February 1993

Experimental Evaluation of a Video Capture Board for Networked Workstations

Sanjay K. Udani
University of Pennsylvania

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-93-31.

Experimental Evaluation of a Video Capture Board for Networked Workstations

Abstract

This thesis examines the architectural issues in the design of a video capture board intended for use in multimedia videoconferencing. The major issues examined are:

Control of reception and transmission of multimedia video streams,

(i) Quality of service and service provision

(ii) Compression requirements and solutions

(iii) Data buffering and card connection strategies

(iv) Handling multiple video streams

Results of measurements for prototype boards designed and constructed at Penn are also given.

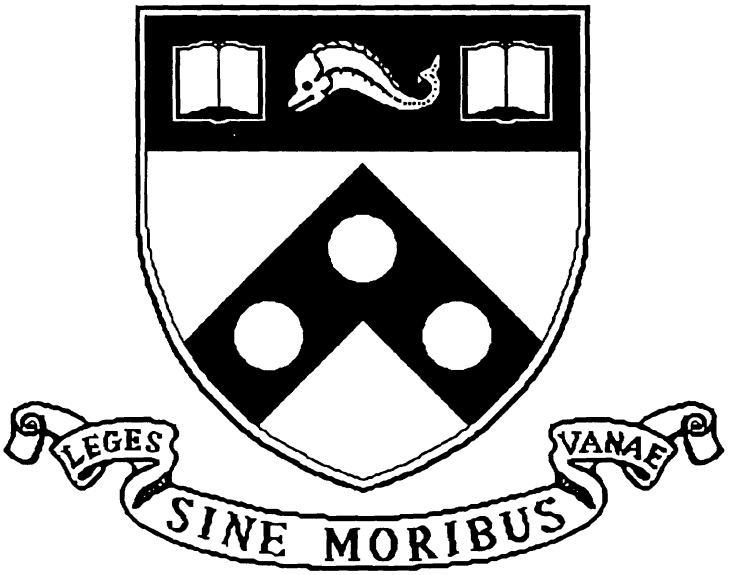
Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-93-31.

Experimental Evaluation Of A Video Capture Board For Networked Workstations

MS-CIS-93-31
DISTRIBUTED SYSTEMS LAB 29

Sanjay Kumar Udani



University of Pennsylvania
School of Engineering and Applied Science
Computer and Information Science Department
Philadelphia, PA 19104-6389

February 1993

UNIVERSITY OF PENNSYLVANIA
THE MOORE SCHOOL OF ELECTRICAL ENGINEERING
SCHOOL OF ENGINEERING AND APPLIED SCIENCE

EXPERIMENTAL EVALUATION OF A VIDEO CAPTURE BOARD FOR NETWORKED WORKSTATIONS

Sanjay Kumar Udani

Philadelphia, Pennsylvania

May 1992

A thesis presented to the Faculty of Engineering and Applied Science of the University of Pennsylvania in partial fulfillment of the requirements for the degree of Master of Science in Engineering for graduate work in Electrical Engineering.

Dr. Jonathan M. Smith
Advisor

Dr. Sohrab Rabii
Graduate Group Chair

Abstract

This thesis examines the architectural issues in the design of a video capture board intended for use in multimedia videoconferencing. The major issues examined are:

- Control of reception and transmission of multimedia video streams
- Quality of service and service provision
- Compression requirements and solutions
- Data buffering and card connection strategies
- Handling multiple video streams

Results of measurements for prototype boards designed and constructed at Penn are also given.

Acknowledgements

I would like thank Jonathan Smith for his support and guidance throughout the project. And Brendan Traw, Alex Doyle and Al Broscius for their invaluable assistance in the design and debugging of the video boards. Thanks also to Supun Samarasekara and Gordon Mcclennen for their help in writing part of the X code.

AURORA is a joint research effort undertaken by Bell Atlantic, Bellcore, IBM Research, MIT, MCI, NYNEX, and Penn. AURORA is sponsored as part of the NSF/DARPA Sponsored Gigabit Testbed Initiative through the Corporation for National Research Initiatives. NSF and DARPA provide funds to the University participants in AURORA. Bellcore is providing support to MIT and Penn through the DAWN project. IBM has supported this effort by providing RISC System/6000 workstations. The Hewlett-Packard Company has supported this effort through donations of laboratory test equipment.

Contents

List of Figures

1 Introduction

Gigabit networks are fast becoming a reality [3]. As computer networks continue to grow in both size and bandwidth, the variety of traffic that they carry has increased from just plain text to a much richer environment, including audio and video [4]. Meetings that once required air travel over thousands of miles can be held in the comfort of each participant's office. Multimedia[1] conferencing presently requires the setting up of direct lines between sites. With the new Broadband Integrated Services Digital Networks (B-ISDN) [6], it will be possible to have multimedia conferencing over a network without media-specific links between the participants. B-ISDN uses the Asynchronous Transfer Mode (ATM) which is a switching technique that uses fixed-size cells to transfer data [8]. It is expected that ATM will become one of the networking standards of the future.

With the ATM standard in mind, we set out to design a video capture board that could be a component of a proposed multimedia conferencing setup. The initial components of our multimedia setup are video, audio, and text. We had several requirements for a video capture board:

1. It should have good cost/performance, and low absolute cost. This allows issues of scale to be studied by distributing the board to several sites.

2. It should have a flexible architecture. In particular, the architecture should be easily adaptable to different hosts. It should also be able to support optional hardware compression.

3. It should connect to an I/O bus, rather than system memory. This offers some degree of portability and allows the board to interact with other peripherals on the host, allowing easier measurement of system behavior. Our choice of an I/O bus was the IBM Micro Channel Architecture which would allow portability across IBM P/S 2 and RISC System/6000 machines.

[1] Multimedia is quite generally a combination of video, audio, text and also other senses that may become transportable in the future, such as the sense of touch.

4. Card-to-card communication, in particular to/from a host's network adaptor, should be possible without host processor intervention.

