

COMPUTER-VIDEO INTERFACE SYSTEM

by

Joan Marie Sulecki

A.B. Bryn Mawr College  
(1981)

Submitted to the Department of  
Electrical Engineering and Computer Science  
in Partial Fulfillment of the  
Requirements of the Degree of


MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

at the


MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June, 1983


Copyright Massachusetts Institute of Technology, 1983

Signature of Author, 

Department of Electrical Engineering  
and Computer Science, May 13, 1983

Certified by 

Prof. David H. Staelin  
Thesis Supervisor

Accepted by 

Arthur C. Smith  
Chairman, Departmental Committee  
on Graduate Students

**Archives**

MASSACHUSETTS INSTITUTE  
OF TECHNOLOGY

SEP 1 1983

LIBRARIES

# COMPUTER-VIDEO INTERFACE SYSTEM

by

Joan Marie Sulecki

Submitted to the Department of  
Electrical Engineering and Computer Science  
on May 13, 1983,  
in partial fulfillment of  
the requirements for the Degree of  
Master of Science in Electrical Engineering

## ABSTRACT

An interface for a Data General computer has been constructed to serve as a tool for experiments in video bandwidth compression. The system comprises the Nova/4 computer with its 130 MB disk and its Direct Memory Access unit, two 64KB Digital Equipment Corporation frame store memories, a monochrome video camera and video monitor, and a substantial amount of control logic.

The system operates in both acquisition and display modes. In the first mode, data flow from the camera, through a frame memory and the DMA, to the disk. The display mode routes data from the disk, through the DMA and frame memory, to the video monitor. Several parameters which are controlled by software allow the camera data to be sampled at either 8 or 4 bits per pel, and with varying horizontal and vertical resolution. The image can likewise be displayed on a full, half, or quarter screen. The coarsest possible quantization is 60 x 128 pels at 4 bits; the most fine is 240 x 128 pels at 8 bits. The images can be processed at effective frame rates ranging from 60 to 10 frames per second. This versatility will be most useful in observing the effect of quantization and frame rate on image quality.

This thesis gives an overview of the total system operation, and treats the interface between the computer and interpolation or sampling circuits in detail.

Significant features of the data path and control section are the use of two alternating frame store memories, the buffering of those memories with FIFOs, and the generation of all the memory driving signals by custom logic. A buffer between the disk and video elements is definitely necessary, and two frame memories are used so that one may be loaded while the other is unloaded. The FIFOs allow the memories to work independently from the DMA, camera, and monitor, so synchronization problems are eliminated. Since the memories require different control signals from those which the Nova provide, all the DEC memory signals must be generated with delay lines and logic.

The video interface system will be a versatile and useful tool.

Thesis Supervisor: Dr. David H. Staelin

Title: Professor of Electrical Engineering

## Contents

Acknowledgements	2
Chapters	
1. Introduction	3
2. Control and Data Path Overview	12
3. The Direct Memory Access Interface	21
4. Memory Control and Buffering	30
5. Practicalities, Performance, and Maintenance	40
6. Summary	52
Appendices	
A. Control Software	55
B. DMA Circuit Diagrams	64
C. DMA and System Logic Circuit Diagrams	69
D. Control and Buffer Circuit Diagrams	74
E. Interconnections	79
F. Memory Timing Diagrams	85
References	89

## Acknowledgements

I wish to express my gratitude to Prof. David H. Staelin for his constant encouragement, to Dr. Philip W. Rosenkranz for his assistance with design and testing, and to Mr. John W. Barrett for his guidance throughout the construction of the system.

Thanks to my collaborators, Biswa R. Ghosh and Henrique S. Malvar, who designed, constructed, and debugged the interpolation and sync generation section, and the analog/digital conversion section of the interface. Thanks to Henrique Malvar also for sharing his insight and ability during the system testing.

Finally, many thanks to Mark W. Merritt for patient teaching and clear advice throughout the design phase of the project.

## Chapter 1 Introduction

Communication by video-conferencing and by "picture phones" has been awaited for many years. The primary technological constraint which must be overcome to create these services is the large bandwidth of video signals. A standard telephone line normally transmits information at a maximum rate of 9.6 kilobits per second (kbps); conventional television is transmitted at about 60 megabits per second (Mbps). The problem is therefore to reduce the amount of information transmitted without drastically reducing the quality of the image. Standard video signals which have been digitally sampled are typically described with 8 bits per pel to yield 256 grey levels. (A pel is a Picture Element. A typical television frame is 512 pels wide by 483 pels long.) Compression schemes which are currently popular can reproduce full motion video images with an average of 18 bit per pel. The best known compression methods are transform, two-band, and linear predictive coding [1] [2].

Numerous facilities exist for research with still images, but few tools are available for the less common research on continuous-tone, full-motion images. Any interframe compression scheme must be tested on actual consecutive frames of a video transmission. An electronic system which digitizes the analog data from a video camera, stores the values on disk, and reconstructs the analog images for a monitor is an essential research tool. Such a system has been built to support a variety of bandwidth compression experiments. Its design, construction,

and performance form the basis of this thesis.

The hardware system operates in two modes, acquisition and display. None of the compression is done in real time. Either the camera data are read and stored, or the disk data are retrieved and displayed on the monitor. The system does not read a frame, process it, and then replay it while the next frame is being read. A diagram of the system components and data paths follows.

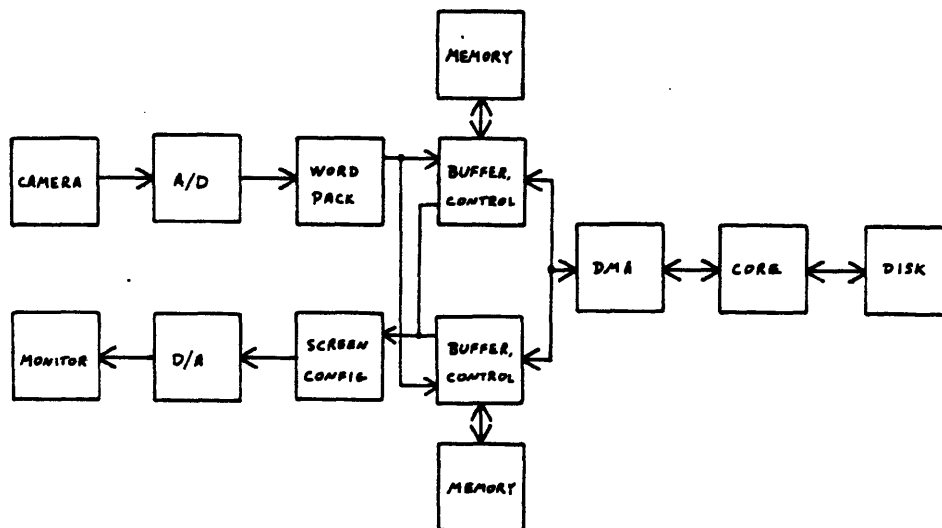


Figure 1-1

In acquisition (or "camera") mode, analog data from the camera are digitized and packed into 16 bit words. The stream of data is then passed through a FIFO (First In First Out) buffer and into a frame memory. The next camera frame is sent through a similar buffer into a second frame memory, while the first frame is directed out of its memory, through the buffer, through a DMA (Direct Memory Access interface), through the computer's core memory, and onto a disk. As even the description suggests, the transfer from frame memory to disk is

much slower than the 60 Hz camera frame acquisition rate. Therefore, after frame 1 is read, the next several camera frames are ignored. Then while frame 1 is sent to disk, the next desired frame is sent to the second frame memory. The number of frames which are skipped depends on several parameters discussed below. Operation continues with frame 2 sent through the path to the disk while the next desired frame travels from the camera to the first frame memory.

In display (or "monitor") mode all the data are originally lodged on the disk. The first frame is directed from the disk, through the core memory, the DMA, a FIFO buffer, and into the first frame memory. Next the second frame follows the same path to the other frame memory while the first frame is prepared for analog conversion by a circuit which adjusts such parameters as frame size and line averaging. From there it is converted to an analog signal, video sync signals are added, and the frame is displayed on the monitor. Once more the path between the disk and the frame memory is slower than the 60 Hz monitor repetition rate, so while frame 2 is loaded into its frame memory, frame 1 is displayed several times. "Several" is again determined by user adjustable parameters. Once all of frame 2 is in its memory, the next start of a monitor frame will select that new frame. Memory 1 is then ready to receive frame 3 from disk. This alternation continues until all frames have been displayed.

The design of the system falls logically into three parts. The analog/digital conversion section (the A/D and the D/A boxes of Figure 1-1) was designed by one student. It resides on two sides of one

printed circuit board. The MATV-0816 analog/digital converter samples a signal in the 0 to 1 volt range from an RCA video camera to create an 8 bit digital representation of the signal. The device can perform up to 16 million conversions per second, but in this application it will not be required to work at any greater rate than 2.5 MHz. The digital/analog conversion is done by an HDG-0805 converter. This chip is of ECL rather than TTL technology, so a TTL-ECL conversion buffer precedes the chip. The analog signal is also smoothed by a low pass Chebyshev filter before appearing on the CONRAC monitor.

The Word Packing and Screen Configuration boxes, designed by a second student, comprise three printed circuit boards and are actually an intricate line store, interpolator, and video sync generator. All of the composite sync and blanking signals needed by the camera and monitor are created here. The primary function of this section, though, is to shape the data into the particular format selected by the user. A 12 bit command register can be set from DIP switches for circuit debugging, but in normal operation it is set, once per experiment, by software. The bit assignments are in Table 1-1.



BIT	NAME	MEANING 0	MEANING 1
0	MO	monitor mode	camera mode
1	XRES	64 pels/line @ 8 bits 128 pels/line @ 4 bits	128 pels/line @ 8 bits 256 pels/line @ 4 bits
2	YRES0	00 - 60 lines/frame 01 - 120 lines/frame	
3	YRES1		
4	HAVG	no averaging	horizontal avg
5	VAVG	no averaging	vertical avg
6	PRES	8 bits/pel	4 bits/pel
7	SWID	1/2 screen width	full screen width
8	SLENO	00 - 1/4 screen length 01 - 1/2 screen length	
9	SLEN1		
10	TSM	tone scale memory on/off	-- not implemented
11	TSTP	data from frame memory	data from test PROM

Table 1-1

Bit 0 is part of the overall control state of the machine and is not important specifically to this section. In camera mode, the three relevant parameters are X-, Y-, and Pel RESolution. The X resolution selects an appropriate horizontal clock for the A/D sampling operation; the Y resolution similarly controls the vertical spacing of samples according to the number of lines required. If the pel resolution is 8 bits, then two samples will be packed into each word; if it is 4 bits, then the circuit will fill each word with four pels. A data frame can therefore be quantized in twelve different ways. The coarsest sampling and lowest quality image would be 60 lines x 128 pels at 4 bits per pel. The whole frame would be represented by 30 kilobits. The finest detail would require 240 kilobits (30 kilobytes) and would be the result of 240 lines x 128 pels x 8 bits or 240 lines x 256 pels x 4 bits. An image of reasonable quality and moderate bandwidth would be 120 lines x 128 pels x 8 bits, or 15 kilobytes (KB). All of these images represent the

camera's entire field of view; they differ in the number of samples which are taken as the field is scanned.

The large remaining part of this circuit is needed to present these variably sized frames on a variably sized monitor screen. The above resolutions are also important in the monitor mode for horizontal and vertical clocks and for unpacking the data words. In addition, a frame which originally filled the camera's full field of view may be displayed on the full screen or on only a fraction of it. The beam always scans the entire screen, but if half length and half width were selected, for example, data would only be displayed when the beam was in the top and left halves of the screen. The output would be blanked when the beam scanned the other three fourths of the screen.

The last two parameters that are implemented in this section are the vertical and horizontal averaging. The full monitor screen is 480 lines by 256 horizontal clock pulses, but since the display is interlaced, each frame is displayed as 240 lines by 256 pels. Given a full 240 line display screen and defining a "frame" to be the data which are displayed in 1/60 of a second (several consecutive "frames" will necessarily contain the same image because of the slow disk) the following description of averaging holds.

With no vertical averaging and 240 lines per frame, lines 1, 3, 5, . . . 479 are displayed, then the next frame fills lines 2, 4, 6, . . . 480. With 120 lines in the frame, the first frame line will appear as screen lines 1 and 3, the second frame line as screen lines 5 and 7, and onwards to frame line 120 appearing as screen lines 477 and

479. The next frame's lines will appear as screen lines 2 and 4, 6 and 8, . . . 478 and 480. Similarly, the lines of a 60 line frame will be repeated four times on the screen.

When vertical averaging is selected with a 240 line frame, the first frame is displayed as above, lines 1, 3, 5, . . . 479. The next frame sends the average of lines 1 and 3 to screen line 2, the average of 3 and 5 to line 4, continuing to the average of 477 and 479 for 478 and a straight repetition of 479 for 480. (Note that this configuration has displayed one whole frame of pure data and one whole frame of averages. This separation of actual and averaged frames will not hold true with smaller data frame sizes.) If there are 120 lines in the data frame and vertical averaging is selected, screen line 1 will be frame line 1, screen line 3 will be the average of frame lines 1 and 2, screen line 5 will be frame line 2, 7 will be the average of 2 and 3, et cetera. The next screen will be taken from the data frame so that screen line 2 is frame line 1, screen line 4 is frame lines 1 and 2 averaged, screen line 6 is frame line 2, and on to screen line 480. A 60 line data frame would fill screen lines 1, 3, 5, 7, 9, 11 as frame lines 1, 1, avg(1,2), avg(1,2), 2, 2, but since line repetition and averaging cannot be done simultaneously, this 60 line frame expanded to a full screen with averaging cannot occur. There are, of course, more and less reasonable choices of parameters. A frame which is 60 lines long would appear better on a quarter or half length screen so it would not require so much repetition. A 60 line frame on a half length screen would follow the rules described above for a 120 line frame on a 240

line screen and could be averaged.

Horizontal averaging is simpler both to describe and to implement. Without horizontal averaging, when a 128 pel line is spread over a 256 pulse screen each pel is "repeated" once, so pels appear as 1, 1, 2, 2, . . . . (Actually the horizontal scan is continuous, not discrete, so each of the 128 pels is simply held as data for twice as long as each of 256 pels would be. The action can be imagined as pel repetition though.) Four repetitions of each pel are necessary with a 64 pel line. If horizontal averaging is enabled at 128 pels per line, the screen width is filled by pel 1, avg(1,2), pel 2, avg(2,3), . . . . At 64 pels the screen line becomes 1, 1, avg(1,2), avg(1,2), 2, 2, avg(2,3), avg(2,3), . . . . Since horizontal averaging is not discrete repetition like vertical averaging, this mode is possible. Each datum, be it a straight value or an average, is simply displayed for twice as long as one of 128 pels would be.

The remaining part of the system, that is the frame memory buffers and control, and the DMA interface, was designed by the author. Several other parameters which are set by software have an effect on this section. The horizontal and vertical resolution are translated into a word count/frame size. The frame-repetition counter for the monitor (or frame-ignore counter for the camera) is the final important parameter. The disk can transfer a 120 x 128 pel x 8 bit frame, or 15 KB, at about 20 frames per second. Thus each disk frame must be repeated at least 3 times on the monitor. A software counter determines how many times each frame has been repeated, so some variation is possible. If the counter

were set to 3, an effective rate of 20 data frames per second can be achieved. If it were set to 4, only 15 new frames would appear each second. Similarly, in camera mode a variable number of frames can be skipped to allow experimentation with frame rate, flicker, and motion quality.

The goals of the project are ambitious. The variability of both input and output parameters and the size and speed of data frames have pushed the disk and the fastest integrated circuits to their limit. The timing and control of the data paths must be very accurate. They will bear detailed discussion in the following chapters.

## Chapter 2 Control and Data Path Overview

The video test system has made use of several large hardware components which the research group has owned for some time. The controlling computer is a Nova/4 by Data General Corporation. It has 76 KW of user core memory and a standard General Purpose Interface Board which is easily used as a DMA. The disk is a Pertec 130 MB formatted unit with a speed through the Nova Data Channel of approximately 320 KB per second. The frame memories were chosen for their size and speed. They are Digital Equipment Corporation MSV11-D,E 64 KB Random Access Memories with a cycle time of approximately 800 ns, or a data rate of 2.4 MB per second.

The disk data rate constrains the overall system data rate. The 320 KB/sec is equivalent to 20 frames of 120 x 128 pels x 8 bits in one second. The frame memories' 2.4 MB/sec is 8 times faster, so they could theoretically support that size frame at 160 frames/second (an unnecessarily high rate), or a 240 x 128 pel frame at 80 fps. The camera and monitor, with their A/D and D/A converters, can of course transfer the largest frames at 60 fps. In addition to the size of the frames and their transfer rate, their shape is also constrained. All 15 kilobytes of a frame do not flow steadily through the interface; they are read or written one line at a time and no data are transferred as the line retraces. So in addition to the capability to move a whole frame at a fast rate, the system must be able to move a single line within a short period of time. If the monitor or camera traces 240 lines per

16.6 millisecond frame, it begins a new line every 63.5 microseconds. Of that time, about 51.2 us are used for data transfer and 12.3 us for retrace. In 51,200 ns the frame memory can transfer a maximum of 64 words. So the memories limit the system to lines of 128 8-bit pels or 256 4-bit pels. Therefore the 120 x 128 pel x 8 bit image is indeed the optimum frame for this hardware. The 8 bit resolution is good, the 15K pel quantization is reasonable, the 20 fps data rate will not flicker, and the 128 pel lines are as wide as possible with the 8 bit resolution.

