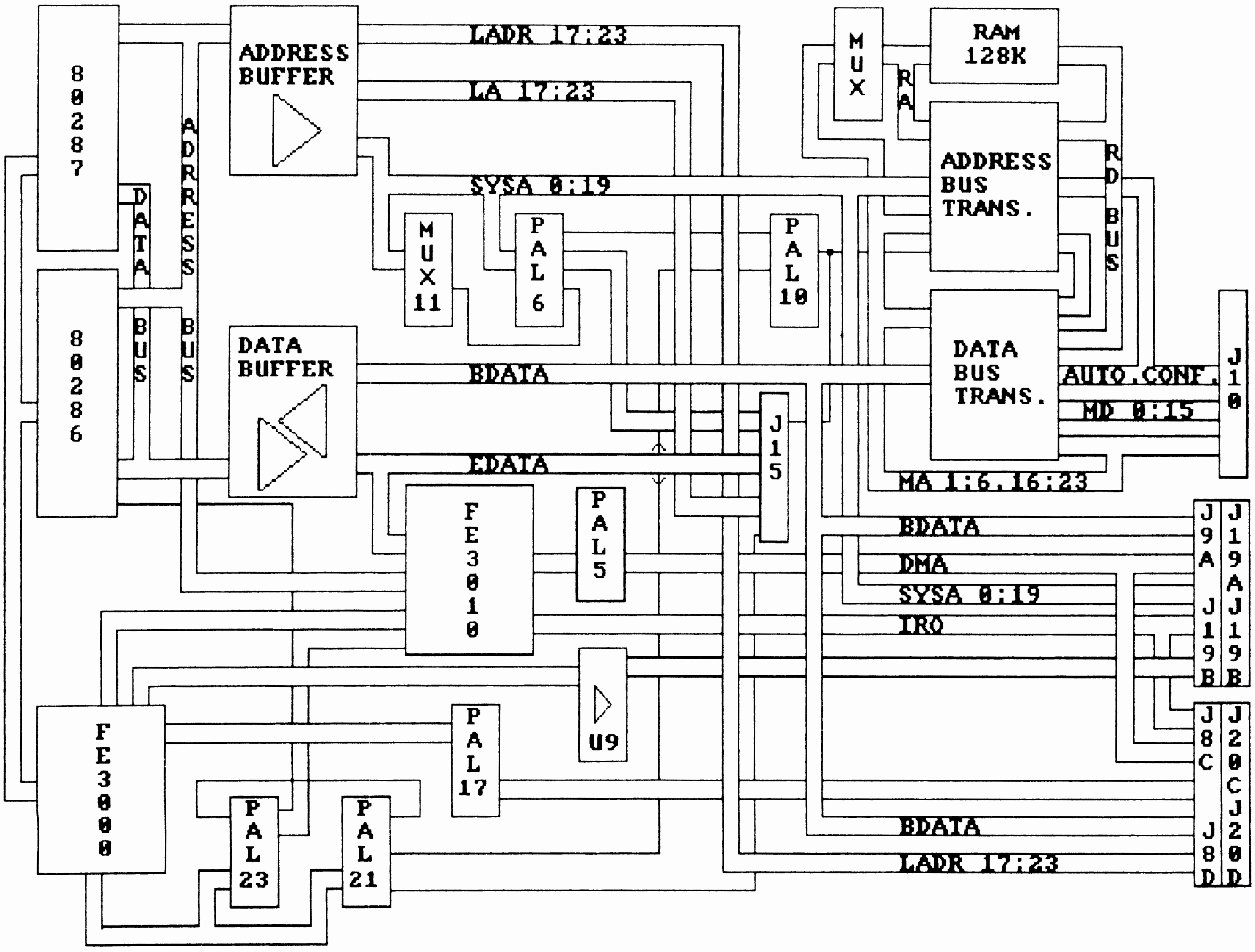
7 - SET UP TEMPORARY EXECPTION PROCESSING - IF SPURIOUS
EXECPTIONS OCCUR, DISPLAY YELLOW SCREEN

8 - CONFIGURE LOCAL MEMORY (CHECK BOUNDARIES, SIZE)
IF PROBLEM, DISPLAY GREEN SCREEN

9 - CHECK SOME CUSTOM IC REGISTERS - IF FAILURE,
DISPLAY BLUE SCREEN

10 - RESTORE SCREEN --- CHANGE TO LIGHT GRAY = SOFTWARE OK

A2286 PC/AT EMULATOR



SECTION FOUR TROUBLESHOOTING

A500 / A2000 SYSTEM TEST

System Test is an auto-boot diskette. Once booted its own cli window will open. Type the commands listed below in order to run each test.