This paper is organized as follows. First, we look at the commercially available video boards and discuss some of their features. Next, we look at the general architecture of a video capture board. We then discuss in more detail issues such as board location, compression, decompression, buffering, transmission and image resolution. Finally we show the measurements of a prototype video board designed and fabricated at Penn.

2 Related Work

A survey of the market showed that none of the commercially available boards could perform up to our requirements for a general purpose real-time video capture board that could do compression with a reasonable price/performance ratio. In addition, few vendors have considered having multi-party conferencing; most of the boards can only handle two-person conferences. In this section, we discuss details of some of the commercial systems. Some of the approximate prices are also included as a reference. In comparison, our Video Board cost approximately $600 for the prototype.

2.1 Hardware Products

VideoPix, from Sun, [19], [17], has a SBus card designed to capture live video. It accepts PAL/NTSC/S-Video as inputs and displays live grayscale video at a rate of 8 frames per second (180 x 120), but this is without any hardware compression. Color video is displayed at slower speeds. JPEG compression is provided in software as an option under the file save menu, but it takes several seconds to save a frame. VideoPix is designed as a framegrabber, not a live video display card. The bottleneck for the VideoPix card is the Sbus, which has a top end data transfer speed of 25 Mbytes/sec – too slow to send uncompressed color video across without completely overloading the bus.

The RasterOps-based TX/PIP for the DECstation 5000 series [14] does about

12 frames per second of 320x240, 8 bit grayscale in an application. The application opens a video window, gets the data and compresses it using a compression algorithm (Software Motion Pictures) and writes the result over TCP/IP to a remote machine. The performance above is on a DECstation 5000/200 running UWS4.2A. DECstations with a slower TurboChannel will see a somewhat reduced bandwidth out of the frame buffer. The approximate price for RasterOps MediaTime is $1999.

Digital Equipment also has a desktop videoconferencing system called 'DECspin' [16]. It allows users to conference with synchronized video (8-bit grey scale or 24-bit color). It supports conferencing for up to 6 participants. Video and audio packets are transmitted via TCP/IP over ethernet and FDDI WANs (T1 lines are supported). Local window display is 30 frames per second (24-bit color, NTSC, 640x480) and transmitted image speeds are up to 21 frames per second (8-bit grey scale, 264 x 192, compressed, over FDDI with TCP/IP). These frame sizes are a little too small for actual conferencing applications. The session can be recorded to hard disk and callers can leave 20 second messages when their calls go unanswered. Messages can be transmitted to other conference members too. A DECspin system costs approximately $4000-5000.

Silicon Graphics (SGI) has several video options for their workstation line [7]. VideoLab is a 24-bit RGB/YUV/CCIR601 (D1) broadcast quality video window on the workstation screen, with real-time image processing , real-time transcoding (bring in RGB & send out D1, for instance), programmable filters and other features. The approximate cost for VideoLab is $15000. The VideoFramer has a 24-bit NTSC/PAL/RGB/S-Video in/out single frame, broadcast quality animation frame buffer. The VideoFramer costs about $7500.

Parallax has a digitizing board with hardware JPEG compression (using the C-Cube chip). Currently this board is only available for Sparc stations. It currently gives only about 10 frames per second because of inefficiency in using X calls to transfer data.

PictureTel and Intel are collaborating on a chip to do H.261, JPEG, MPEG, and some other PictureTel proprietary encoding techniques. Further information on In-

tel's Digital Video Interactive (DVI) system can be obtained by anonymous ftp from debra.dgbt.doc.ca (192.16.212.15) in the /pub/dvi directory.

2.2 Software Products

DEC provides a software compression option with its 'Software Motion Pictures' which comes with XMedia. It can decompress a very small image in real-time (15 fps) on a DEC DS5000/240, but is highly asymmetric – compression takes about 60 times as long as decompression.

Berkeley has a software implementation of the Quad Trees compression algorithm that can compress/decompress 340 x 240 images at 15 fps on a Sparc2. This seems to be the limit for software compression techniques.

QuickTime for the Macintosh does not do video compression. It simply provides a software environment in which files of compressed video may be displayed in real-time and edited. To compress a video source, you have to use the VideoSpigot software which is capable of compressing between six and seven frames per second. The compression used is called Motion-JPEG, a combination JPEG and MPEG.

In conclusion, most of the boards provided only limited access to the video - it would be difficult to produce the type of data necessary for the multimedia videoconferencing applications that we envision. The prices for these boards are also fairly high, thus reducing their potential for usage.

2.3 Others

On the academic side, MIT is also working on a videoconferencing system. Their approach is slightly different in that the system is a ATM cell-generator. The input is from a video camera and the output from the system is in the form of ATM cells. A bus architecture is not used, and a DSP controls the main circuit. We discuss some of the pros and cons of this method on page 12. This approach fits well with the Desk Area Network [9] from

Cambridge. The Desk Area Network proposes using an ATM switch to interconnect components of a workstation so that all attachments (memory, hard drive etc.) communicate using ATM.

Pandora is a joint project between Olivetti Research Cambridge and the University of Cambridge Computer Laboratory [10]. This project digitizes video as with other boards, but the digitized video is displayed on the screen in a different manner. The digital information is converted to analog, and this signal is mixed with the actual video signal coming from the host to the screen. This cuts down on the bus bandwidth at the expense of added complexity.

3 General Architecture

We envision future multimedia conferences to involve both large numbers of participants and rich sensory environments. Many believe that compression is a way to achieve video with limited bandwidth. However, with the high bandwidth networks of the future, compression is actually a technique for enabling more participants and even richer sensory environments. Multiple participants, however, imply that facilities for servicing their output must be available on the workstation they are connected to. If many participants are desired, cost becomes of paramount importance.