The data flow through the system is closely controlled by software. Each disk and DMA transfer must be initialized and triggered, and the frame repetition or ignoring must be monitored. Typical control programs for camera and monitor mode operation, CC11 and CM11, appear as Appendix A. Data frames on the disk are appended together to form one large CONTIGUOUS file. Within the program bounds two half-frame-size buffers are allocated. The frame memories' starting address is switch settable to either 000000 octal or 100000 octal; even the largest frame will use only half of the memory. The camera or monitor will always transmit or receive a full frame of data, be that 60 x 64 pels or 240 x 128 pels, but the disk and DMA will transfer only a half frame during each operation. With this configuration the core buffers need only total the size of one frame rather than two, and the transfer rate is not diminished.

The two figures which follow depict the path which data follow through the system. In Figure 2-1, the camera mode, the leftmost box represents the disk, the next pair is the two half-frame core buffers,

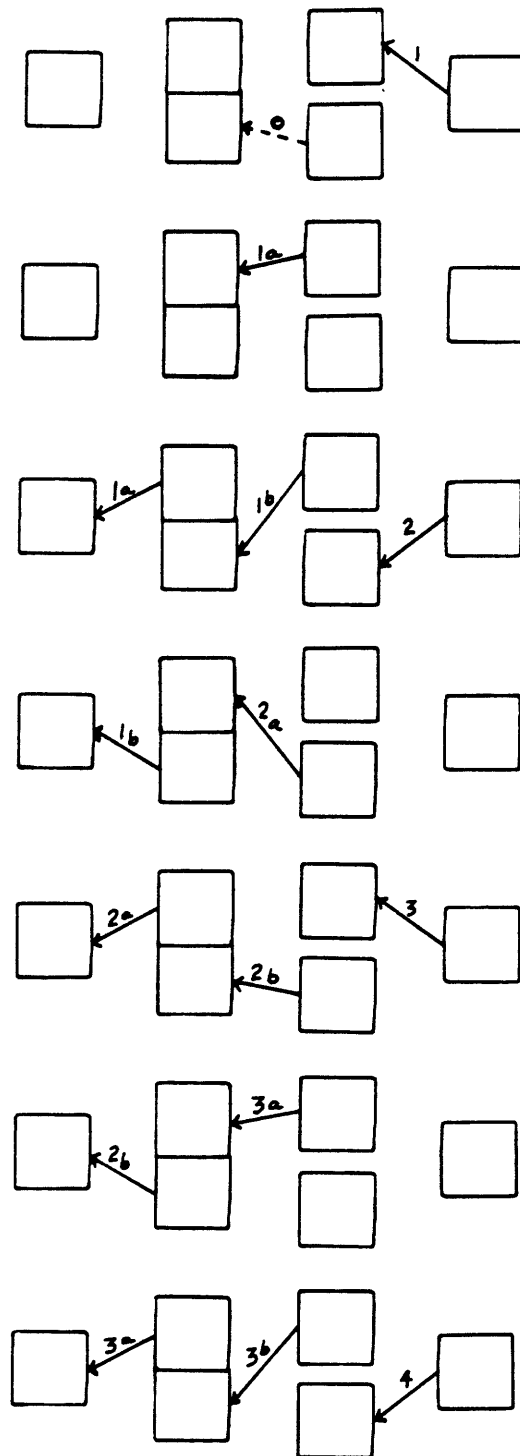


Figure 2-1



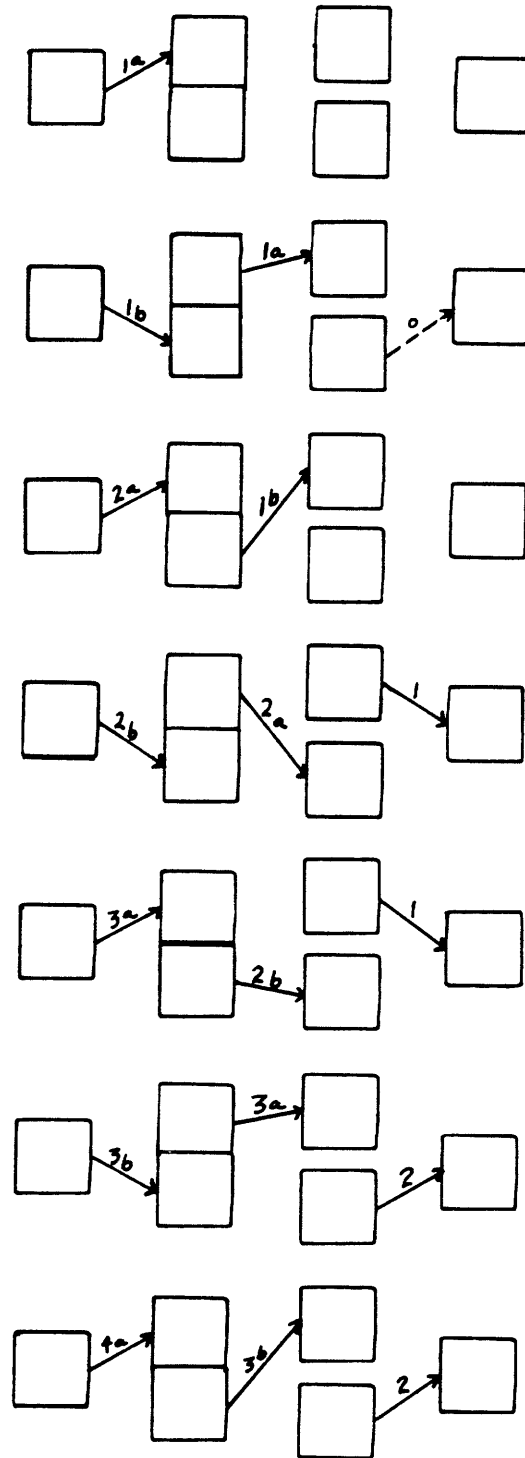


Figure 2-2

next are the frame memories, and the rightmost is the camera. Numbers 1, 2, 3, 4, are the full frames acquired by the camera; several frames will have been ignored between each of them. Labels 1a, 1b, etc. represent the first and second halves of frame 1 as they move between memory and disk. The first three segments of the scheme are a startup sequence. First the camera deposits frame 1 in frame memory 1. (The dotted line 0 appears because the memory-to-core operation must always occur when the camera to memory transfer is active. The spurious data which are labelled 0 will be ignored.) Then while the camera is idle, the first half of the frame is sent to core. The camera acquires another frame as the first half frame finally is stored on the disk and the second half frame goes to core. The next four segments form the major cycle of data transfer. The camera is first idle while a first half frame enters core and the preceding second half frame leaves it. Then the camera takes data as a second half frame enters core and the preceding first half frame leaves it. The end of the cycle (not shown) occurs under software control. The program must specify the number of frames which are to be read. When enough are acquired, the last frame is sent through the core as before, but there is no camera input. The disk file is closed and the program ends.

Figure 2-2 shows the monitor operation with the boxes representing, from left to right, the disk, core buffers, frame memories, and monitor. Again there is a three stage setup and a four stage cycle. The first half frame is sent from disk to core. It then moves to the frame memory while the second half frame goes to core, and

finally the second half frame is sent to memory as the next half frame leaves the disk. The startup data in a RAM are all high, so the frame 0 which must be displayed will flash an all white screen. Now a whole frame is in memory and the next step can transfer it to the monitor (where it will be repeated several times) while the second memory and the core receive different parts of the second frame. When the second memory is full the monitor begins to take its data from there and the disk sends its next frame through core to the first memory. This alternation continues until the program detects the end of the data and stops the disk. Another three steps (not shown) send the final frame to the monitor, and the program ends.

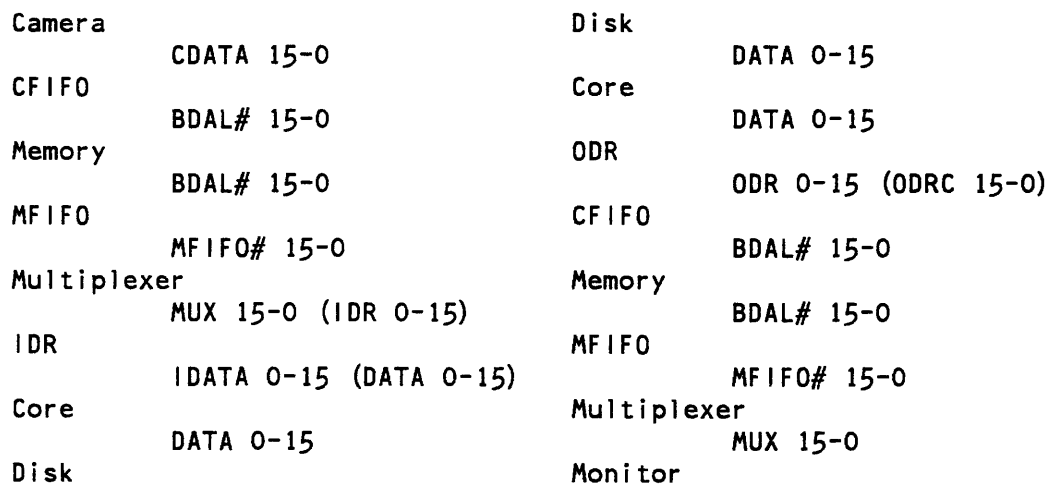


Figure 2-3

Figure 2-3 depicts the data paths for both modes in fuller detail. The left side of each column lists the devices which hold data, and the right side shows the data lines which connect each pair of devices. Signal names including # are SIGNAL1 on CTRL1 and SIGNAL2 on CTRL2. Lines without the # are common to the DMA, CTRL1 and CTRL2. Whether

data are flowing from camera to disk or from disk to monitor, they must pass through seven stages. Looking at MEM1 and its associated control board, CTRL1, the CDATA lines pass the camera data across the system backplane to the CFIFO on the control/buffer board. The BDAL1 lines which transfer them from that buffer, through the memory and out to the MFIFO form a tri-state bus which carries both the memory address and data alternately. The outputs of the second FIFO, MFIFO1 15-0, go to a 2:1 multiplexer which resides on the DMA board. The other multiplexer inputs are the comparable MFIFO2 signals from CTRL2. From the multiplexer the data move to the DMA's Input Data Register (IDR). The Nova-related busses label the most significant bit of data and addresses 0 and the least significant bit 15; the memory and control busses are labelled 15 to 0 to conform to the DEC memory notation. At this junction of the multiplexer and IDR, therefore, the data lines are labelled both MUX 15-0 and IDR 0-15. From the IDR the data move to the core via IDATA 0-15 (on the interface, or DATA 0-15 on the computer data channel bus) and to the disk on DATA 0-15. These DATA lines also carry both addresses and data.

Output to the monitor follows a similar path. Data leave the disk on DATA 0-15, reside in the core and leave there for the DMA's Output Data Register (ODR) on DATA 0-15 again. Similar to the IDR, the ODR's output lines form the bridge between the Nova and DEC notations, so the lines ODR 0-15 are also ODRC 15-0 as they enter the CFIFO on the control board. Again data move from CFIFO to memory to MFIFO on BDAL1 15-0 and from there to the multiplexer via MFIFO1 15-0. The multiplexer, which

this time resides on one of the screen configuraton boards, selects between the MFIFO data of both control boards and sends its MUX 15-0 output to the monitor.

For a given mode and at any given time, each control board will be directing only one half of the data flow operations. For example, if data are moving from disk to MEM1 via CTRL1, then another frame of data is being conveyed from MEM2 to the monitor by CTRL2. After a short time the roles will exchange and the data flow will be from disk to MEM2 and from MEM1 to the monitor. The two modes and two memories thus lead to a system with four states.

Two control signals represent the four states of the machine. Signal M0 (named according to a compatible Nova signal) determines the system mode and is constant throughout an experiment. A high M0 indicates camera mode, and a low M0 defines monitor mode. The second signal, MEMSEL (memory selector), determines which frame memory is linked to the DMA and which is tied to the camera or monitor. If MEMSEL is high, then MEM1 is communicating with the DMA and MEM2 is tied to either the camera or monitor. If MEMSEL is low, then MEM1 is associated with the camera or monitor and it is MEM2 which is tied to the DMA. Table 2-1 illustrates the states. These signals have implications about reading and writing and numerous other effects, but ultimately this paragraph is their definition.

STATE	MO	MEMSEL	MEM1	MEM2
0	0	0	MON	DMA
1	0	1	DMA	MON
2	1	0	CAM	DMA
3	1	1	DMA	CAM

Table 2-1

Reading and writing must be defined relative to the disk, the core, and the frame memories. Table 2-2 defines which input/output or read/write signals are needed by each part of the system.

MO	MEMSEL	Disk	DMA	MEM1	MEM2
0	0	RDB	DCHO	BDIN	BDOUT
0	1	RDB	DCHO	BDOUT	BDIN
1	0	WRB	DCHI	BDOUT	BDIN
1	1	WRB	DCHI	BDIN	BDOUT

RDB = read disk block                      WRB = write disk block  
DCHO = write to data channel from core  
DCHI = read from data channel to core  
BDIN = read frame memory              BDOUT = write to frame memory

Table 2-2

It will at this point be more constructive to analyze the read/write characteristics of the DMA and of the control boards independently, regardless of exactly in which state they occur. The DMA functions are simpler than the control functions and will therefore be described first.

### Chapter 3 The Direct Memory Access Interface

The General Purpose Interface Board is a 15" Nova-compatible printed circuit board that implements most of the complex interface protocol logic. Half of the board is reserved for user logic to modify the general protocol for specific adaptations such as a DMA. Appendix B includes the logic diagrams for the general interface, with the DMA driving signal connections noted.

Like every Nova peripheral, the DMA has a specific device code. The DMA code is 45 octal. Part of the circuit on page 65 recognizes this code whenever it appears in an I/O type program instruction, i.e. when a DATA OUT or DATA IN command is directed to this peripheral. Programmed I/O can be Input or Output on one of three channels, A, B, or C and additionally each instruction can pulse the START, CLEAR, or IOPULSE lines. For example, DOCS 45 sends data to device 45 on channel C and sends a start pulse; DIAP 45 reads channel A from device 45 and sends an IOPLS signal.

Each Nova peripheral reveals its status to the CPU by its BUSY and DONE flags. The BUSY flag is automatically set by a START pulse and is reset when the DONE is clocked. The DMA DONE is wired to set when the WDCNTZ1 (word count zero) signal appears at the end of a half-frame transfer. The CLEAR pulse also automatically sets the DONE and an IORESET clears both status flags. Though peripherals are capable of generating interrupts, the DMA does not use this capability. So its interrupt mask is not set, it never requests an interrupt, and it merely

passes the interrupt priority signal untouched to the next device on the backplane.

The Data Channel is a high speed channel which transfers data between the core and a peripheral and which, once a transfer is initiated, is outside program control. The DMA makes extensive use of this feature. Each peripheral on the bus has a priority based on its physical position. The boards in backplane slots nearest to the CPU have the highest priority. Each device receives a Data Channel Priority signal (DCHPIN) and passes it (DCHPOUT) unaffected if it does not require the data channel, or inverts it if it does require the data channel. Devices further along the backplane will receive the inverted signal and will not be able to request the data channel themselves. The DMA uses the priority line in this simplest way.

Both the disk and the DMA use the data channel simultaneously since one block of data moves between the core and the memory while a second one moves between the disk and the core. Each peripheral enables a data channel request line in its own way, but the actual request is not generated until it is clocked by the request enable (RQENB) which the CPU sends to all peripherals. The data channel logic is in page 66. Once a START pulse is received on device 45, the DMA enables a data channel request whenever the appropriate FIFO is ready. In monitor mode, once the DCHR is enabled it remains so until the entire half-frame block is transferred. In camera mode, the ready line will oscillate as the DMA unloads the FIFO, so the DCHR enable will be clocked periodically by the ready line. If only the DMA were using the channel



it would make a request on every RQENB clock. But the disk also uses the data channel. It is much slower than the DMA, but it has higher priority. Whenever the disk controller requests the data channel, the next RQENB will give it access to the channel and the DMA will wait for its next transfer until the following RQENB. Since the disk is about four times slower, the DMA will have transferred its frame long before the disk has finished its transfer, so the entire disk and memory operation will be limited by the disk rate.