KEYTEST	- Full keyboard test - press all keys to confirm proper operation - click left mouse button in upper left corner to EXIT
KB	- Keyboard matrix test - pressing the keys indicated by the program will - test the matrix and seven dedicated key functions
SCREENTEST	- Screentest is low resolution graphic screen made up of 6 bit planes. Used to test RGB linearity and HALF BRIGHTS. - screentest will display a RGB COLOR BAR scale and a 16 level gray scale in both normal and half bright mode. Normal scales are on top. - Press space bar to EXIT screen.
SHOW.HIRES.TEST	- Hires.Test is a 4 bit plane high resolution 640 by 400 screen that displays NTSC color bars, 8 level gray scales, horizontal resolution lines, and an interlace test. - Press space bar to EXIT screen.
SHOW BALLOON	- Balloon is a 6 bit plane hold and modify picture used to verify proper HAM decoding. - This will test for a bad Agnus or Denise chip. - Press space bar to EXIT screen.
CUBEROTE	- Cuberote is an animated cube whos speed and direction is controlled by the mouse X-Y position. The cube is blue with shading of three visible sides. Cuberote uses double buffering for a smooth animation. Direct manipulation of the Copper List permits the display of 16 shades blue while only using 2 bit planes. - Click left mouse button in upper left corner to EXIT screen.
SYSTEST	- Systest will run the following test Realtime clock, Blit, Chip/Fast Ram Sprites/Bliter, and Disk I/O. - Reboot computer to EXIT program.

Type the following for customizing systest.

```
SYSTEST 246d           ; A500 NO A501             PCB
SYSTEST 2436cd         ; A500 WITH A501 OR A2000  PCB
SYSTEST 1245679d       ; A500 NO A501             BURN-IN
SYSTEST 12345679cd     ; A500 WITH A501 OR A2000  BURN-IN
SYSTEST 12345678d      ; A500 NO A501             AGING
SYSTEST 12345678cd     ; A500 WITH A501 OR A2000  AGING
```

AMIGA 1.3 STARTUP SEQUENCE - OVERVIEW

---

1 - DELAY 1/3 SECOND (LET HARDWARE SETTLE)

2 - JUMP TO ROM CODE IN DIAG CART (IF FOUND)