A typical multimedia conference might be held between four distributed participants. Each participant will need to transmit a video stream and receive three streams from the others. If there happens to be more than one conference going on simultaneously, the network load will be increased even more and it is quite clear that the bandwidth used by each video stream has to be minimized as far as possible. This is especially true in the case of multimedia conferencing where multiple users will each be sending and receiving streams of various media types - video, audio, text, etc.

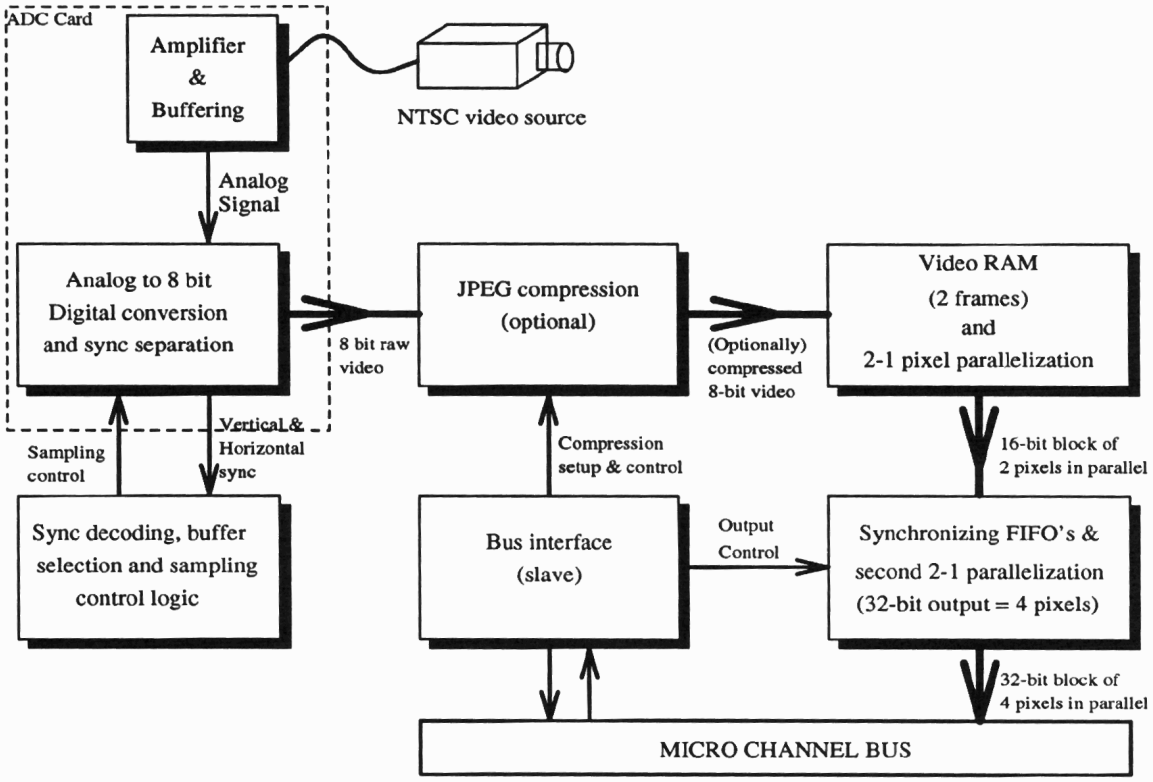


Figure 1: Video Board Block Diagram

3.1 Architecture Blocks

We will look into the issues of sending video over the network in more detail below. Regardless of the architecture used, a video capture board has to have the following basic blocks present (Figure 1):

- An analog-to-digital converter
- Some form of buffering - typically, video RAM
- Logic to control the sampling and synchronization
- A bus or network interface
- Compression

Although the compression block is not required for the video board to function, it is necessary if network bandwidth is a consideration. We look into the layout of the blocks in our Video Board in more detail below.

4 Analog-to-digital Conversion (ADC)

The ADC is the most noise sensitive part of the circuit due to its analog circuitry. In an effort to reduce the possibility of conversion errors, we decided to use a commercially available ADC evaluation board for our design. We have allocated sufficient room on the Video Board to transfer the ADC circuitry on board once the digital end of the board is stable. The ADC board from TRW was modified to suit our purposes. The original board could only take in analog signals and convert them to a 8-bit digital output. After modifications, the board is now able to take in an NTSC video signal[2] and convert it to 8-bit grayscale under the control of the sampling and synchronization logic. In addition, a National LM 1881 sync extractor chip was also added on to the ADC board to enable us to gather the Vertical and Composite Sync signals. These outputs are used to decipher the beginning of each video frame and video line respectively.

A NTSC frame (typically with a 640 x 480 pixel resolution) consists of two interlaced fields, called the odd field and the even field. Each field (with a resolution of 640 x 320 pixels) is updated 30 times a second, resulting in a effective frame update rate of 60 Hz. In our ADC, we have chosen not to use interlacing as our picture size is going to be 320 x 240, which makes it ideal for us to sample just one field for each of our frames. The 320 x 240 size was selected as it offers a good tradeoff between acceptable resolution/picture size and data bandwidth. A simplified diagram of a frame is shown in Figure 2.