The remaining standard elements of the DMA board are four registers rather than control logic. Data pass from the data channel to the DMA via the Output Data Register (ODR) and from the DMA to the data channel through the Input Data Register (IDR) on page 67. Two other registers act as an address counter and a word counter (see page 66). To prepare the DMA for a data channel transfer the address register must be loaded with the starting address of the core data buffer. The word count register is similarly loaded with MINUS the number of words to be transferred. The address register is loaded by the DATA OUT B line, that is a DOB 45 program instruction, and the word count register is loaded by DATA OUT C, or a DOC 45 instruction. Both counters are clocked by DCHA, the data channel acknowledge which occurs every time a word passes through the data channel. The ODR can be clocked by a DATA OUT A instruction or a DCHO signal. The first responds to the DOA of programmed I/O; the second to the Data Channel Output clock. The IDR is loaded by DCHI, the data channel input signal, gated by a line which tells when the appropriate MFIFO is ready to supply data. In addition

to being loaded, the IDR outputs must be enabled onto the tri-state DATA lines of the computer. They are enabled by a signal that indicates that the DMA is active in the camera mode, and that it is data and not addresses which are needed. The address register output is enabled onto the DATA lines when an address is required. All the registers are cleared by a CLEAR pulse. Compound signals such as the register loads are generated in the custom logic section of the DMA board and connected to the registers through the X and Y wire wrap pins.

Experiment initialization is implemented in both hardware and software. A nonstandard peripheral device can only be used if it has been identified to the interrupt handler and the system map. A control program such as CM11 must therefore begin with these steps. Since most registers in the DMA and the control section are cleared by a CLEAR pulse, it is best also to insert a clear, NIOC 45, at the beginning and end of each program. The initialization of a program must include the loading of the command register with the twelve parameters that direct the screen configuration. In addition to running to the CFIFOs, the ODR lines connect the ODR to the command register. So the single DOA instruction used in a program loads the command word into the command register. One more initial step is required. The number of words in a full frame must be loaded onto the control boards with a DOCP instruction, but a further discussion of this word will appear below.

Ignoring the first and last frame transfers in which only part of the data channel operation is active, a typical monitor mode program would proceed as follows. A DMA transfer is readied by loading one core

buffer's location into the address counter with a DOBP instruction, and the negative of the half-frame size into the word counter with a DOCS instruction. The P triggers the IOPLS and indicates the first half of a frame. The S causes a START pulse and the DMA transfer begins. The data channel requests will trigger DCHO and DCHA signals to clock the data into the ODR and increment the address and word counters. The corresponding disk transfer is begun by loading three accumulators with the relevant core address, the block location in the disk file where data are to be found, and the half-frame word count. An RDB instruction then begins to transfer the second half of the frame from the disk. When the operation is completed, both DMA and disk are reinitialized and the next segments of data are transferred. When it is the second half of a frame on the DMA, there is no IOPLS.

When a new frame is started on the DMA, the IOPLS clears a hardware counter which records the number of times that the frame on the monitor has been repeated. Each vertical drive signal (VDR), i.e. the start of each screen display, increments the counter. As soon as both halves of a frame have passed through the DMA, the program interrogates the counter in a tight loop. When the frame repetition limit is reached, the next VDR triggers the program to toggle MEMSEL and begin the next DMA transfer.

DMA operation in the camera mode is very similar. All the same initializations of the system map, interrupt handler, command register, and control board frame size must be done. Again ignoring the first and last frames, the DMA address and word counters are loaded with DOB and

DOCS instructions, and the first half of a camera frame is clocked through the IDR by DCHI and DCHA signals. The disk begins to receive the second half of the previous frame after it has been initialized with core address, file displacement, and word count. It is at this point that the frame-ignoring counter is checked. If the counter is finished, the next frame is to be read and operation continues with a DOBP/DOCS setup and start, and a disk transfer.

The DMA requires little custom logic, but a number of system control signals which are used by both CTRL1 and CTRL2 are generated on the DMA board. The custom logic for the DMA and for the system control appears as Appendix C. It consists of many isolated segments. Integrated circuits U8 and U9 are the command register which has already been fully described. Similarly, signals DMA1 through DMA5 are used with the registers and data channel requests and have been discussed. U12 through U15 form the 2:1 multiplexer which selects between MFIF01 and MFIF02 when the DMA reads camera data. The master signal, M0, originates at the command register, is inverted for general use, and is also gated by DCHA for the DMA itself. The following several gates in Appendix C simply create the active high equivalents of several active low signals.

The next part of Appendix C is a status register. Two bits are the state variables M0 and MEMSEL. The remaining six bits are three error flags from each control board. ADOV indicates a memory address overflow, i.e. the address counter did not stop at the frame size limit. CINR is Camera (FIFO) Input Not Ready, active when the CFIFO input has

been clocked, but the input word is not ready to accept data. The word is lost in this error condition. Similarly, the Monitor (FIFO) Output Not Ready (MONR) is active when the MFIFO's unload clock has pulsed but there is no word ready to be removed from the FIFO. The word taken as data on that clock will be meaningless. Once set, these bits are active until the next data frame (not frame repetition) is begun. The status word is interrogated by a DIA instruction and printed on the computer console when the frame repetition counter is tested.

The following small circuits create important system control signals. Since the camera and monitor work with a full frame, their memory address limit must be the full frame size rather than the half size. The control boards' word count latches take their input from the output of the DMA's word counter. The word count register is reloaded each half frame by a DOC, but a control board's latch need only be loaded once. The latch must thus be clocked on a DOC, but not on every DOC. To distinguish the single control board load, the latch is clocked on a DOCP instruction only. The IOPLS will follow the DATA OUT C pulse very quickly so the monostable extends DATA OUT C so that it is still active when the pulse occurs. The DOCP is thus recognized for the clock and an isolated DOC or IOPLS will not be captured.

WDCNTZ1D is merely the DMA word count zero, i.e. transfer finished, signal, extended by one memory cycle length. It is used to clear the major timing sync signal when a DMA transfer is finished. If it is not extended the last word is lost.

The next large group of gates extends the START pulse, creates and

toggles MEMSEL, and captures the vertical drive signals from the camera and monitor. Each DMA half-frame transfer begins with a very short START pulse. Many control board signals must be gated by the "DMA active" condition, so STRFFL, the long START FLIP-FLOP, signal becomes active on a START and remains so until the DMA transfer is complete. Other signals are active only at the beginning of the DMA cycle. STRFFS, the short START FLIP-FLOP is set by START and reset by the first memory cycle sync signal.

MEMSEL and its complement are created by a flip-flop. The initial NIOC clears MEMSEL and the IOPLS toggles it at the beginning of every full frame through the DMA. Since the first MEMSEL is 1, and consequently MEM1 is tied to the DMA, the first disk frame in display mode will go to MEM1 and the first camera frame in acquisition mode will go to MEM2.

Both the camera and the monitor generate active low Vertical Drive and Vertical Blanking signals when the beam scan retraces from the lower right corner of the field to the upper left. The Vertical Drive is active for 0.571 ms and the Vertical Blanking for 1.33 ms. Data begin to be displayed or acquired 700 ns after the trailing edge of Vertical Blanking. When a frame memory communicates with the camera or monitor, its timing circuit must be triggered by the appropriate Vertical Drive to anticipate the start of the frame. So VDRC and VDRM are gated by M0 and M0- to become a single VDR.

Finally there is the frame repetition counter. The four bit counter is cleared by IOPLS and clocked by VDR. The counter value, the

carry, and the VDR pass through an open collector buffer directly to the IDATA lines and are interrogated by a DIB instruction. The four bits allow the monitor to repeat a single image up to 15 times, or the camera to skip up to 15 frames. With 60 frames per second, a count of 15 allows only 4 different images per second, 10 allows 6 images, and 3 allows an effective rate of 20 frames per second.

These signals, then, constitute the DMA and overall system logic. The real complexity of the DMA is prewired to fit Nova interface conventions and the system controls are of the general initialization type. The control and timing section which follows is considerably more complex.

## Chapter 4 Memory Control and Buffering

The two frame memories are manufactured by Digital Equipment Corporation and are compatible with DEC's LSI-11 bus and protocol, not with those of the Nova/4. The five LSI-11 signals necessary to drive a memory must therefore be created and synchronized with the operation of the Nova computer, the camera, and the monitor. To accomplish this task, each memory is driven by its own control board which generates addresses and timing signals, and buffers the data that enter and leave the memory. The two control boards, CTRL1 and CTRL2, are identical since they must perform the same tasks in an experiment at alternate times. A switch on one board will be set to respond to MEMSEL and on the other board to respond to MEMSEL-. On the circuit diagrams of Appendix D and in the following discussion, MEMSEL and MEMSEL- refer to the master state variable generated on the DMA board. MEMSEL# and MEMSEL#- (# = 1,2) refer to the version of that signal which board CTRL# has selected. In normal operation, MEMSEL1 is MEMSEL and MEMSEL2 is MEMSEL-. Other signals which occur identically but independently on each control board will also be labelled SIGNAL#. For example, VDR is common to the entire system, but the generalized BSYNC# is BSYNC1 on CTRL1 and BSYNC2 on CTRL2.

The role of the CTRL boards as buffers in the data path will be considered first.

The memories stand between the DMA and the camera or monitor. Both the DMA and camera write to each memory, and the DMA and monitor



read from them. It would be difficult for each of these devices to drive the memory cycle directly, and the system would be highly susceptible to noise and variation in logic delays. To remove the need for this coupling, two 16 word FIFOs buffer the data which enter and leave the memory. The Nova and DEC timing signals can then be completely independent. A second important use of the FIFOs is in the masking of asynchronisms caused by memory refreshing. The dynamic RAMs' internal refresh circuit will steal a memory cycle every 2 ms, or about 8 times per camera or monitor frame. The supposedly steady memory timing will thus be somewhat syncopated. Since the camera and monitor will transfer data at a perfectly steady 1.26 MHz, the pauses in the memory response would cause data to be lost. Since a FIFO's load and unload clocks are independent, the camera and monitor can clock it steadily even when the memory occasionally lags.

A third use of the FIFOs is also important. The memory cycle time is approximately 820 ns, while the 1.26 MHz translates into 794 ns per word. With this difference, the memory could deliver only 62 words in the time the monitor requested a full line of 64 words. If the memory fills a FIFO before a monitor frame starts, and continues to transmit data even while the monitor is in horizontal retrace mode, then data will always be ready for the monitor to unload. Similarly, if a FIFO is empty whenever a camera frame begins, it can accommodate the few words per line which were supplied faster than the memory could receive them. The memory again proceeds while the camera retraces a line, so the extra 2 or 3 words are quickly absorbed and the FIFO will not overflow.

The FIFOs thus perform three crucial tasks. They decouple the memories from the DMA, camera, and monitor; they eliminate any problems which refresh asynchronism would cause; and they allow the memories to be slightly slow and still to meet the required video data rates.

The CFIFO (page 75) buffers data which enter the memory. Both the camera data lines and the DMA's Output Data Register lines pass through tri-state line receiver/drivers to the CFIFO's data inputs. The camera data are enabled when the memory is taking data in acquisition mode, and the DMA lines are enabled in monitor mode. The FIFO is cleared by an IOPLS at the beginning of each DMA frame. It need not be cleared between camera frames because only one frame will actually be acquired between IOPLS assertions, though several will have occurred. If the FIFO receives data from the DMA, it is clocked by DCHO, the Data Channel Out signal; when communicating with the camera, it takes its clock from the camera's horizontal clock. Data are written into the "last word" of the FIFO. When this location is empty, the CIR# ("camera input ready") is active, but as a word is loaded, the ready line drops. An inactive CIR# is an indication not that the FIFO is full, merely that the last location is full and it cannot receive another word. Each word that is loaded will trickle through the FIFO to the last available location, and when a word is unloaded, all data currently in the buffer will move forward one location. As soon as the last location is free again, the CIR# line will rise. The input ready line is used to enable the DMA data channel request and consequently the DCHO clock for the CFIFO, but the camera clock is free-running. If a load clock does occur when CIR#

is inactive, the CINR# error bit is set.

The CFIFO's tri-state outputs are tied to the memory's BDAL# address/data lines. They are enabled only in the states when MEM# is receiving data and only when the memory write cycle requires a data word rather than an address. The unload clock is derived from the memory timing signals which are discussed below. As with the input, there is a COR#, or camera output ready, signal which monitors the first word location. When the unload clock is high and there is a word in the first location, the COR# is active. As the word is unloaded, the COR# drops and rises again when a new word trickles into first position. The unload clock, and indeed the whole memory timing cycle, is gated by the output ready. Since the memory cycle will wait for the FIFO output, no status bit is needed.

Data which are read from the memory are buffered by the MFIFO (see page 76). It is in many ways a mirror image of the CFIFO. The memory's address/data lines form the input to this FIFO and the load clock is a memory signal generated on the CTRL# board. The MIR#, or monitor input ready signal, again indicates when the last word is free to accept data, and since it is on the memory side of the FIFO, it is used to enable the memory cycle. The MFIFO output communicates with the DMA when it receives data in camera mode, and with the monitor as it buffers data in display mode. The DMA's DCHI, Data Channel Input, or the monitor's horizontal clock are the unload clocks. MOR#, the monitor output ready signal, does gate the DMA cycle and its unload clock, but the monitor clock is not inhibited by the FIFO. There is thus a MONR# status bit

which indicates that the output was clocked but no word was ready. The outputs are labelled MFIFO#. They are tri-state, but always enabled because the address/data line multiplexing for the DMA occurs at a later part of the data path, and the monitor data lines carry only data. The output of each MFIFO is sent to a 2:1 multiplexer on the DMA board and to a similar one on a monitor control board. MEMSEL selects which data words reach the DMA and MEMSEL- selects the input to the monitor.

These, then, are the data paths through the control boards. The memory addresses which share the data lines are also generated on these boards.

As described in Chapter 3, the full frame size is latched on the CTRL boards by a DOCP instruction. This number then is the upper limit for the frame memory addresses, and is one input to a comparator (see page 77). An address counter is the input both to the comparator and to a tri-state buffer. While the address is within range, the trailing edge of the memory cycle initiator, BSYNC#, clocks the counter so the address is ready at the start of the next cycle. It is enabled onto the BDAL# address/data memory lines by ADDAT#, one of the timing signals. The minor complexity of the address counter comes in its clearing. The initial programmed CLEAR pulse sets the counter to zero and thereafter whenever the counter reaches the frame size limit, the comparator's equal signal, CMDONE#, resets it. The VDR, i.e. the camera or monitor vertical drive, also resets the counter to ensure that each frame begins at the correct location. Normally a CMDONE# will occur only as often as VDR does, but during testing a small amount of data is repeated several

times to fill a screen, so both reset methods are needed to ensure alignment.

Five basic inputs and one response control the DEC memory. Triggering the lines in the correct order and with the correct relative spacing is straightforward, but as with the address counter, enabling and clearing the various sections at the proper time are intricate operations. Page 78 shows the timing circuitry, and the timing specifications which it must duplicate appear in Appendix F.

The upper left of the diagram describes the three mechanisms for triggering a memory cycle. The upper center enables the timing according to state and FIFO conditions. The lower right shows the final signals which are created. Since the operation is cyclic, there is a great deal of interdependence among the various aspects of the circuit. The specific required delays are generated by Technitrol TTL delay lines, so Schottky integrated circuits are used for the rest of the timing circuit to minimize additional delay. Even with these S integrated circuits, the memory cycle would not have been fast enough to meet the video elements' data rate without the FIFO buffering.

A memory cycle is enabled in states 0 and 3 if the CFIFO output is ready or in states 1 and 2 if the MFIFO input is ready. The first case sets a read/write indicator, RW#, and implies a memory write, or BDOUT# operation; the second sets RW#- enable a memory read with the BDIN# signal. In this case, "state" refers to the state as board CTRL# sees it. CTRL1 will observe the same states as the master timing does since MEMSEL1 is MEMSEL. Since CTRL2 uses MEMSEL- as MEMSEL2, it will perform

its "state 00" actions when the master control is seeing state 01. It is most useful to refer to the states as they appear to the generalized control board rather than to the absolute system control. All of the signals reach the memory board through inverting receivers. They are not buffered as they leave the control board, so they are generated active low. The diagram is consistent with those in the other appendices in labelling active low signals with an overbar. For ease of prose notation, the logic level is not pertinent, so signals will be denoted only by their name, even if they are active low.

When a memory cycle is enabled, there are three ways to trigger BSYNC#, the primary sync signal. One represents the start of a DMA frame, the second is a vertical drive, and the third is a continuation of either mode which triggers a cycle as soon as it is allowed by the specifications. From the vantage of MEM1, the beginning of a monitor mode experiment would proceed as follows. The control board would be unaffected as the first half frame travels from disk to core, but the START pulse will begin to fill the CFIFO. With STRFFS, the extended START pulse, set, the rise of COR1 will trigger the first memory cycle by creating BSYNC1. One memory cycle, or about 820 ns, later the CONTIN1 line is set. If the COR1 is already active, the CONTIN1 triggers the next BSYNC1. If the FIFO is not ready, the active CONTIN1 acts as an enable and the rise of COR1 will clock the BSYNC1. CONTIN1 will drive the memory cycles until the end of the frame.

MEM2 begins this mode in communication with the monitor. (It writes meaningless data for the first frame, but this concession allows

the mechanism to be kept general.) Each VDR runs through a delay line to create FSTRT2, a frame start pulse which begins 75 ns after the VDR ends and persists for 75 ns. Since the MFIFO is empty, MIR2 is active, and the FSTRT2 can trigger the first BSYNC2. As on CTRL1, the end of the first memory cycle sets CONTIN2 and since the MIR2 is still active, the next BSYNC2 is generated. In this mode, the MFIFO will be completely full before the monitor clock actually starts its display, so when MIR is no longer active, the memory cycles will stop until the FIFO begins to be unloaded to the monitor. Once there is room again in the FIFO, the rise of MIR2 clocks the next BSYNC2 and CONTIN2 perpetuates the cycle as before.