3 - DISABLE/CLEAR ALL INTERUPTS & DMA'S

4 - TURN ON SCREEN

5 - DISPLAY DARK GRAY SCREEN -HARDWARE OK

6 - CHECKSUM ROM - IF BAD DISPLAY RED SCREEN

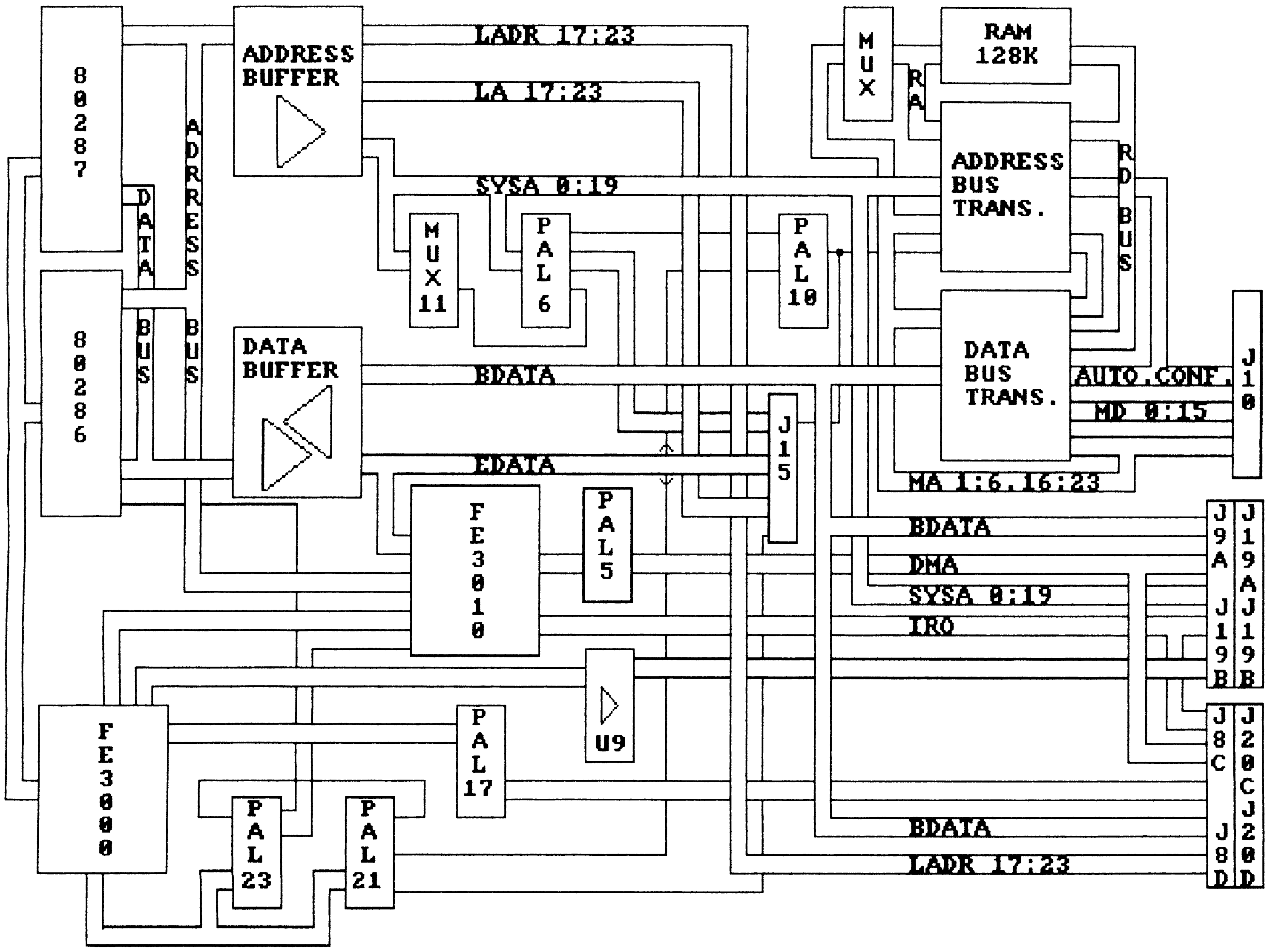
7 - SET UP TEMPORARY EXECPTION PROCESSING - IF SPURIOUS EXECPTIONS OCCUR, DISPLAY YELLOW SCREEN

8 - CONFIGURE LOCAL MEMORY (CHECK BOUNDARIES, SIZE) IF PROBLEM, DISPLAY GREEN SCREEN

9 - CHECK SOME CUSTOM IC REGISTERS - IF FAILURE, DISPLAY BLUE SCREEN

10 - RESTORE SCREEN --- CHANGE TO LIGHT GRAY = SOFTWARE OK

A2286 PC/AT EMULATOR


80287
80286
DATA
ADDRESS
BUS
BUS
ADDRESS BUFFER
DATA BUFFER
LADR 17:23
LA 17:23
SYSA 0:19
BDATA
EDATA
MUX 11
PAL 6
PAL 10
MUX
RA
RAM 128K
ADDRESS BUS TRANS.
RD BUS
DATA BUS TRANS.
AUTO.CONF.
MD 0:15
J10
J15
MA 1:6,16:23
FE3010
PAL 5
BDATA
DMA
SYSA 0:19
IRQ
J9A J19B
J19A J19B
U9
FE3000
PAL 17
PAL 23
PAL 21
J8C J8D
J20C J20D
BDATA
LADR 17:23