[2]A NTSC video output is available from any commercial video camera or VCR

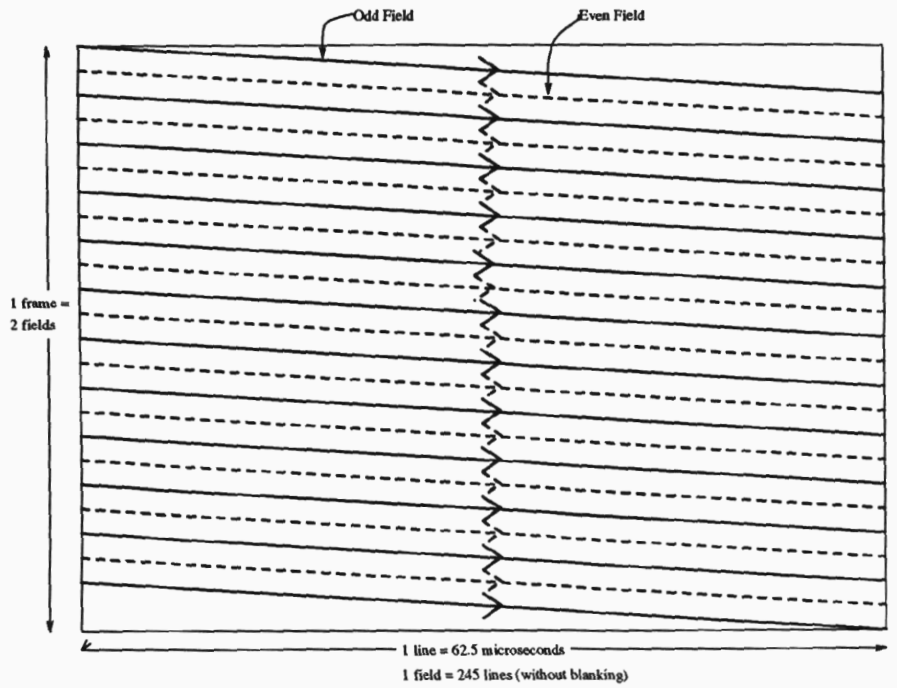


Figure 2: Simplified diagram of an interlaced NTSC frame

5 Sampling and Synchronization Logic

The ADC board needs some control logic to decide when the NTSC signal is sampled. All of the glue logic for the Video board has been fit into one EPLD, the Altera EPM5128 JC. The sampling logic works independently of the bus interface logic, and uses the Vertical Sync and Composite Sync signals from the ADC to decide which part of the NTSC signal to sample. Sampling is done at a rate of 10 MHz (i.e. every 100ns). This rate was chosen as it corresponds to approximately 640 samples per line, and 100ns is also the streaming rate on the IBM R/S 6000. The sampling logic fills the video buffers alternately, and the algorithm used for this is discussed in more detail in the Video Buffering section (Page 9).

In addition to sampling, the PAL is responsible for the flow of data from the video buffers through the FIFO's and Latches to the bus. It also manipulates the clocks of the video buffers and FIFO's to line up four pixels in parallel (Figure 3). The Micro Channel bus is capable of 32-bit transfers, but one pixel consists of only 8-bits. In order to utilize the available data path, we transfer data across the bus in blocks of four pixels in parallel

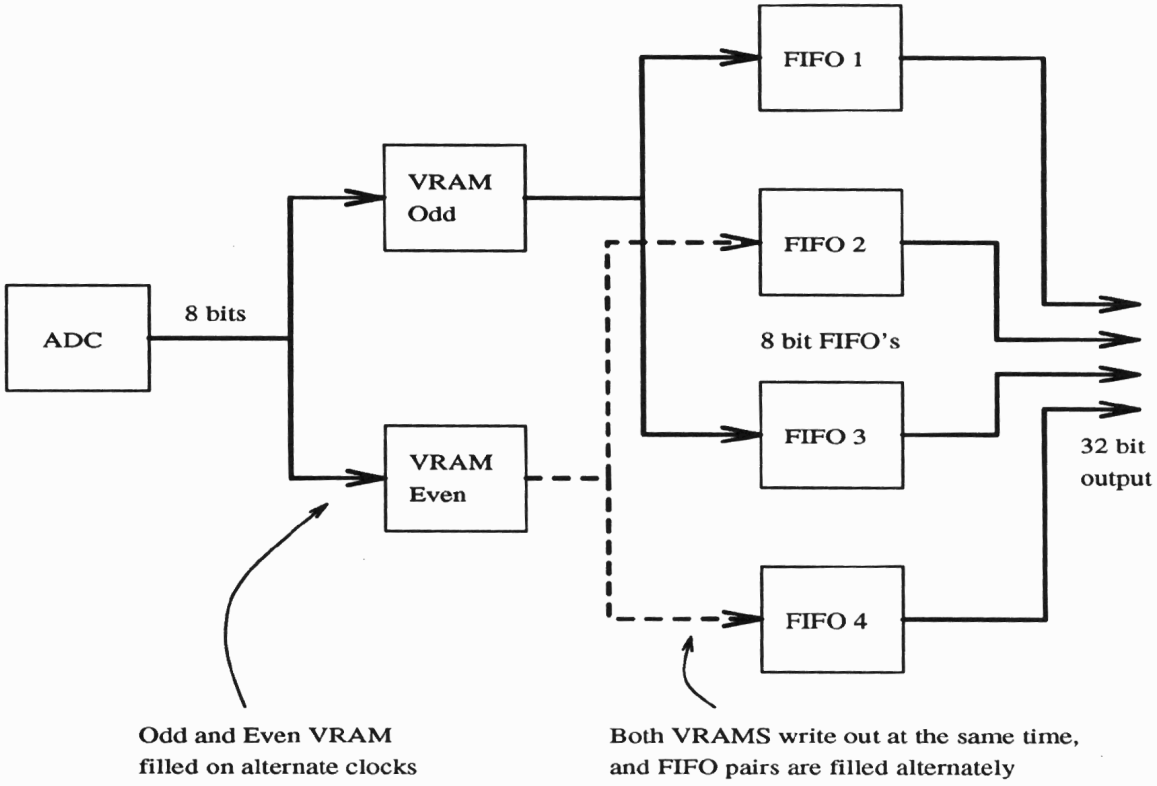


Figure 3: Lining up four pixels in parallel.

(i.e. 32-bits). Although this does not increase the actual overall transfer speed (since sampling and streaming both have 100 ns clocks), it does reduce bus usage by a factor of four. Finally, the PAL also handles the Bus interface controller which is discussed on page 6.

5.1 Video Buffering

Video buffers (VRAMS) are used to store frames before being sent over the bus. The double buffered method is preferred as it allows access to a frame at any time. In most other applications, that is all that would need to be said about buffering. In the case of a real-time video stream however, we need to look at the buffering scheme in more detail.