Memory cycle initiation is similar in acquisition mode. MEM2 prepares to receive camera data with the first VDR and its FSTRT. Since the camera clock will not have started, COR2 will be inactive and the circuit will wait for data to appear in the CFIFO. Once the camera starts, the rise of COR2 triggers the first BSYNC2 and CONTIN2 continues thereafter. The DMA will initially communicate with MEM1. The MFIFO will at first have no data, and the DMA will wait. Since the empty MFIFO causes an active MIR1, the STRFFS rise will trigger the first memory cycle. CONTIN1 ensures that the cycles proceed, while the DMA begins to unload the MFIFO as soon as a word is ready at the output. Operation again continues until the end of the frame.

BSYNC# is the master memory timing signal. Once it is triggered the remaining signals follow directly. Its assertion will be taken as time 0 in the following description. The address/data lines must carry

the memory address from 75 ns before BSYNC# until 25 ns after it is asserted. ADDAT# is the line which enables the tri-state outputs of either the address or data registers. It is set for data by a copy of BSYNC# which has been delayed 25 ns and is reset for the address 100 ns later (well before the next BSYNC#). The read/write line, BWTBT#, also is required in the -75 ns to +25 ns period so it is triggered by the same delayed BSYNC#. When the memory is being read, the BDIN# line may be asserted at this 25 ns point; when it is a memory write cycle, the comparable BDOUT# can be asserted no sooner than 50 ns after BSYNC#. The memory responds to BDIN# or BDOUT# with its BRPLY#. Timing specifications show that the response will occur within 0 to 125 ns. The system is performing with a response time of 80 to 120 ns. BDIN# and BDOUT# may be released 150 ns after BRPLY# occurs and BSYNC# may be released 100 ns after that. The memory will release BRPLY# soon after BDIN# or BDOUT# is released. All the triggers are created by tapping a delayed version of the appropriate signal at the proper time. The next BSYNC# may follow the end of BRPLY# by no fewer than 300 ns, so it is the trailing edge of BRPLY#, delayed by 300 ns, which sets CONTIN#.

There is one last memory input which is not part of the timing sequence. BBS7# is active when memory bank 7 (the 4 kilowords beginning at location 160000 octal) is addressed. The system should never use this high order bank, but to be accurate, the signal is generated by monitoring the three most significant address bits. It should always be inactive.

If the timing signals can be correctly initiated and perpetuated,



then there remains only to stop them. Since CONTIN# would generate a BSYNC# following every BRPLY#, ad infinitum, it must be cancelled at the end of a frame. If the memory is tied to the DMA, the WDCNTZ1 indicates the end of a frame. This signal is extended slightly to become WDCNTZ1D since the DMA runs a few words ahead of the memory (at least in monitor mode). WDCNTZ1D thus holds the CONTIN# flip-flop's clear low either until the next START, if another half frame is to follow immediately, or until MEMSEL toggles. When the memory is linked with the camera or monitor, an active CMDONE# or VDR indicates the end of a frame. The CONTIN# flip-flop will be cleared by CMDONE# and, somewhat redundantly, by VDR. It remains cleared for the duration of VDR, and only the next FSTRT or STRFFS can restart the memory cycle.

These two chapters describe the video system's data paths and the timing and control signals which direct the data along the proper route. The following chapter will describe special considerations about the design, system testing and performance.

## Chapter 5 Practicalities, Performance, and Maintenance

### 5.1 Practical Considerations

The two standard considerations of time and space are of concern in the video system. Speed is the motivation for several design decisions. Though normally memory space is a constraint, this system currently uses only half of its memory capacity. The space concern is in the great number of signals which are passed between printed circuit boards.

To make the memory cycle time as short as possible, Schottky integrated circuits are required for most of the CTRL board elements. They have the advantage of being 3 to 4 times faster than the Low-power Schottky integrated circuits, but the disadvantage of drawing 2 to 5 times more current than the LS chips. The entire interface, with the exception of the DMA board, draws 6 to 7 amps from the +5V source. A very sturdy power supply is required for these chips, but they do deliver the necessary speed.

The system's interconnections pose a more serious problem than its speed. Data and control lines travel between a memory and its control boards, and to the camera and monitor controllers (Word Packing and Screen Configuration in Figure 1.1) on a wire-wrapped backplane (see Appendix E). These connections are short and pose little difficulty. Almost 100 connections must be made between the DMA board and the two control boards, including four 16 bit wide data paths. A connection

diagram and listing appear in Appendix E. Since the cables are four feet long, twisted pairs are needed to reduce noise pickup. Standard twisted pair cables can carry 13 signals and 13 grounds on the 26 pin connectors which the CTRL boards support. Even with 6 jumpers, this capacity is not sufficient. The solution to this problem of massive data transfer is to use the relatively new Augat SGF twisted pair cables. These cables use 2 of the 26 pins as grounds, and each of the 24 remaining signal lines is twisted with a wire connected to one of these grounds. These cables increase the capacity of the 6 jumpers to 144 signals, which is sufficient. Forty-four signals are common to the DMA board and to both CTRL boards. These lines are carried on the DMA board by a 50 pin connector which splits into two 26 pin connectors on CTRL1. They are wire-wrapped to another pair of connectors and are sent to CTRL2 on two simple 26 pin SGF jumpers. Twenty-four signals are created on CTRL1 and a parallel, but independent, set are generated on CTRL2 (signals such as CINR1 and CINR2, BSYNC1 and BSYNC2). Twenty-four from each board meet in a 50 pin DMA board connector after passing through another split cable. Sixteen signals must pass between one of the monitor control boards and the DMA through a cable which is an SGF 26 pin connector on the monitor board and is attached to a Nova backplane extender on the DMA board. Another short cable connects the two monitor control boards.

For all the complexity of these interconnections, they perform well. A few of the signals which are carried by the 4 foot long split cables seem to pick up interference, so they are RC filtered as they

enter the destination board. The interference does not appear to be widespread; the many data lines, for example, appear to be noise free. The SGF twisted pair cables have solved the problem of board interconnection satisfactorily.

## 5.2 Testing and Performance

Can the video system perform as it was designed to perform? The data paths do all retain their integrity. On a 15K pel monitor screen, a few dozen pixels, or .2%, seem to be improperly transmitted at this stage of testing. The random errors should diminish as the system is fine-tuned.

The DMA and disk have been completely tested as both an input and an output system, and they function exactly as described. The camera mode has not been fully tested; no real data have yet been stored on the disk. The monitor mode has been almost completely debugged. Since much of the timing circuitry is common to both the acquisition and display modes, many problems which were corrected for monitor operation could be simultaneously corrected for operation in the camera mode. Thus testing of the acquisition system will proceed much more quickly than the display testing.

Clearly the Nova interface can be tested independently of the memory system. The disk can be exercised as both source and destination and the disk file and core locations can be examined and checked for data reliability. Once the disk controller is proven to function (and

since it is the standard controller, the proof is immediate) and understood, the DMA can be tested. Data from a known disk file can be checked in core and on the ODR lines to test the DMA as output device. If the DMA is triggered in camera mode while the IDR is driven by a square wave generator, the varying data can be traced in the core and on the disk to verify the DMA's acquisition capability.

The memory control system is best tested in display mode after the DMA is functioning. A known test pattern can then be traced as it passes through the FIFOs and memory to the monitor control boards. The monitor itself is not needed until the final stage of testing; many errors become apparent by observing only the control lines. Testing of the acquisition mode should occur last of all. Since the camera input is less easily specified than a disk file, it is best to fine-tune the system when the memories definitely respond to their controls, the FIFOs definitely can maintain a certain pace, and the control signals definitely occur in their proper relationship.

The only equipment needed to test the DMA is a fast oscilloscope. The pulses such as START and IOPLS last only 400 ns, but the controlling program can be looped so that they can be captured as a periodic wave form by the oscilloscope. Once the memory timing signals come under scrutiny though, a program loop which continually restarts the transfer creates a very artificial environment. A frame transfer is inherently a single action, not a periodic function. In addition, it is the interrelation of many signals of short duration which is crucial to the operation of the system. A logic analyzer is an almost essential tool

in this situation. (A storage scope would be a partial substitute.) A Tektronix 7D02 Logic Analyzer was used to test the control boards. It provides an 8 channel storage capability and can sample all 8 lines at periods varying from 20 ns to 5 ms. It is able (though barely able) to detect the shortest pulses and offsets in the design, which are of only 25 ns duration. (Of course no tool is ever perfect. A 10 ns sample period would have many times been useful, but the timing channel feature was in general completely adequate.) In addition to the timing option, the analyzer provides 24 address line probes and 16 data probes which can be sampled by an external clock (such as BSYNC). Using this tool, the address counter and all the data lines, BDAL, ODR, IDR, CDATA, and MFIFO, could be monitored very easily. The analyzer subtracted weeks from the debugging schedule.

Page E5 is a timing diagram of some memory control and FIFO signals which was compiled from several passes of the logic analyzer. The analyzer samples each signal every 20 ns so the lines are artificially square and are accurate only to 20 ns. Some pulses were verified on a standard oscilloscope. The delay lines all perform very accurately, so though FSTRT- appears to be 80 ns wide, it has been verified as a 75 ns pulse. The diagram was generated in monitor mode when the memory contents were being displayed by the monitor. The VDR/FSTRT initiation is clearly working; BSYNC- falls with FSTRT- since the MFIFO is cleared and ready (MIR active). BRPLY- follows BSYNC- usually within 80 ns, and exactly 300 ns after BRPLY- is released, CONTIN- falls to trigger the next BSYNC-. A 20 to 40 ns propagation

delay is evident between the activation of CONTIN- or MIR and the setting of BSYNC-. A delay of 20 ns is not unreasonable for two logic gates and a flip-flop. The initiation by FSTRT- and clearing by CMDONE- are important features of the diagram which are functioning as designed. Also important is the cycle time. BSYNC- drops approximately every 840 ns, though it must be +/- 20 ns because of the sampling. This rate is slightly slower than expected with Schottky chips, but well within the functional limits. The 16 word deep FIFOs will not be near the point of overflowing.

Page E6 is a comparable diagram of a control board taking data from the DMA. They are drawn to the same scale and sampled at the same rate. It is immediately obvious that this process is occurring much slower than the monitor transfer. The data transfer is enabled when DCHS- drops on a START pulse, but it cannot start until after the first DCHA. DCHA loads the CFIFO's first datum, then when the CFIFO's output becomes ready, COR triggers the first BSYNC-. The second word is loaded into the CFIFO shortly after the next DCHA, but it does not emerge at the other end of the FIFO until 1560 ns later. Once the pattern is established, the BSYNCs are separated by approximately 1760 ns. The memory is working in this state at about half its capability. It seems inefficient to load one word, let it move to the front of the FIFO, load the next word, let it travel through 16 locations, et cetera. A different scenario would be to fill the CFIFO with the DMA before the memory begins to empty it. There would then be a much shorter delay between COR signals since a word would only have to move one position to

make the output ready again. The memory could then work at approximately its 840 ns cycle time. But because the DMA still can work only half as fast as the memory, only the first 30 words of a 64 word line would be moved at the faster rate, and in addition all the time needed to fill the FIFO initially must be added to the frame transfer time. The alternative approach does not offer any real increase in speed. The DMA transfer is inherently unable to use the memory's full capabilities, so the straightforward design performs as well as any more convoluted approach.

Both the DMA and monitor aspects of display mode perform largely as expected. The crucial part of the design, i.e. the memory timing signals, occur predictably and accurately. A few problems, such as a horizontal band at the half-frame point of the screen, an unexplained vertical band, and some non-random noise, still plague the display, but their elimination will require only minor design changes. The bulk of the controller design has been proven to function as desired.

Acquisition mode has not been fully tested, though since it clocks the same FIFOs and requires the identical timing signals, no major design errors are expected to appear.

### 5.3 Maintenance -- Past and Future Flaws

As all the capabilities of the video system are tested in actual experiments, some design flaws will certainly appear for the first time. It will be useful for those responsible for the system at that time to



be aware of the classes of problems which have been detected thus far, since the same classes will tend to recur in the future.

At the surface, errors are either clearly logical or totally non-logical. If signal A occurs at the wrong time, but its trigger, signal B, also occurs then, the chain of influence can gradually be traced until the incorrect signal is found and corrected. Alternatively, if signal A is incorrect and all of its sources appear to be functioning properly, or worse yet, if signal A is intermittently incorrect, then the simple logical trace will not easily isolate the problem. In the early stages of system testing, usually an observable error is the result of several equally major flaws, and the checkpoint can only be passed when all the flaws have been eliminated. Once the system is functional though, errors should be expected to have only one major cause.

The non-logical errors have two common causes, bad electrical connections and noise. The problems which are detected in the earliest test stages are those such as open collector outputs which are not pulled up, a line that is driven by a signal and its complement, or a chip that is not grounded. All have serious effects, but are relatively easy to detect with an oscilloscope, and rarely occur in later stages of testing.

Noise can be a very serious problem. It is visible in the video system primarily when edge-triggered signals are triggered at unexplained times. "Unexplained" may or may not mean random; one flip-flop consistently cleared at the same incorrect time, although its

clear line appeared steady. The logic analyzer was least useful when an error was caused by noise. If, as in the above case, a flip-flop were cleared incorrectly and all its inputs were steady on the analyzer display, there could still be, and would indeed have to be, a glitch on the clear or clock which occurred between the 20 ns samples. In testing this system, whenever a logic change simply could not occur, but did occur, the solution was to filter one of the inputs.

These glitches have apparently occurred for two reasons. A very small number of the signals which travel through the 4 foot split cables were noisy at the destination end of the cable, but not at the source. Fortunately this interference is not widespread. Some glitches were not caused by noise, but apparently by varying delays within integrated circuits. For example, if the four chips of a comparator and the four chips of a counter which create its input are not well synchronized, the comparator outputs could momentarily be at the wrong level and create a glitch. The electrical, noise, and glitch errors are difficult to detect, but relatively simple to correct with RC filters. As the system is tested with larger frames and consequently faster camera and monitor clocks, the integrated circuits will be forced to work at speeds closer to their limits. Glitches and noise can probably be expected in the future.

Logical errors can also be of two types. The first occur when a desired logic equation is not implemented correctly. These errors are relatively easy to detect with the analyzer; simply monitor all relevant signals and one or more will be at the wrong level. Some of these

errors may arise in the future, but most should have been eliminated in early testing. If triggers are inverted or occur on the wrong conditions, the error is serious enough to stop operation completely. They are not subtle errors.

The final class of error is often very subtle. These flaws are design misconceptions, or decisions made in ignorance of all their implications. Errors of this type have been the most common, and will probably continue to be the most prevalent as the system is fully exercised. The crucial elements of the system control are triggering and cancelling the timing signals. The same basic FIFO and memory elements are used in each state, but they are controlled by three different devices, the camera, monitor, and DMA. There must therefore be 2 to 4 versions of every clock, every clear, and every enable. They must all be gated by different combinations of the state variables, M0 and MEMSEL, so that alternate versions of the same trigger do not interfere with each other. Some signals occur, for example, in states 0 and 1, some in states 0 and 2, others only in state 3. Several gates were designed with the correct versions of the inputs, but with incorrect gating; e.g. a clock which should occur in state 1 would be dominating in state 2. A variation of the problem which has occurred regularly, is the creation of overqualified signals. The memories are actually quite independent of the camera, monitor, and DMA, and conditions of one element should not usually be required to trigger another element. The final version of many signals is much cleaner than the original.

Many of these design problems cannot be predicted in the design phase, or even in the early test phase. The interrelationships among aspects of the system are simply too numerous and intricate to analyze fully in theory. The logic analyzer is here essential for seeing exactly what the signals actually do, rather than what they are intended to do. Though the system is largely functional, design errors of both the under- and overqualified types are certain to arise as the final states are tested.

The intricacy of the controls is not surprising. Any system with bi-directional data paths will require systematic enabling of each segment of the path. This system is perhaps more complex than one in which data travel from peripheral to disk and back to the same peripheral, because in this case the peripheral is actually two devices. The design with heavily qualified data controls is sound. A simplification of the FIFO controls would require extra, highly redundant FIFOs and the complexity would merely shift to another part of the design.

Unpredictable and irregular delays, members of an error class which would be expected in a system in which timing is so important, are notably missing. The TTL delay lines are very accurate and consistent, and the Schottky integrated circuits introduce very little unwanted delay into the timing path. The memory timing signals occur consistently according to specifications.

It is probable that some design and glitch errors will arise as the system is first exercised, but once the operation of various states

is verified, few flaws should be expected.

## Chapter 6 Summary

Investigation of algorithms which narrow the bandwidth of video signals requires a means of collecting raw data and of observing processed data. A computer system which acquires and stores video camera data and which retrieves and displays the images on a video monitor is an essential tool for studies in the growing field of video bandwidth compression. The design and construction of such a system has been discussed in this thesis.