There are two ways of filling each buffer (Buffer 1 and Buffer 2). The first way is to fill each buffer in sequence, and if one (say Buffer 2) is being read from, then Buffer 1 is updated continuously. A problem arises if Buffer 2 is still being read after Buffer 1

has been filled. Buffer 1 will once again be written to, except that the read for Buffer 2 will have finished before Buffer 1 is filled again. If another read is requested immediately after the previous read (a perfectly reasonable case), Buffer 2 (with its stale data) will be read again, and the vicious cycle continues. We could create a wait state and have the host wait till Buffer 1 is filled, but having wait states for a real-time circuit is generally not good practice. Unless there is a change in the read cycle, Buffer 1 will not be read from, but will always be updated, and Buffer 2 will never be updated but will always be read from. This method is obviously unacceptable.

The method we employ has each buffer being filled in sequence until a read is requested. If (for example) Buffer 2 is being read from and Buffer 1 has just been filled, neither buffer is updated. Only after Buffer 2 has finished its read does a buffer get written to. At first glance this may seem like a bad way doing things, but careful inspection will show that this method actually has several advantages. There is never a case where host has to wait while one of the buffers is being filled. This is critical, especially in multimedia conferencing where synchronization of each of the media streams is so important. Secondly, assuming that reading a frame doesn't take significantly longer than writing one, the buffers will always provide a frame that is, at most, two frames old. With video being sent out at 20-30 frames per second, this delay will not be noticeable.

6 Bus interface

The 68C11 Bus Interface chip is used to decode the control signals from the bus and to transfer data across the bus. There are two modes of operation, programmed I/O and streaming. The programmed I/O mode is used when transferring data from the Video card to the host memory as a slave cannot stream to the host on the Micro Channel bus. This method is slower than the second mode, streaming. When transferring data to the ATM Host Interface (a bus master), the Video card streams data out at a much higher rate. The 68C11 also takes care of the card setup during power up.

7 Image Resolution & Fields

We are currently able to display the video as 640 x 480, 320 x 240, and 160 x 120 pixel frames, although the CIF (Common source Intermediate Format, 360 x 288 pixels) can easily be accommodated. At present, our board takes the NTSC format that is common in the US and Japan.

The resolution of the sampled image is 640 x 240 - i.e. one half of an interlaced frame. We currently display video with 320 x 240 resolution, so every other pixel is skipped from a 640 pixel line to get a 320 pixel line. For the 640 x 480 display, the 640 x 240 frame is enhanced by duplicating each line. The resulting image shows little noticeable loss in detail. This duplication method was used instead of real interlacing as it provides a much faster and simpler method of obtaining images in real-time. In addition, with the line-duplication method, we get an effective rate of 60 'frames' per second from the camera with little noticeable loss in resolution. This increase also allows greater flexibility in image storage and handling as we can now sample a 'full' frame in half the time. The 320 x 240 and 160 x 120 frames were produced by sub-sampling the initial frame. In general, we found the 320 x 240 frame size to have the best speed/resolution ratio.

The addition of color will increase the required bandwidth by a factor of two or three. Currently, our goal is to reduce bandwidth and to produce live video with sufficient detail for normal conferencing applications. Because of that, the current card produces a 256 shade grayscale image, not a color one.

In future versions, a possible compromise would be to have color available on an optional basis. If the there is sufficient bandwidth to spare, color can be switched on. On a loaded network, grayscale video can be transmitted instead. This is an interesting issue that needs to be looked at in more detail.

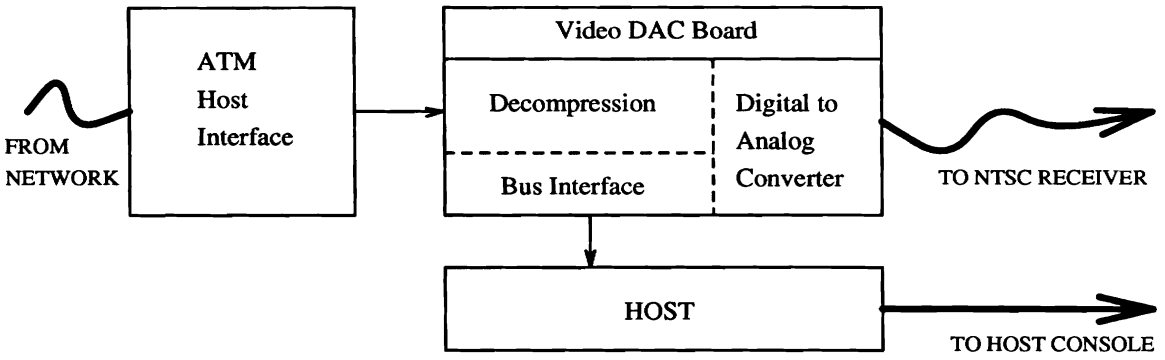


Figure 4: Digital to Analog Conversion

8 Digital to Analog

We are assuming that the video stream from the transmitter is intended for two types of receivers - a host computer (i.e. display on a terminal) and a NTSC compatible viewer (e.g. the Video Wall from Bellcore). The compressed stream has to be uncompressed to produce digitized video which can then be sent to the host for display on a terminal. This digitized video has to be sent to an DAC to produce a NTSC compatible signal. We plan to design a card that will take compressed video from the Host Interface (or equivalent network interface) and produce both uncompressed digitized video and a NTSC signal (Figure 4).

8.1 Host Attachment

One of the questions that comes to mind when looking at the video board is whether to have it as a stand alone box that connects directly to the network through its own ATM Interface — an ATM cell camera. The host bus would not have to handle the load of the video, and a bus interface would not be necessary. There are several reasons why we chose the bus method instead of the stand alone method.