The video interface system comprises a Nova/4 computer with its 130 MB disk and its Direct Memory Access unit, a video camera and video monitor, two 64KB frame storage memories, and six printed circuit boards of driving logic. It operates in two modes. As an acquisition system, data sampled by the camera flow through the frame memories and the DMA to the disk. As an output device, the data are sent from the disk through the DMA and memories, and are displayed on the monitor. The hardware must be able to sustain the 60 frame/second video rate, while maintaining the integrity of finely quantized images. The speed requirements necessitate the use of fast memories, fast integrated circuits, and a fast disk. The disk's 320 KBps data rate is the primary system constraint. If it were replaced though, the memories could not process the same image size at any greater speed, so the system performance of 20 standard frames per second through the DMA and 60 fps through the video elements seems to be fixed.

Critical aspects of the system design are the memory timing and

data buffering. The control lines for the DEC RAMs must be driven by custom logic since the Nova and DEC controls are not compatible. The memories can be pushed to a 820 to 840 ns cycle time. Operation at this rate requires accurate and predictable control signals. The Schottky chips which create the timing signals are pushed to perform at the high switching rate with minimal internal delay and high noise immunity. In general, the timing circuit does meet the speed and accuracy requirements.

Two 16 word FIFOs are used to buffer the data which enter and leave each memory. They perform three crucial functions. The FIFOs decouple the memory from the DMA, camera, and monitor; they eliminate asynchronism problems caused by memory refresh; and they allow the memory to work slightly slower than the camera or monitor and yet to maintain an adequate data rate. By allowing all the segments of the system to work independently, the FIFOs have simplified both the design and testing of the interface.

The video system is a versatile instrument. Though the data paths require some complex control, they are effectively very general. The system will proceed at video rates regardless of the nature of the data, so either video or previously stored still images can be displayed on the monitor. The software-controlled frame rate allows experimentation with image degradation caused by flicker. The horizontal, vertical, and pel resolution parameters permit an image to be quantized in twelve different ways (from 60 x 128 x 4 through 240 x 128 x 8), and variable display sizes can augment or cancel some of the effects of the sampling

resolution by expanding or compressing the images. The video interface system will support a wide range of visual experiments.



Appendix A: Control Software

CC11: Acquisition Mode

CM11: Display Mode

```

        .TITL CC11
        .TXTM 1
        .TXTN 0
        .EXTD TCHR
        .EXTN .BINO
        .EXTN .UCEX
        .ZREL
F1:     FRM1
F2:     FRM2
ERR:    ERR
BINO:   .BINO
CRET:   15
CR:     3207
FREP:   4
MSK37:  37
MSK40:  40
WDC:    7400
NFRM:   400
FSZ:    17000
C5:     5
C10:    10
TMP1:   0
DEV:    45
FILE:   .+1*2
        .TXT "VIDEO.DD"
        .NREL
DCT:    .+1
        .BLK 3
                                ;SETUP
CC11:   NIOC 45
        LDA 0,FILE
        SUB 1,1
        .SYSTEM
        .OPEN 0           ;OPEN CHANNEL 0 FOR DISK
        JMP @ER
        LDA 0,DEV
        LDA 1,DCT
        MOVOL 1,1
        MOVR 1,1         ;BIT 0=1 FOR DATA CHAN
        LDA 2,C5
        .SYSTEM
        .IDEF           ;IDENTIFY DMA TO INT HANDLER
        JMP @ER
        LDA 1,F1
        .SYSTEM
        .STMAP         ;PUT DMA ON SYSTEM MAP
        JMP @ER
        LDA 3,FSZ
        DOCP 3,45       ;FRAME SIZE TO CAM ADDR REG
        LDA 1,FREP
        LDA 2,MSK37
LP0A:   DIB 0,45
        AND 2,0
        SUBZ 1,0,SNC   ;CAM ON LAST REPEAT?
        JMP LP0A
        LDA 2,MSK40
LP0B:   DIB 0,45
        AND 2,0,SZR   ;READY TO SWAP MEMORIES?
        JMP LP0B
                                ;FIRST FRAME -- CAMERA (& GARBAGE DMA) ONLY
        LDA 3,CR       ;COMMAND REGISTER

```

```

DOA 3,45
LDA 3,F1 ;CORE ADDR
DOBP 3,45 ;P(PULSE) ONCE PER DMA FRAME
LDA 3,WDC ;WDCNT, S(TART) ONCE PER DMA XFER--2/FRM
NEG 3,3
DOCS 3,45 ;START 1ST CAM FRM & OTH HALF FRM FROM FMEM
W1: SKPDN 45 ;WAIT FOR DMA
    JMP W1
    LDA 3,F1 ;OVERWRITE OTH DMA FRAME -- WAS GARBAGE
    DOB 3,45 ;NO PULSE
    LDA 3,WDC
    NEG 3,3
DOCS 3,45 ;1ST HALF FRAME FROM FMEM
W2: SKPDN 45 ;WAIT FOR DMA
    JMP W2
    LDA 1,FREP
    LDA 2,MSK37
    STA 1,TMP1
LP1A: DIB 0,45
      AND 2,0
      SUBZ 1,0,SNC ;CAM ON LAST REPEAT?
      JMP LP1A
    LDA 2,MSK40
LP1B: DIE 0,45
      AND 2,0,SZR ;READY TO SWAP MEMORIES?
      JMP LP1B
    LDA 1,TMP1
                                     ;CAMERA, DMA, AND DISK
GO1: LDA 3,F2
      DOBP 3,45 ;PULSE STARTS 2ND CAMERA FRAME
      LDA 3,WDC
      NEG 3,3
      DOCS 3,45 ;2ND HALF FRAME FROM FMEM
      LDA 0,F1
      SUB 1,1
      LDA 2,WDC
      .SYSTEM
      .WRB 0 ;1ST HALF FRAME TO DISK
      JMP @ER
W3: SKPDN 45 ;WAIT FOR DMA
    JMP W3
                                     ;MAJOR LOOP FOR DISK, DMA AND CAMERA
FRM: LDA 3,F1
      DOB 3,45
      LDA 3,WDC
      NEG 3,3
      DOCS 3,45 ;1ST HALF OF NTH FRAME FROM FMEM
      LDA 0,F2
      LDA 2,C10
      ADD 2,1 ;IF AC1 USED FOR OTHER THAN WRB, STORE & RESTORE IT
      LDA 2,WDC
      .SYSTEM
      .WRB 0 ;2ND HALF OF N-1ST FRAME TO DISK
      JMP @ER
W4: SKPDN 45 ;WAIT FOR DMA
    JMP W4
    DIA 0,45
    STA 1,TMP1
    JSR @BINO
    @TCHR
    LDA 0,CRET

```

```

JSR @TCHR
LDA 1,FREP
LDA 2,MSK37
LP2A: DIB 0,45
      AND 2,0
      SUBZ 1,0,SNC ;CAM ON LAST REPEAT?
      JMP LP2A
      LDA 2,MSK40
LP2B: DIB 0,45
      AND 2,0,SZR ;READY TO SWAP MEMORIES?
      JMP LP2B
      LDA 1,TMP1
G02:  LDA 3,F2
      DOBP 3,45
      LDA 3,WDC
      NEG 3,3
      DOCS 3,45 ;2ND HALF OF NTH FRAME FROM FMEM
      LDA 0,F1
      LDA 2,C10
      ADD 2,1
      LDA 2,WDC
      .SYSTEM
      .WRB 0 ;1ST HALF OF NTH FRAME TO DISK
      JMP @ER
W5:   SKPDN 45 ;WAIT FOR DMA
      JMP W5
      LDA 0,NFEM
      SUB 1,0,SNR ;IF MORE FRAMES NEEDED, REPEAT MAJOR LOOP
      JMP LST
      MOVL 0,0,SNC
      JMP FRM
                                           ;LAST FRAME -- DISK AND DMA ONLY
LST:  LDA 3,F1
      DOB 3,45 ;NO PULSE
      LDA 3,WDC
      NEG 3,3
      DOCS 3,45 ;1ST HALF OF LAST FRAME FROM FMEM
      LDA 0,F2
      LDA 2,C10
      ADD 2,1
      LDA 2,WDC
      .SYSTEM
      .WRB 0 ;2ND HALF OF 2ND-LAST FRAME TO DISK
      JMP @ER
W6:   SKPDN 45 ;WAIT FOR DMA
      JMP W6
      LDA 3,F2
      DOB 3,45
      LDA 3,WDC
      NEG 3,3
      DOCS 3,45 ;2ND HALF OF LAST FRAME FROM FMEM
      LDA 0,F1
      LDA 2,C10
      ADD 2,1
      LDA 2,WDC
      .SYSTEM
      .WRB 0 ;1ST HALF OF LAST FRAME TO DISK
      JMP @ER
W7:   SKPDN 45 ;WAIT FOR DMA
      JMP W7
      LDA 0,F2

```

```
LDA 2,C10
ADD 2,1
LDA 2,WDC
.SYSTM           ;2ND HALF OF LAST FRAME TO DISK
.WRB 0
JMP @ER
DIA 0,45
JSR @BINO
@TCHR
.SYSTM
.RTN
ERR: .SYSTM
.ERTN
JMP ERR
FRM1: .BLK 7400
FRM2: .BLK 7400
.END CC11
```

```

        .TITL CM11
        .TXTM 1
        .TXTN 0
        .EXTD TCHR
        .EXTN .BINO
        .ZREL
F1:     FRM1
F2:     FRM2
ER:     ERR
BINO:   .BINO
CRET:   15
CR:     3206
FREP:   4
MSK37:  37
MSK40:  40
WDC:    7400
NFRM:   400
FSZ:    17000
C5:     5
C10:    10
TMP1:   0
DEV:    45
FILE:   .+1*2
        .TXT "VIDEO.DD"
        .NREL
DCT:    .+1
        .BLK 3

                                ;SETUP
CM11:   NIOC 45
        LDA 0,FILE
        SUB 1,1
        .SYSTEM
        .OPEN 0           ;OPEN CHANNEL 0 FOR DISK
        JMP @ER
        LDA 0,DEV
        LDA 1,DCT
        MOVOL 1,1
        MOVR 1,1          ;BIT 0=1 FOR DATA CHAN
        LDA 2,C5
        .SYSTEM
        .IDEF             ;IDENTIFY DMA TO INT HANDLER
        JMP @ER
        LDA 1,F1
        .SYSTEM
        .STMAP           ;PUT DMA ON SYSTEM MAP
        JMP @ER
        LDA 3,FSZ
        DOCP 3,45        ;FRAME SIZE TO MON ADDR REG
                                ;FIRST FRAME -- DISK ONLY
        LDA 0,F1
        SUB 1,1
        LDA 2,WDC
        .SYSTEM
        .RDB 0           ;1ST HALF FRAME FROM DISK
        JMP @ER
                                ;DISK AND DMA
        LDA 3,CR         ;COMMAND REGISTER
        DOA 3,45
        LDA 3,F1        ;CORE ADDR
        DOBP 3,45       ;P(ULSE) ONCE PER DMA FRAME
        LDA 3,WDC       ;WDCNT, S(TART) ONCE PER DMA XFER--2/FRM

```

```

NEG 3,3
DOCS 3,45 ;START 1ST HALF FRM TO MEM1
LDA 0,F2
LDA 2,C10
ADD 2,1 ;IF AC1 USED FOR OTHER THAN RDB. STORE & RESTORE IT
LDA 2,WDC
.SYSTM
.RDB 0 ;2ND HALF FRM FROM DISK
JMP @ER
W1: SKPDN 45 ;WAIT FOR DMA
JMP W1
LDA 3,F2
DOE 3,45 ;NO PULSE
LDA 3,WDC
NEG 3,3
DOCS 3,45 ;2ND HALF FRM TO MEM1
LDA 0,F1
LDA 2,C10
ADD 2,1
LDA 2,WDC
.SYSTM
.RDB 0 ;1ST HALF OF 2ND FRM FROM DISK
JMP @ER
W2: SKPDN 45 ;WAIT FOR DMA
JMP W2
STA 1,TMP1
LDA 1,FREP
LDA 2,MSK37
LP0A: DIB 0,45
AND 2,0
SUBZ 1,0,SNC ;MON ON LAST REPEAT?
JMP LP0A
LDA 2,MSK40
LP0B: DIB 0,45
AND 2,0,SZR ;READY TO SWAP MEMORIES?
JMP LP0B
LDA 1,TMP1
;MAJOR LOOP FOR DISK, DMA AND MONITOR
FRM: LDA 3,F1
DOBP 3,45
LDA 3,WDC
NEG 3,3
DOCS 3,45 ;1ST HALF OF NTH FRM TO MEM
LDA 0,F2
LDA 2,C10
ADD 2,1
LDA 2,WDC
.SYSTM
.RDB 0 ;2ND HALF OF NTH FRM FROM DISK
JMP @ER
W3: SKPDN 45 ;WAIT FOR DMA
JMP W3
LDA 3,F2
DOE 3,45
LDA 3,WDC
NEG 3,3
DOCS 3,45 ;2ND HALF OF NTH FRM TO MEM
LDA 0,F1
LDA 2,C10
ADD 2,1
LDA 2,WDC

```

```

        .SYSTEM
        .RDB 0          ;1ST HALF OF N+1ST FRM FROM DISK
        JMP @ER
W4:     SKPDN 45        ;WAIT FOR DMA
        JMP W4
        DIA 0,45
        STA 1,TMP1
        JSR @BINO
        @TCHR
        LDA 0,CRET
        JSR @TCHR
        LDA 1,FREP
        LDA 2,MSK37
LP1A:   DIB 0,45
        AND 2,0
        SUBZ 1,0,SNC ;MON ON LAST REPEAT?
        JMP LP1A
        LDA 2,MSK40
LP1B:   DIB 0,45
        AND 2,0,SZR ;READY TO SWAP MEMORIES?
        JMP LP1B
        LDA 1,TMP1
GO1:    LDA 0,NFRM
        SUB 1,0,SNR  ;IF MORE FRAMES ON DISK REPEAT MAJOR LOOP
        JMP LST
        MOVL 0,0,SNC
        JMP FRM
                                ;LAST FRAME -- NO DISK
LST:    LDA 3,F1
        DOBP 3,45
        LDA 3,WDC
        NEG 3,3
        DOCS 3,45     ;1ST HALF OF LAST FRM TO MEM
W5:     SKPDN 45        ;WAIT FOR DMA
        JMP W5
        LDA 3,F2
        DOB 3,45
        LDA 3,WDC
        NEG 3,3
        DOCS 3,45     ;2ND HALF OF LAST FRM TO MEM
W6:     SKPDN 45        ;WAIT FOR DMA
        JMP W6
        DIA 0,45
        JSR @BINO
        @TCHR
        STA 1,TMP1
        LDA 1,FREP
        LDA 2,MSK37
LP2A:   DIB 0,45
        AND 2,0
        SUBZ 1,0,SNC ;MON ON LAST REPEAT?
        JMP LP2A
        LDA 2,MSK40
LP2B:   DIB 0,45
        AND 2,0,SZR ;READY TO SWAP MEMORIES?
        JMP LP2B
        LDA 1,TMP1
GO2:    .SYSTEM
        .RTN
ERR:    .SYSTEM
        .ERTN

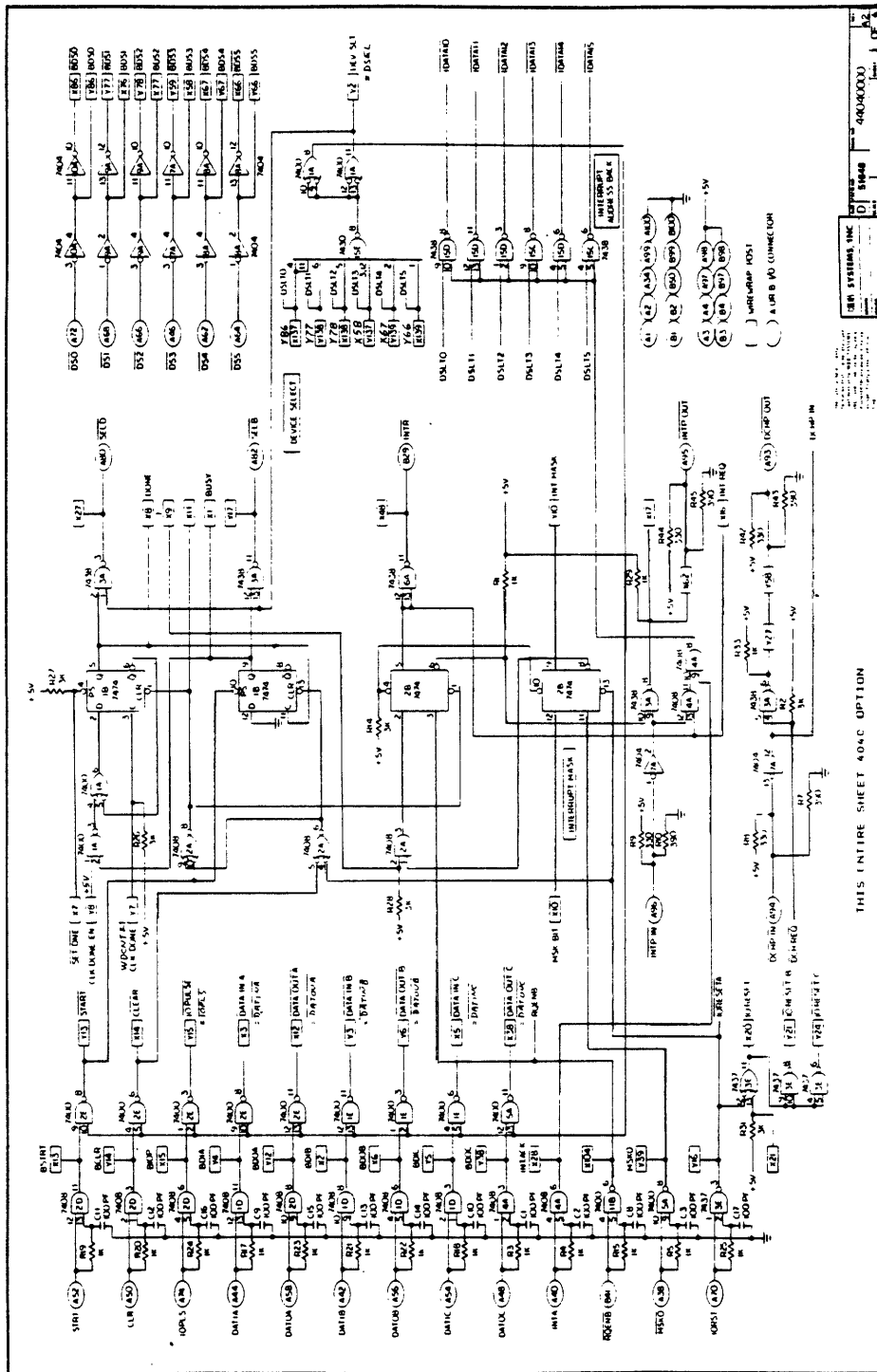
```



```
JMP ERR
FRM1: .BLK 7400
FRM2: .BLK 7400
      .END CM11
```

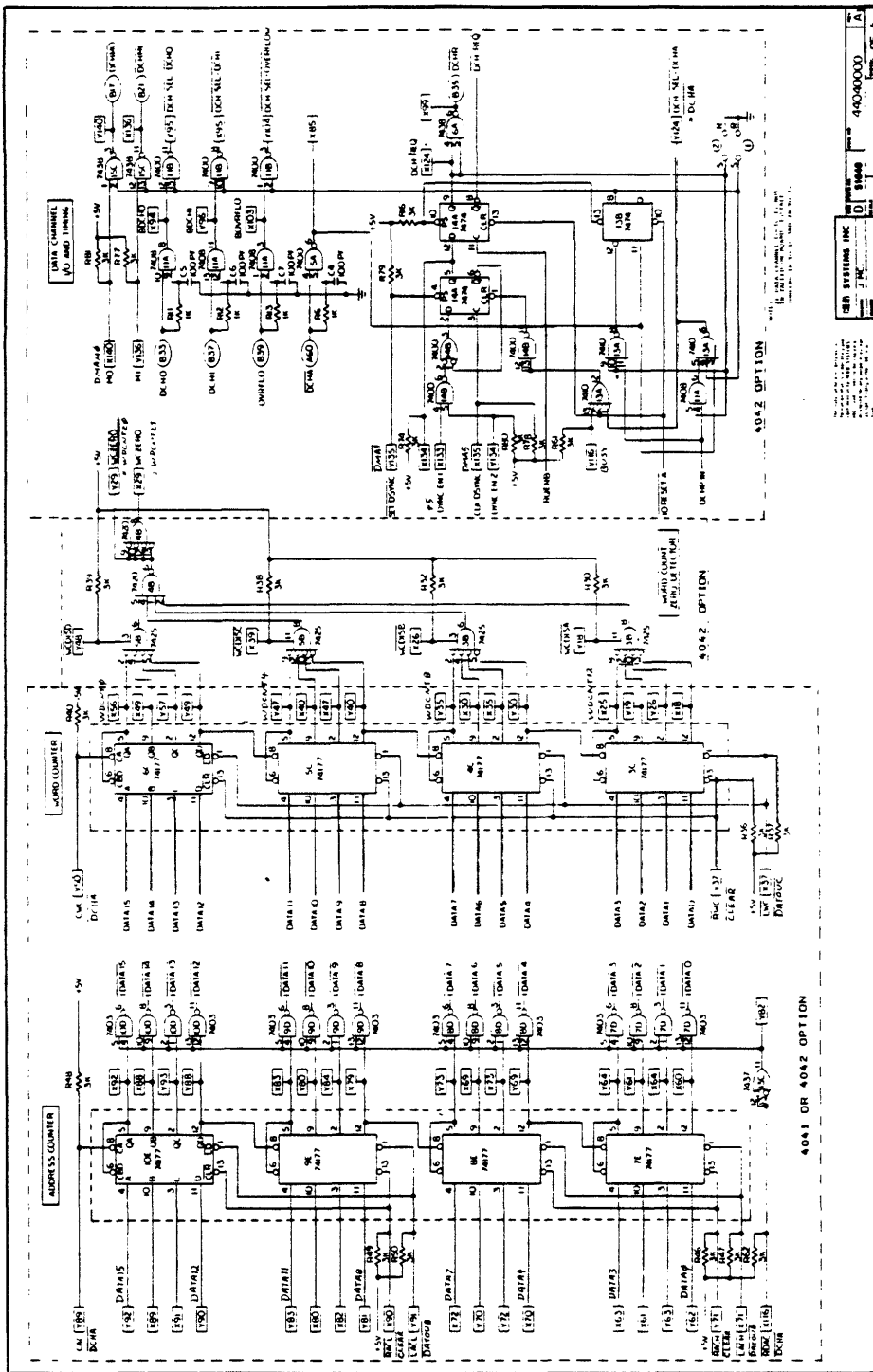
## Appendix B: DMA Circuit Diagrams

Labels A1 to A100 and B1 to B100 are backplane connector pins. S1 to S33 and T1 to T33 are wire-wrap connections to unused A and B pins. X1 to X140 and Y1 to Y140 are wire-wrap connections to prewired integrated circuits on the board, and the means of joining user logic and the general logic. C1-1 to C1-50 and C2-1 to C2-50 are jumper connections to the CTRL boards.



THIS ENTIRE SHEET 404C OPTION

74180 (18)  
 74138 (19)  
 7414 (20)  
 7400 (21)  
 7401 (22)  
 7402 (23)  
 7403 (24)  
 7404 (25)  
 7405 (26)  
 7406 (27)  
 7407 (28)  
 7408 (29)  
 7409 (30)  
 7410 (31)  
 7411 (32)  
 7412 (33)  
 7413 (34)  
 7415 (35)



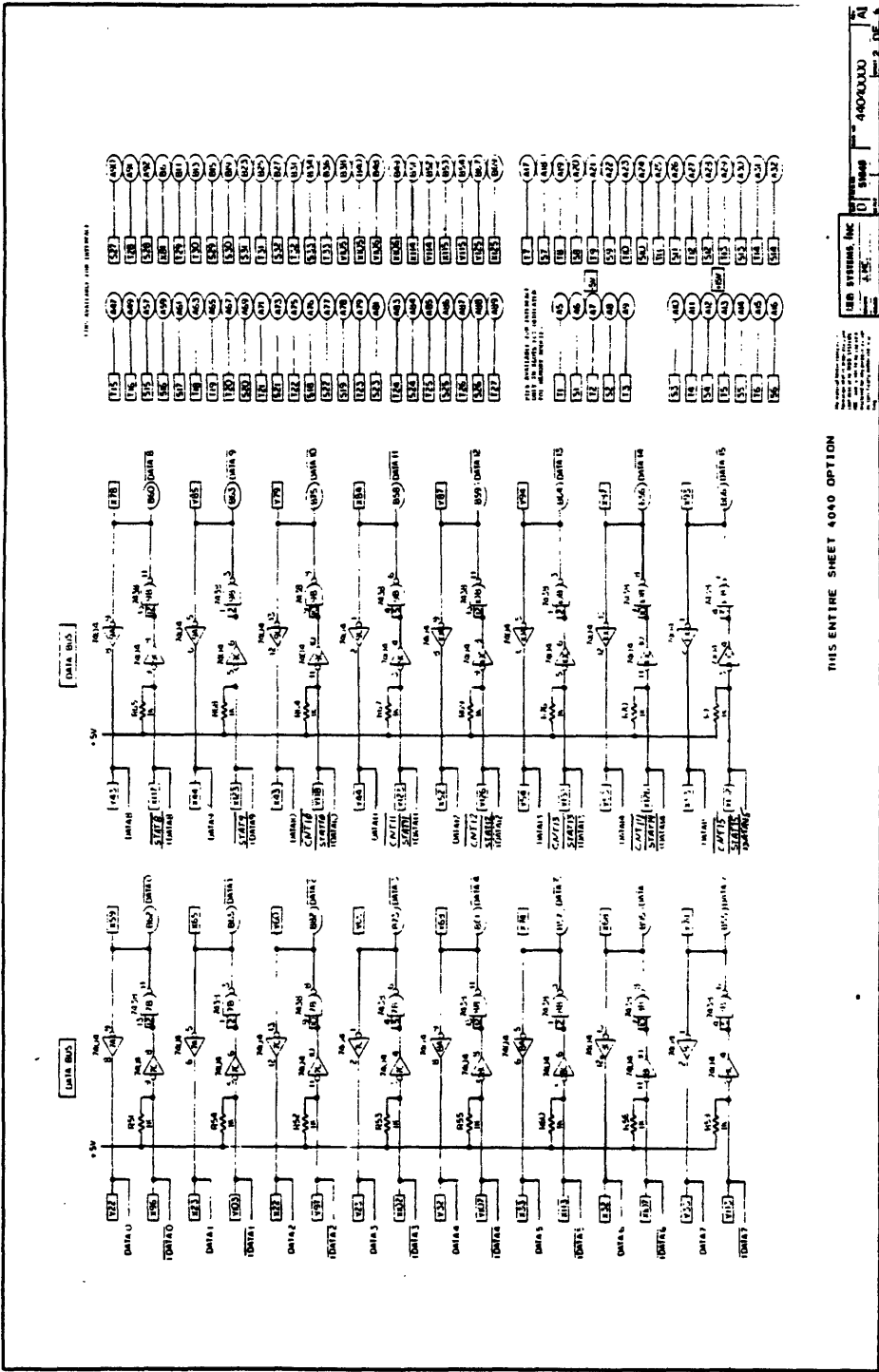
4042 OPTION

4042 OR 4042 OPTION

4041 OR 4042 OPTION

IBM SYSTEMS INC  
 0 81440  
 44040000

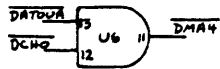
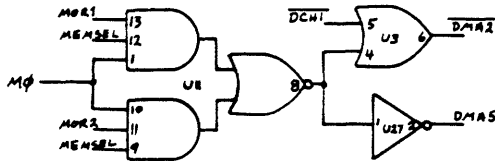
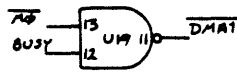
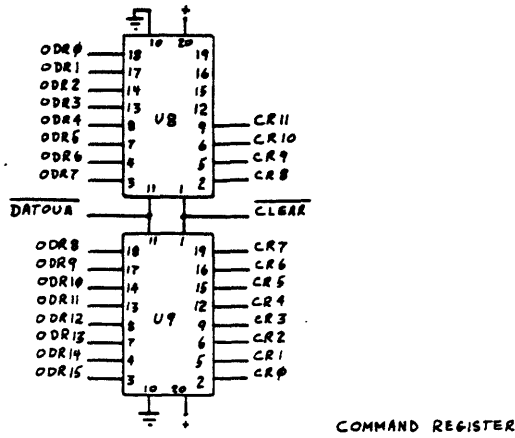




THIS ENTIRE SHEET 4040 OPTION

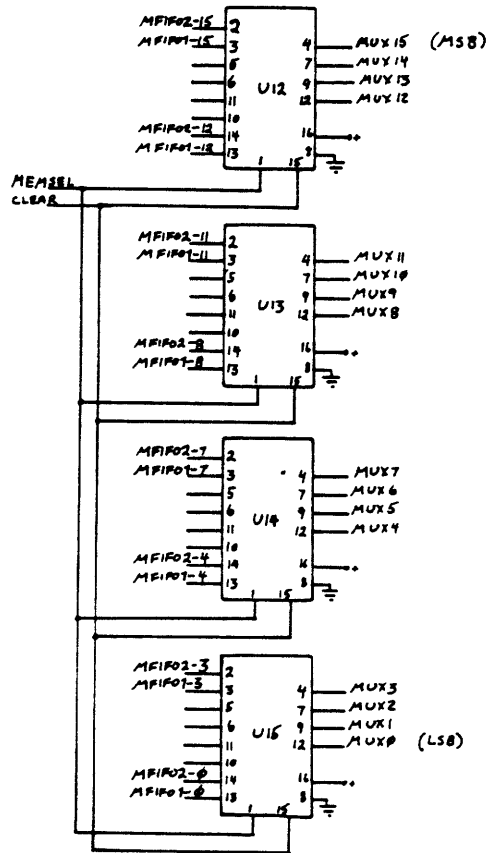
IBM SYSTEMS INC  
 4400000  
 4040

## Appendix C: DMA and System Logic Circuit Diagrams



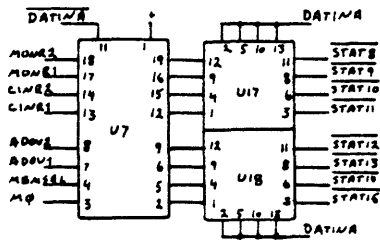
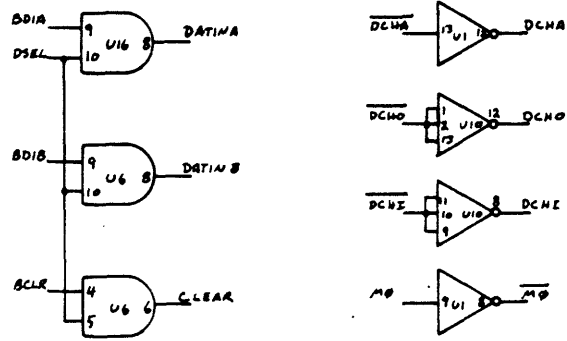
- 74LS00 · U19
- 74LS04 · U27
- 74LS08 · U6
- 74LS10 · U10
- 74LS32 · U3
- 74LS51 · U11
- 74LS273 · U8, U9



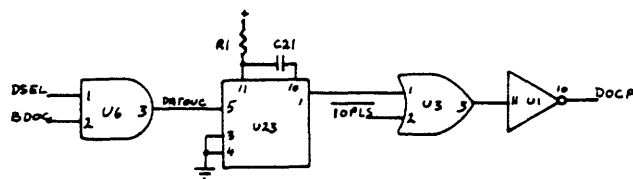


74LS157-U12, U13, U14, U15

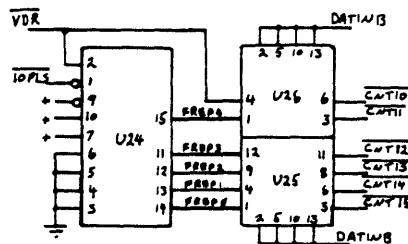
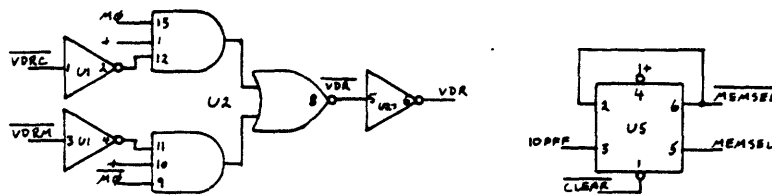
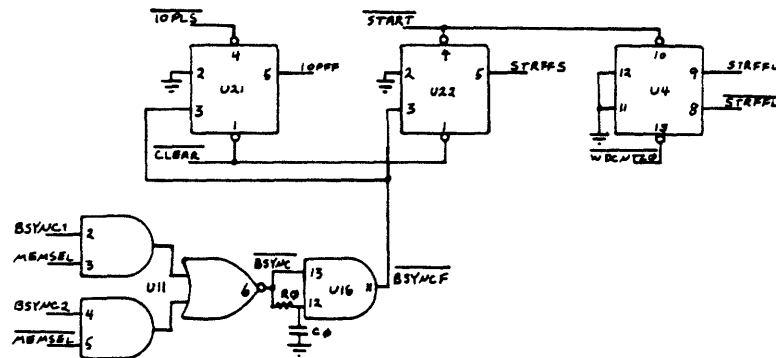
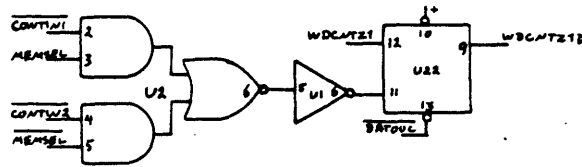
MULTIPLEXER



STATUS REGISTER



- 74LS05 · U1, U18
- 74LS04 · U1
- 74LS08 · U6, U16
- 74LS10 · U10
- 74LS32 · U3
- 74LS121 · U23
- 74LS273 · U7
- 10KΩ · R1
- 510pF · C21