By using a R/S 6000 Micro Channel bus as the base, we are able to have much more flexibility in conducting our experiments. We have been able to create and test a Video Board without any ATM equipment present. For our board to generate ATM cells,

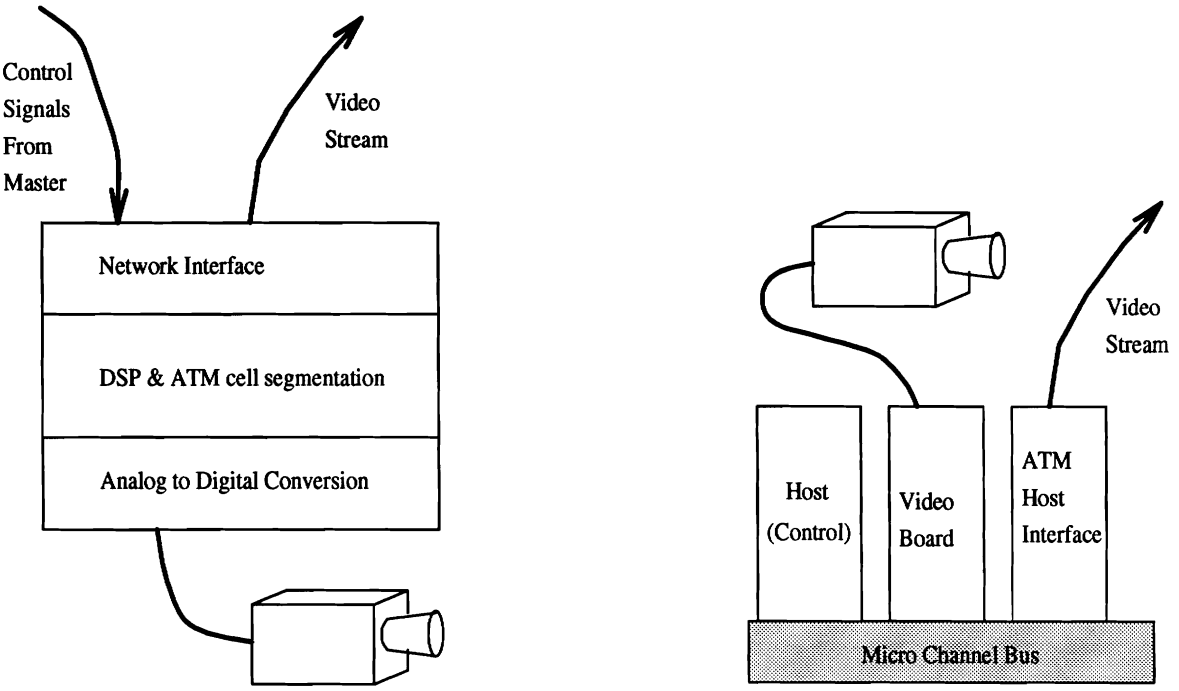


Figure 5: Comparison of our setup and an ATM cell camera.

the video stream is sent directly from the Video Board to the ATM Host Interface [18] with minimal manipulation by the host CPU. The same Host Interface is also able to handle data from other devices. If, at some later point, an ATM cell camera is required, we will simply combine the Video Board with the Host Interface. We believe that combining the two boards is much easier than splitting an ATM camera into two parts, should the need arise.

At a more general level, our configuration can be looked at as a more flexible version of an ATM cell camera, as shown in Figure 5.

9 Compression

There are two schools of thought on video compression. One point of view is that video should be left uncompressed to allow manipulation at any point of the stream. The price

paid for this freedom is the large (a factor of 75 or more [15]) increase in bandwidth. This method is feasible for short distances where unused high bandwidth links are available, but for longer distances, the bandwidth usage is simply not economical. The other perspective is that compression is necessary to reduce network loading. In our model of a multimedia conference, we are assuming that bandwidth considerations will have a higher priority [13] than image manipulation. We also assume that all participants of a conference will not necessarily have standardized equipment - some participants may have a Video Wall display, others may have only their consoles. In the second case, video will have to go through the bus before being displayed. If uncompressed digitized video is the only available source, the bus of that participant will be completely overloaded by the three or more video streams.

We have included a table (Table 1) of the bandwidth taken up by some of the more common video streams. All of the rates are for a 320 x 240, 30 frame-per-second stream. In addition, the the table also shows whether it is feasible (shown with a "$\surd$") to have videoconferencing over the network shown on the horizontal axis. For this, we made it a requirement that the video stream should not take up more than 10% of the network bandwidth so as to allow multiple streams on the same network (without completely freezing up other users !). If this 10% requirement is relaxed, the networks that could carry a two-way stream are also shown (with a "$\approx$").

Table 1 shows fairly clearly that without compression, multi-way videoconferencing of acceptable quality is limited to gigabit networks. While every user in the future may have a 1 Gbps link to his/her terminal, it is unlikely this will occur within the next few years. This strengthens the case for compression. The 30:1 compression mentioned in Table 1 is a conservative rate and is currently achievable in hardware. The 75:1 rate (or higher) should be available in the near future.

If compression is to be used, the question of where it should occur arises. There are three possible locations for compression -

Stream Type	Bandwidth (Mbps)	T1 (1.5 Mbps)	Ethernet (10 Mbps)	T3 (45 Mbps)	Gigabit (620 Mbps)
24 bit color	55.3				√
8 bit grayscale	18.4			≈	√
24 bit color, 30:1 compressed	1.8		≈	√	√
8 bit grayscale, 30:1 compressed	0.6	≈	√	√	√
24 bit color, 75:1 compressed	0.7	≈	√	√	√
8 bit grayscale, 75:1 compressed	0.25	≈	√	√	√

Table 1: Video Stream Rates and Network Bandwidths

In the hardware:

- Immediately after the video has been digitized. In this way, the compression would have to be on board. The advantage of this is in the bus bandwidth that is saved by sending compressed video over the bus instead of uncompressed data. In addition, the number of frames per second can be greatly increased with the reduced size of the frames. The framing can be then done either on the board or by the host. The disadvantage is that any processing of the image (e.g. insertion of text) becomes a problem. Any modifications will have to be done on the video board, not the CPU, as it is not possible to manipulate the compressed data.

- On the ATM Host Interface or other equivalent network interface. The advantage of this method is that video is compressed at the very last moment before transmission, allowing easy manipulation right up to the point of transmission. The disadvantage is that the bus will be saturated by transmission of video from the video card to the Host Interface. This makes it unattractive to use this method except in cases where bus bandwidth is freely available.

In the software:

- In the host memory/CPU. This method offers the greatest flexibility in image manipulation, but has the speed delay inherent in all software. High compression rates at real-time speed are simply not yet feasible at the software level.

Of the three, having compression immediately after digitizing seems the most feasi-

ble method. Any manipulation/addition to the image will have to be done at the receiving end. Sending the manipulation data along with the compressed image and then manipulating the image at the receiving end is possible with proper software management. The tremendous savings on bandwidth - up to a factor of 50 - far outweigh the inconvenience of delayed manipulation.

One argument against compressing video immediately after digitizing is that it makes manipulation of the picture at the source (e.g. zoom, object selection, highlighting etc.) impossible. We look at this issue from a different perspective. Our primary goal is to provide an acceptable quality of real-time video without loading up the bus or the network. In practice, few videoconferencing applications will place image manipulation high on their priority list. A simple stream of video is all that is necessary. Considering that an uncompressed video stream would take up over 30 Mbps compared to the 1 Mbps for a compressed stream, we believe that the trade-off is quite clearly in favor of compression.

9.1 Types of Compression

Compression methods rely on both redundancies in the data and also nonlinearities of human vision. There are three main methods of compression [1]: inter-frame compression, intra-frame compression and lossless compression. The inter-frame method uses compression in space whereas intra-frame compression uses compression in time. In general, both these methods are lossy compression methods. Lossless compression is seldom used for motion video because of its low compression rates (usually no better than 3:1). The main uses of lossless compression are in sensitive applications such as medical imaging where resolution and accuracy are of critical importance.

MPEG [11] is a combination of inter-frame and intra-frame compression techniques, resulting in high compression rates. It is a fairly new standard, and hardware implementations of MPEG are not available as of yet. MPEG is capable of real-time compression rates of over 100:1. This standard is intended primarily for motion video, and

promises a compressed bit rate of under 1.5 Mbps for color video with acceptable quality.

JPEG [21] is an intra-frame coding standard that was originally intended for still pictures. It is now commonly used in both still and motion video compression. The compression rates are lower than that of MPEG, but this compression technique is commonly available in both hardware and software. Single chip implementations of JPEG are available that can perform real-time coding at rates of up to 100:1 [2].

Several other proprietary methods are also being introduced that claim compression rates of 175:1 or more [5]. The video compression field is moving at a very rapid pace and it is impossible to predict which compression technique will become the *de facto* standard.

In selecting the type of compression for our board, we had two main criteria: the compression type should be parameterizable so that the user (or application) can select the desired compression/quality tradeoff and the compression type should be well established - the hardware should be commercially available. We decided on JPEG compression for our board because of the easily available hardware, but MPEG and p*64 [12] compression standards are also strong possibilities once they become more widely used. In addition, our usage of grayscale images aids JPEG, which tends to retain image structure while distorting color values.

9.2 Decompression

Real time video decoding is much easier (i.e. faster) than real time video encoding. In decompressing the video stream, we face the same issues as in compressing — where to place the decompression process. The compressed stream comes in from the Host Interface (or equivalent) card and is sent to the decompression stage. One of the issues at this stage is in the handling of multiple video streams. In the case of transmission, things are simpler - there is only one source of video. For the receiver, there may be many streams coming in simultaneously. Assuming a total of 4 participants, this means there are 3 streams of 30 frames per second coming in to the receiver that need to be uncompressed and displayed.

In effect, we need to handle decompression at a rate of 90 frames per second - or more. At present, we are not aware of any decompression chip that can handle decompression at that rate. There are two alternatives to this problem:

1. To have multiple decompression chips, one for each video stream.
2. Decrease the number of frames per second for each stream to keep the number of frames uncompressed per second constant. As the number of streams increases, the frame rate per stream is decreased to keep the decompression rate constant. This ensures that the decompression circuit is never overloaded.
3. A combination of (1) and (2). With multiple decompression chips, the total decompression rate can be increased, and this rate is kept constant regardless of the number of streams coming in.

While method (1) produces the best results, it is clearly impractical. First, it would be difficult to have a dedicated decompression chip for each channel simply because of the limited space on the bus and card. In addition, it is impractical to fix in hardware the number of streams that a receiver can handle.

Method (2) is the least complicated from the hardware point of view, and is the best approach as long as a decrease in frame throughput is acceptable as the number of streams increases.

We believe that method (3), a hybrid of (1) and (2) is the best approach in terms of hardware and software complexity. In hardware, it is fairly straightforward to start off with one decompression chip and leave room for more chips to be connected in parallel. A multiplexer and some glue logic are the only additional components necessary. An increase in the number of decompression chips is completely transparent to the software. The only difference will be the increase in the number of frames per second processed. A receiver can now choose the quality of service it wants at the hardware level, based on cost or other factors.