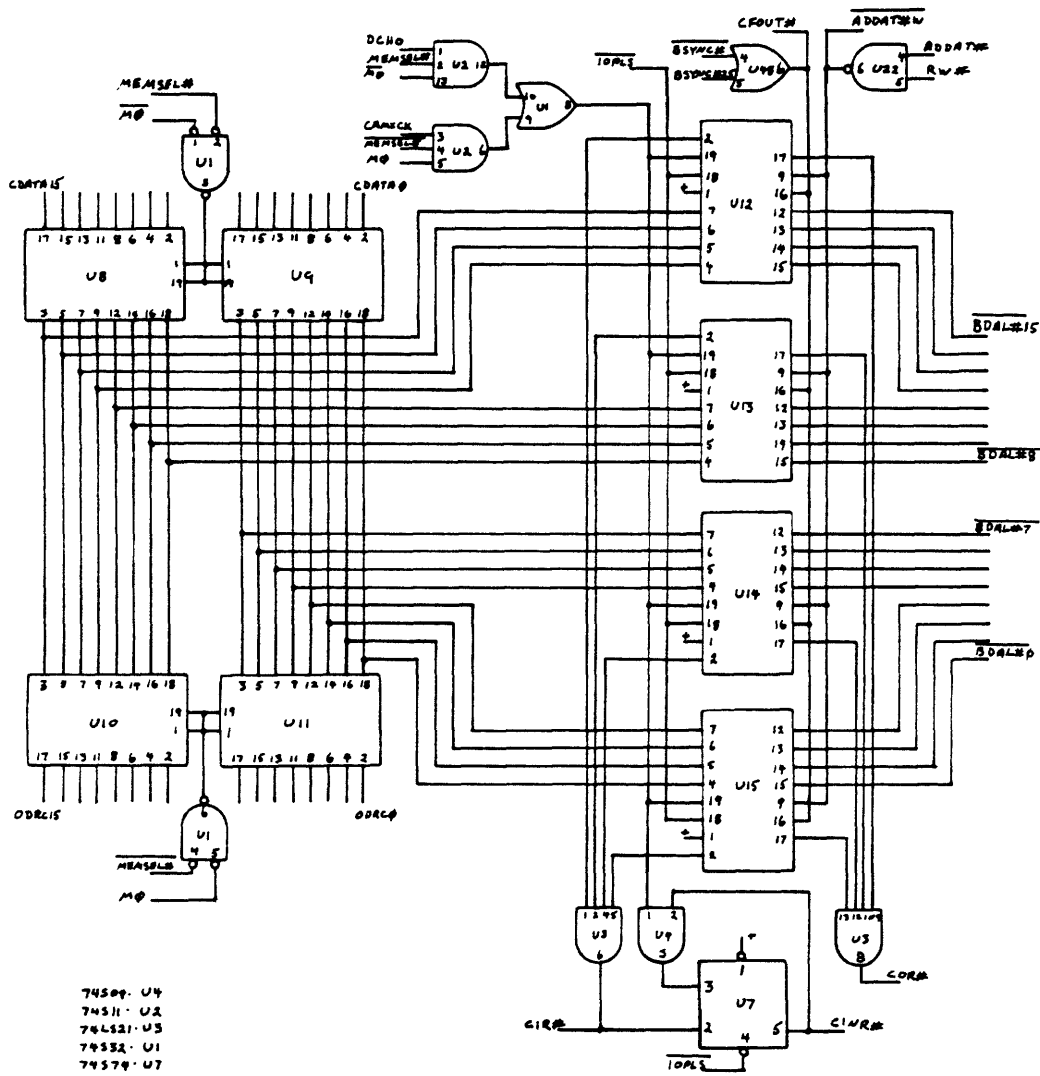


74LS03 - U15, U16  
 74LS04 - U1, U27  
 74LS08 - U16  
 74LS51 - U2, U11  
 74LS74 - U4, U5, U21, U22  
 74LS161 - U24  
 1KΩ R0  
 300pF C0

REPETITION COUNTER

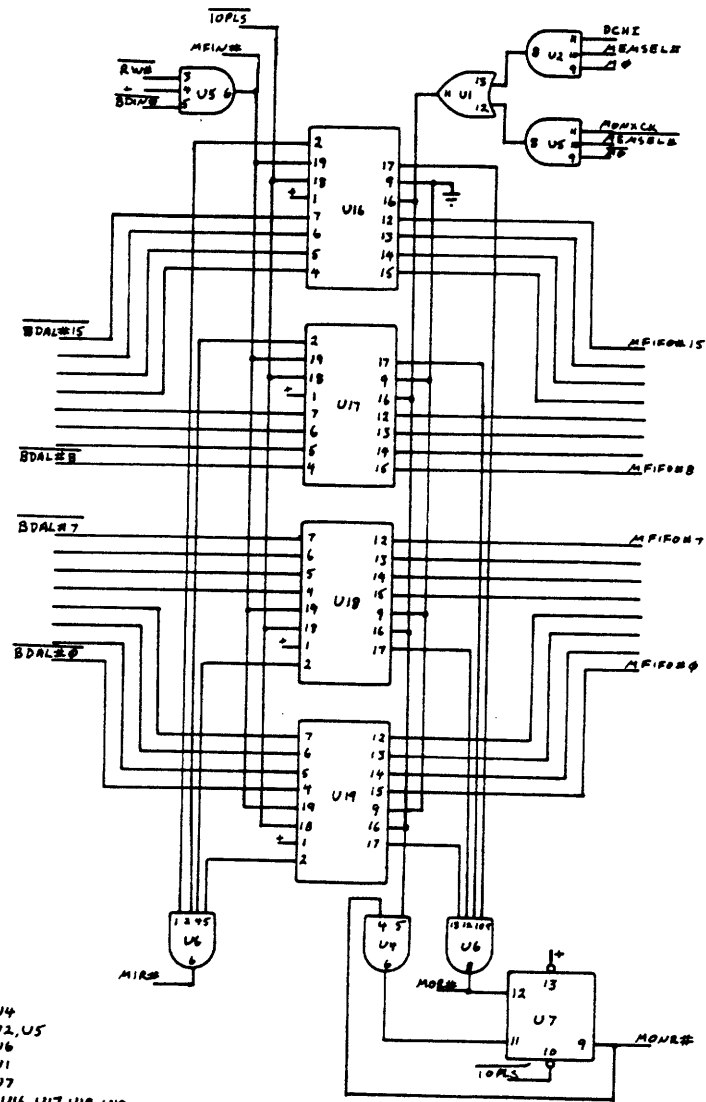
## Appendix D: Control and Buffer Circuit Diagrams

- 1) CFIFO
- 2) MFIFO
- 3) Address Counter
- 4) Memory Timing

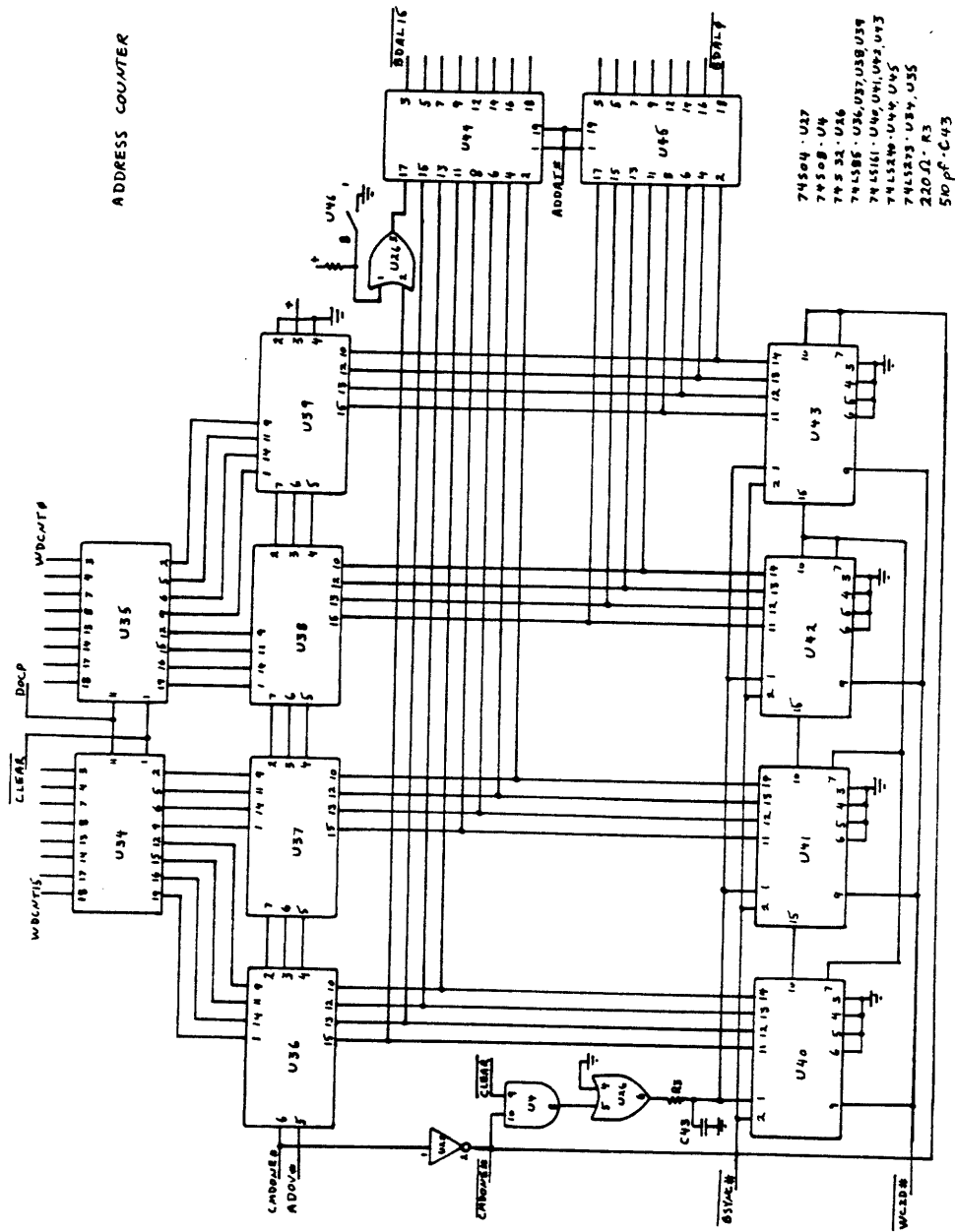


- 74504 - U4
- 74511 - U2
- 74521 - U3
- 74532 - U1
- 74570 - U7
- 745225 - U12, U13, U14, U15
- 745244 - U8, U9, U10, U11

CFIFO: INPUT BUFFER

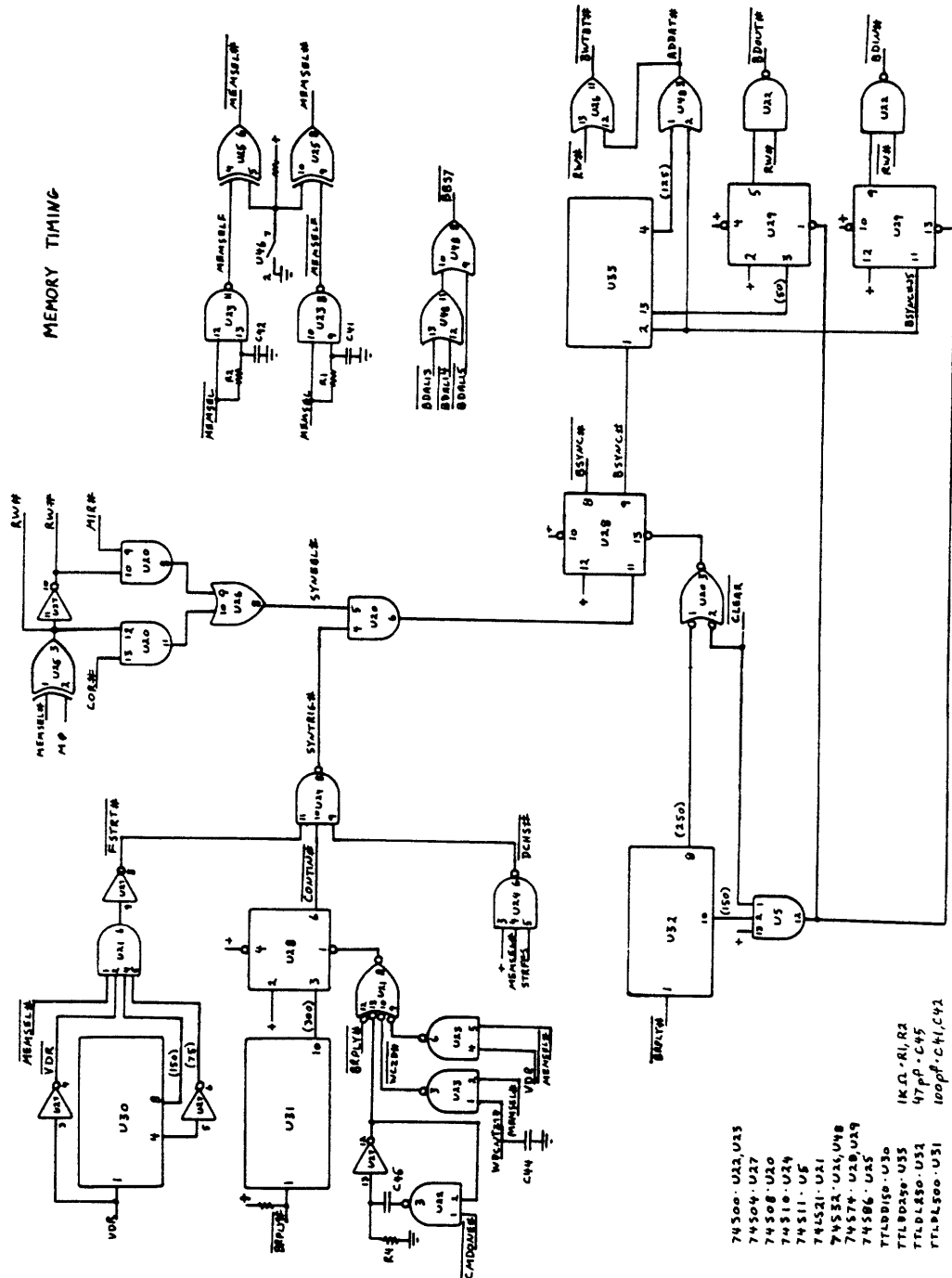


ADDRESS COUNTER



- 74504 - U27
- 74508 - U4
- 74532 - U46
- 74558 - U36, U37, U38, U39
- 74551 - U40, U41, U42, U43
- 74559 - U44, U45
- 74552 - U34, U35
- 220Ω - R3
- 510pF - C43

# MEMORY TIMING



- 74300 - U32, U33
- 74304 - U27
- 74308 - U20
- 74310 - U24
- 74311 - U8
- 74LS21 - U21
- 74LS22 - U25, U28
- 74LS74 - U23, U29
- 74LS86 - U25
- 74LS150 - U30
- 74LS245 - U35
- 74LS180 - U32
- 74LS300 - U31
- 1KΩ - R1, R2
- 97pF - C45
- 100pF - C41, C42



## Appendix E: Interconnections

- 1) Backplane listing
- 2) Cable layout
- 3) Cable listing

Backplane Connector A

PIN	MEM#	CTRL1	CTRL2	MON1	MON2	MON3	A/D/A
A1		MFIF01-0	MFIF02-0	MFIF01-0	CBLNKD-	PIXCLK	PIXCLK
B1		MFIF01-1	MFIF02-1	MFIF01-1	CSYNCD-	HDRC-	HDRC-
C1	BDAL16 +5	MFIF01-2	MFIF02-2	MFIF01-2		DRDY	DRDY
D1	BDAL17 +5	MFIF01-3	MFIF02-3	MFIF01-3	VDRC-	VDRC-	VDRC-
E1		MFIF01-4	MFIF02-4	MFIF01-4	PRES	PRES	
F1		MFIF01-5	MFIF02-5	MFIF01-5	RESET-	RESET-	
H1		MFIF01-6	MFIF02-6	MFIF01-6	MO	MO	
J1	GND	GND	GND	GND	GND	GND	GND
K1		MFIF01-7	MFIF02-7	MFIF01-7	MO-		MO-
L1		MFIF01-8	MFIF02-8	MFIF01-8			
M1	GND	GND	GND	GND	GND	GND	GND
N1		MFIF01-9	MFIF02-9	MFIF01-9			
P1		MFIF01-10	MFIF02-10	MFIF01-10			
R1		MFIF01-11	MFIF02-11	MFIF01-11			
S1		MFIF01-12	MFIF02-12	MFIF01-12			
T1	GND	GND	GND	GND	GND	GND	GND
U1		MFIF01-13	MFIF02-13	MFIF01-13			
V1		MFIF01-14	MFIF02-14	MFIF01-14			
A2	VCC	VCC	VCC	VCC	VCC	VCC	VCC
B2							
C2	GND	GND	GND	GND	GND	GND	GND
D2	+12V						
E2	BDOUT#-	BDOUT1-	BDOUT2-	CLK1	CLK1	CLK1	CBLNKD-
F2	BRPLY#-	BRPLY1-	BRPLY2-	CLK2	CLK2	CLK2	CSYNCD-
H2	BDIN#-	BDIN1-	BDIN2-	CLK4	CLK4	CLK4	
J2	BSYNC#-	BSYNC1-	BSYNC2-				
K2	BWTBT#-	BWTBT1-	BWTBT2-				
L2		CDATA0	CDATA0	MFIF02-13		CDATA0	
M2		CDATA1	CDATA1	MFIF02-14		CDATA1	
N2		CDATA2	CDATA2	MFIF02-15		CDATA2	
P2	BBS7#-	BBS71-	BBS72-				
R2		CAMXCK	CAMXCK			CAMXCK	
S2		MONXCK	MONXCK	MONXCK			
T2							
U2	BDAL#-0	BDAL1-0	BDAL2-0				
V2	BDAL#-1	BDAL1-1	BDAL2-1				

Backplane Connector B

PIN	MEM#	CTRL1	CTRL2	MON1	MON2	MON3	A/D/A
A1		MFIFO1-15	MFIFO2-15	MFIFO1-15			
B1		CDATA3	CDATA3	MFIFO2-0		CDATA3	
C1	ADR18 GND	CDATA4	CDATA4	MFIFO2-1		CDATA4	
D1	ADR19 GND	CDATA5	CDATA5	MFIFO2-2		CDATA5	
E1	ADR20 GND	CDATA6	CDATA6	MFIFO2-3		CDATA6	
F1	ADR21 GND	CDATA7	CDATA7	MFIFO2-4		CDATA7	
H1		CDATA8	CDATA8	MFIFO2-5		CDATA8	
J1	GND	GND	GND	GND	GND	GND	GND
K1		CDATA9	CDATA9	MFIFO2-6		CDATA9	
L1		CDATA10	CDATA10	MFIFO2-7		CDATA10	
M1	GND	GND	GND	GND	GND	GND	GND
N1		CDATA11	CDATA11	MFIFO2-8		CDATA11	
P1		CDATA12	CDATA12	MFIFO2-9		CDATA12	
R1		CDATA13	CDATA13	MFIFO2-10		CDATA13	
S1		CDATA14	CDATA14	MFIFO2-11		CDATA14	
T1	GND	GND	GND	GND	GND	GND	GND
U1		CDATA15	CDATA15	MFIFO2-12		CDATA15	
V1	VCC	VCC	VCC	VCC	VCC	VCC	VCC
A2	VCC	VCC	VCC	VCC	VCC	VCC	VCC
B2					-12V	-12V	
C2	GND	GND	GND	GND	GND	GND	GND
D2	+12V						
E2	BDAL#-2	BDAL1-2	BDAL2-2	MC0		MC0	MC0
F2	BDAL#-3	BDAL1-3	BDAL2-3	MC1		MC1	MC1
H2	BDAL#-4	BDAL1-4	BDAL2-4	MC2		MC2	MC2
J2	BDAL#-5	BDAL1-5	BDAL2-5	MC3		MC3	MC3
K2	BDAL#-6	BDAL1-6	BDAL2-6	MC4		MC4	MC4
L2	BDAL#-7	BDAL1-7	BDAL2-7	MC5		MC5	MC5
M2	BDAL#-8	BDAL1-8	BDAL2-8	MC6		MC6	MC6
N2	BDAL#-9	BDAL1-9	BDAL2-9	MC7		MC7	MC7
P2	BDAL#-10	BDAL1-10	BDAL2-10				
R2	BDAL#-11	BDAL1-11	BDAL2-11				
S2	BDAL#-12	BDAL1-12	BDAL2-12				
T2	BDAL#-13	BDAL1-13	BDAL2-13				
U2	BDAL#-14	BDAL1-14	BDAL2-14				
V2	BDAL#-15	BDAL1-15	BDAL2-15				



SGF Connector 1

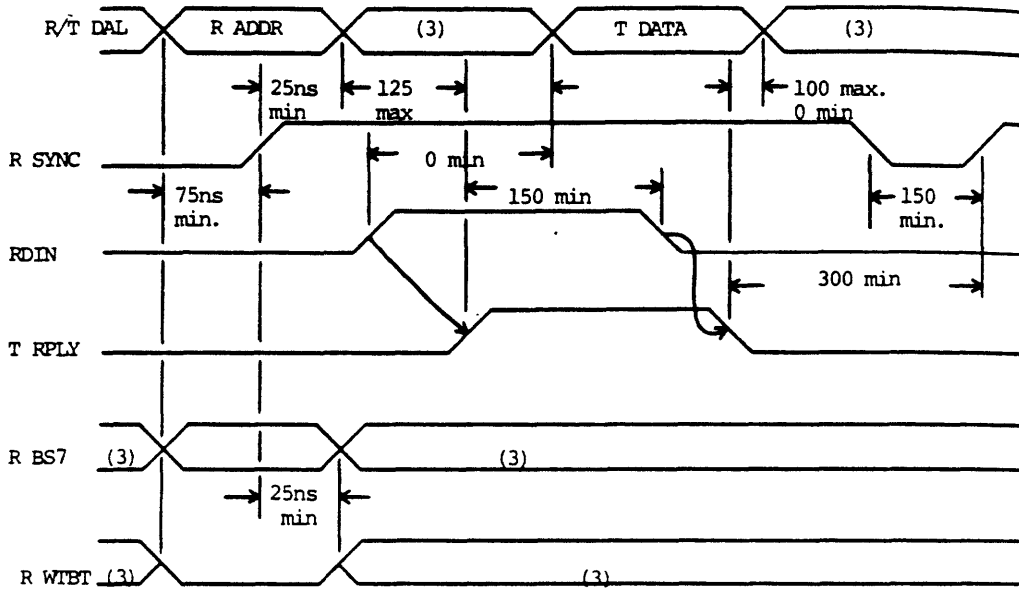
NAME	CTRL1	DMA	NAME	CTRL1	DMA
GND	J1-1	C1-1	GND	J2-1	
WDCNT-14	J1-2	C1-2	ODRC-14	J2-2	C1-14
WDCNT-12	J1-3	C1-3	ODRC-12	J2-3	C1-15
WDCNT-10	J1-4	C1-4	ODRC-10	J2-4	C1-16
WDCNT-8	J1-5	C1-5	ODRC-8	J2-5	C1-17
WDCNT-6	J1-6	C1-6	ODRC-6	J2-6	C1-18
WDCNT-4	J1-7	C1-7	ODRC-4	J2-7	C1-19
WDCNT-2	J1-8	C1-8	ODRC-2	J2-8	C1-20
WDCNT-0	J1-9	C1-9	ODRC-0	J2-9	C1-21
MEMSEL	J1-10	C1-10		J2-10	C1-22
MEMSEL-	J1-11	C1-11	VDR	J2-11	C1-23
MO	J1-12	C1-12	DCHA	J2-12	C1-24
MO-	J1-13	C1-13	STRFFS	J2-13	C1-25
WDCNT-15	J1-14	C1-26	ODRC-15	J2-14	C1-38
WDCNT-13	J1-15	C1-27	ODRC-13	J2-15	C1-39
WDCNT-11	J1-16	C1-28	ODRC-11	J2-16	C1-40
WDCNT-9	J1-17	C1-29	ODRC-9	J2-17	C1-41
WDCNT-7	J1-18	C1-30	ODRC-7	J2-18	C1-42
WDCNT-5	J1-19	C1-31	ODRC-5	J2-19	C1-43
WDCNT-3	J1-20	C1-32	ODRC-3	J2-20	C1-44
WDCNT-1	J1-21	C1-33	ODRC-1	J2-21	C1-45
DCHO	J1-22	C1-34	WDCNTZ1D	J2-22	C1-46
DCHI	J1-23	C1-35		J2-23	C1-47
IOPLS-	J1-24	C1-36	DOCP	J2-24	C1-48
CLEAR-	J1-25	C1-37		J2-25	C1-49
GND	J1-26		GND	J2-26	C1-50

SGF Cable 2

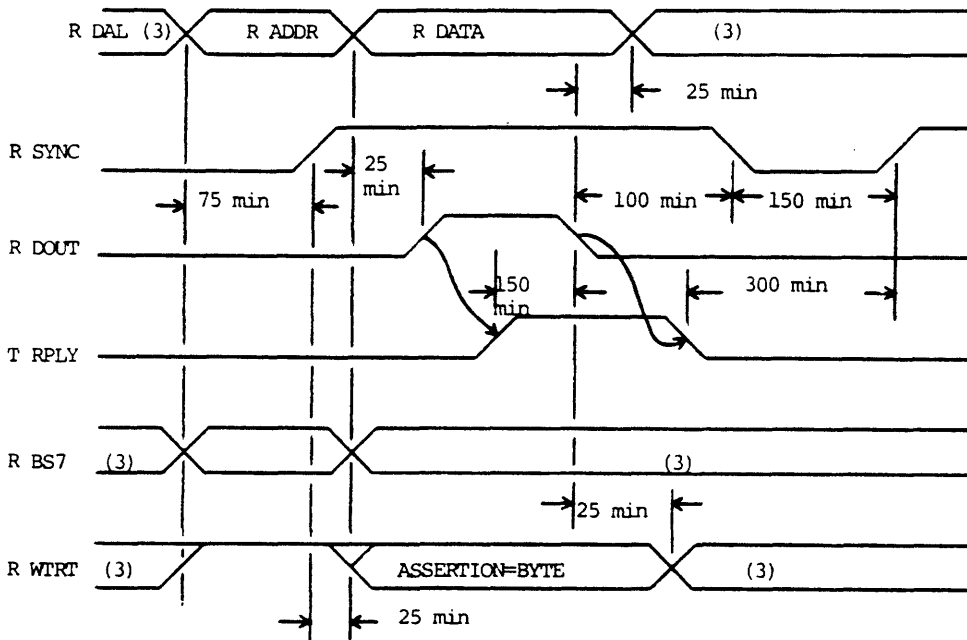
NAME	CTRL1	DMA	NAME	CTRL2	DMA
GND	J51-1	C2-1	GND	J52-1	
MFIFO1-14	J51-2	C2-2	MFIFO2-14	J52-2	C2-14
MFIFO1-12	J51-3	C2-3	MFIFO2-12	J52-3	C2-15
MFIFO1-10	J51-4	C2-4	MFIFO2-10	J52-4	C2-16
MFIFO1-8	J51-5	C2-5	MFIFO2-8	J52-5	C2-17
MFIFO1-6	J51-6	C2-6	MFIFO2-6	J52-6	C2-18
MFIFO1-4	J51-7	C2-7	MFIFO2-4	J52-7	C2-19
MFIFO1-2	J51-8	C2-8	MFIFO2-2	J52-8	C2-20
MFIFO1-0	J51-9	C2-9	MFIFO2-0	J52-9	C2-21
MONR1	J51-10	C2-10	MONR2	J52-10	C2-22
CINR1	J51-11	C2-11	CINR2	J52-11	C2-23
ADOV1	J51-12	C2-12	ADOV2	J52-12	C2-24
CMDONE1	J51-13	C2-13	CMDONE2	J52-13	C2-25
MFIFO1-15	J51-14	C2-26	MFIFO2-15	J52-14	C2-38
MFIFO1-13	J51-15	C2-27	MFIFO2-13	J52-15	C2-39
MFIFO1-11	J51-16	C2-28	MFIFO2-11	J52-16	C2-40
MFIFO1-9	J51-17	C2-29	MFIFO2-9	J52-17	C2-41
MFIFO1-7	J51-18	C2-30	MFIFO2-7	J52-18	C2-42
MFIFO1-5	J51-19	C2-31	MFIFO2-5	J52-19	C2-43
MFIFO1-3	J51-20	C2-32	MFIFO2-3	J52-20	C2-44
MFIFO1-1	J51-21	C2-33	MFIFO2-1	J52-21	C2-45
CIR1	J51-22	C2-34	CIR2	J52-22	C2-46
MOR1	J51-23	C2-35	MOR2	J52-23	C2-47
CONTIN1-	J51-24	C2-36	CONTIN2-	J52-24	C2-48
BSYNC1	J51-25	C2-37	BSYNC2	J52-25	C2-49
GND	J51-26		GND	J52-26	C2-50

## Appendix F: Memory Timing Diagrams

- 1) Memory Specifications
- 2) Memory to Monitor, Actual Operation
- 3) DMA to Memory, Actual Operation



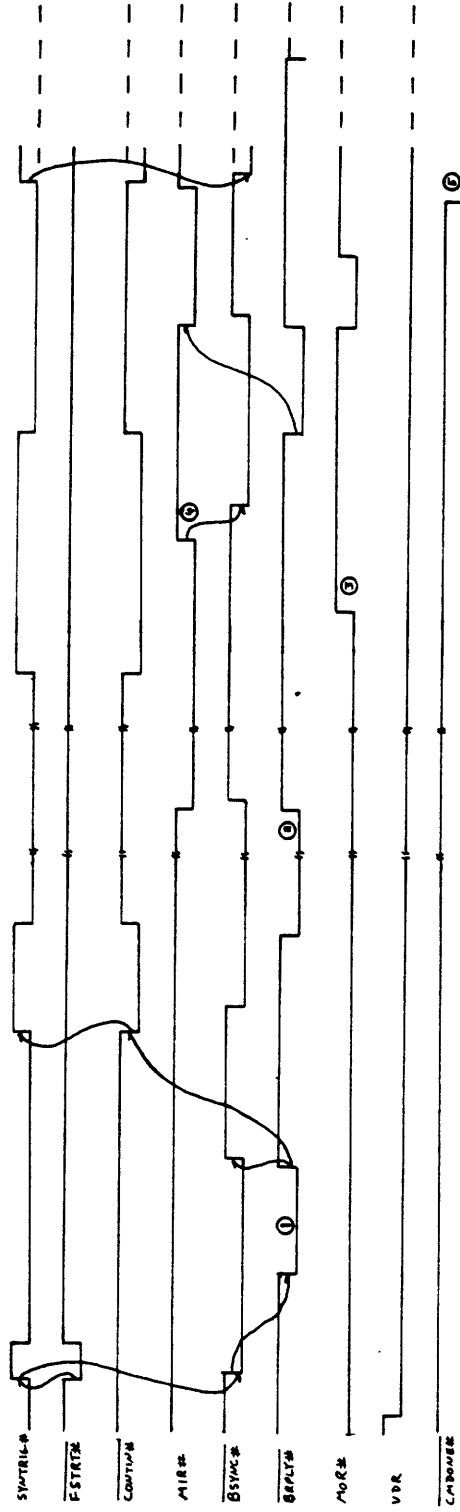
DATI Bus Cycle Timing (Read Memory)



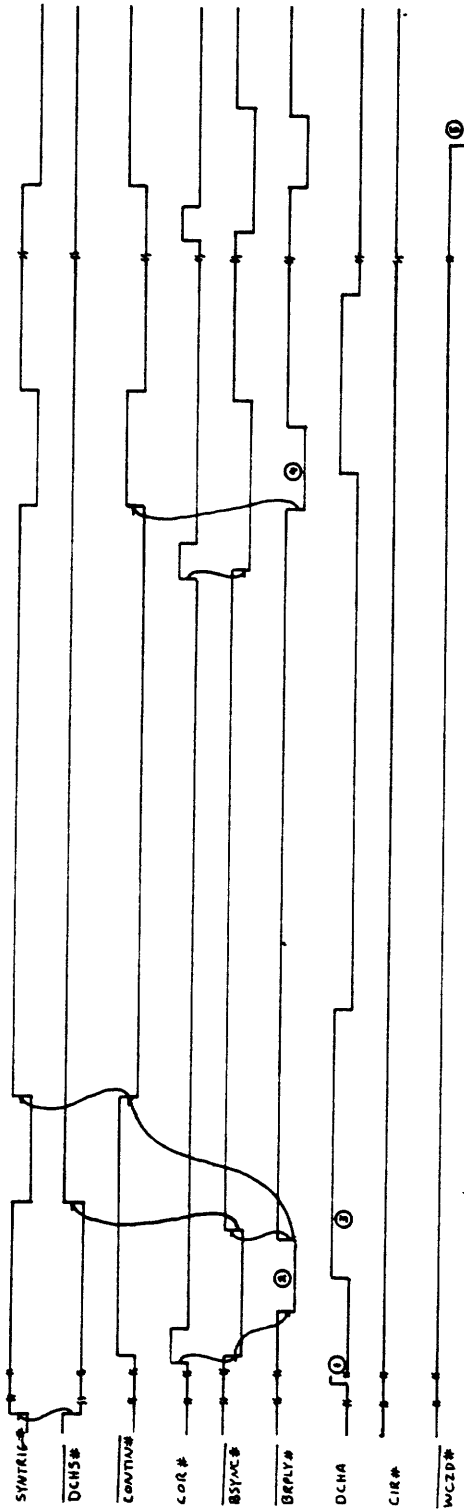
DATO or DATOB



STATE 0:  
MEMORY TO ADAPTOR



STATE 1:  
DMA TO MEMORY



- ① WORD 1 TO FIFO
- ② WORD 1 FROM FIFO
- ③ WORD 2 TO FIFO

- ④ WORD 2 FROM FIFO
- ⑤ END OF FRAME. STOPS CONTING#

## References

1. Pratt, William K. Digital Image Processing. New York: John Wiley & Sons, 1978.
2. Proceedings of the IEEE: Special Issue on Image Processing. Vol. 69, No. 5, May, 1981.