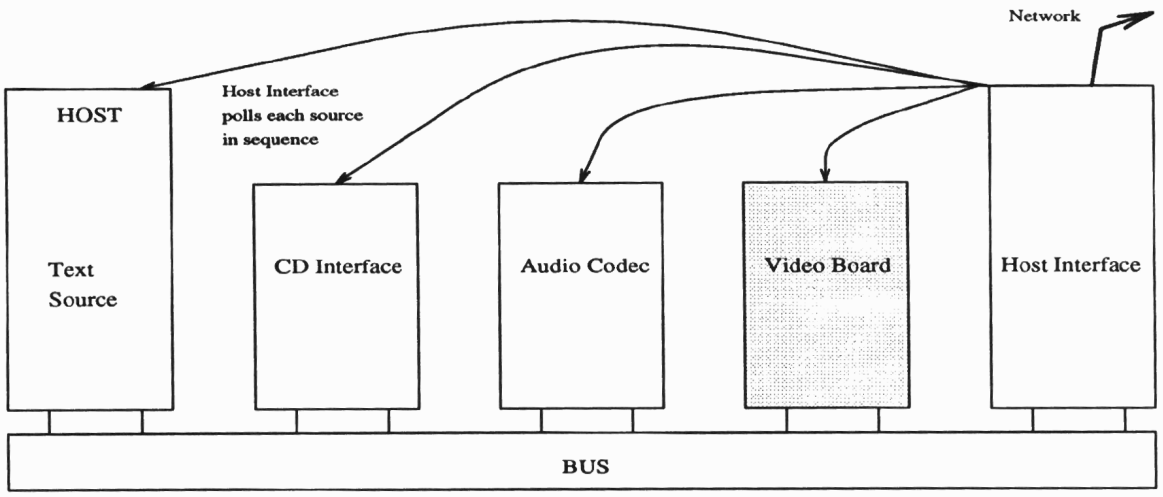


Figure 6: Transmission of media from various sources

10 Transmission

The video stream is meant to be integrated with other multimedia sources such as audio and text. The resulting multimedia stream that is sent out is a *slice* of environment, with each *slice* consisting of a piece of video, a piece of audio and the accompanying text. It is possible to have the host grab each piece of media, transfer it to memory then finally retransmit it over the bus to the Host Interface, but this results in fairly high latency and also a waste of bus bandwidth. We propose to have the Host Interface, under the control of the host, poll each of the media sources and obtain the slices of data from each of the sources and transmit them directly (Figure 6). This reduces bus usage as well as allowing for much lower latency for transmission, which is critical for live streams.

10.1 Quality of Service (QOS)

In practice, a client may not want to pay the premium of receiving the maximum QOS stream. The client may be willing to accept grayscale instead of a full color stream of video in exchange for a lower cost. Or the client may want only 5 frames per second instead of the maximum of 30. There are a multitude of possible combinations of varying QOS, especially in a stream carrying multimedia.

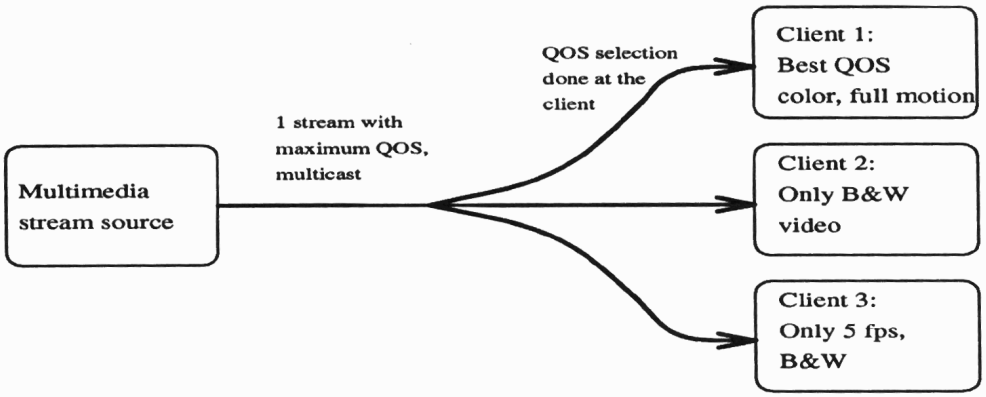


Figure 7: QOS selection at the client

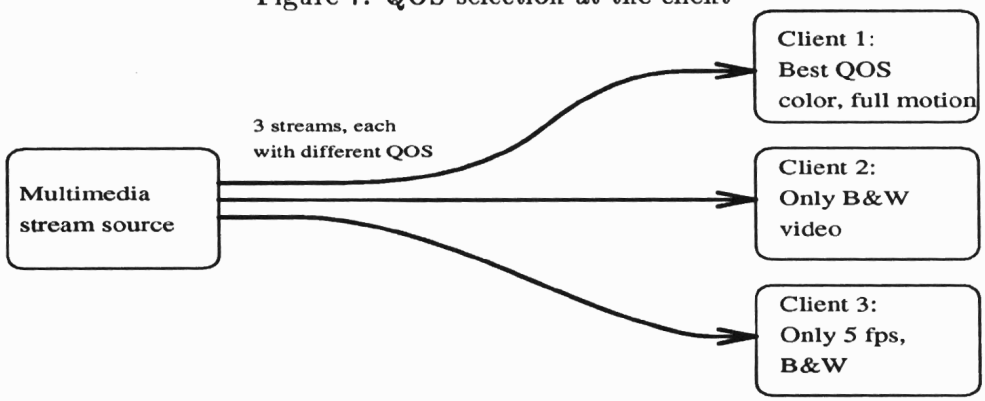


Figure 8: Multiple streams of varying QOS

We perceive two models in handling the issue of quality of service in multimedia conferencing. In the first model, each client (who may also be a source) has the same stream handling capability, at least at the initial interface between the client and the network. The only reason clients may want to use different qualities of service is because of cost considerations. Software is then used to filter out the unused higher-quality portions of the stream. This model allows the source(s) to multicast streams at the maximum QOS, thereby simplifying the transmission process to multiple clients (see Figure 7).

In the second model, each source has the responsibility of sending the correct QOS stream to each of the clients (Figure 8). This implicitly assumes that each of the clients may not be able to even receive the maximum QOS stream from a given source. In practical terms, the cost decision is made at the hardware level. A high QOS client can

receive a lower QOS stream, but not vice versa.

We believe the first model is a better choice as it allows a change in the QOS at the software level. Also, we believe that the cost of the actual interface hardware will not differ by much for different QOS streams. A host interface and video board will still be needed, regardless of the QOS used; only the amount of logic and storage necessary will differ. Furthermore, each source will need additional logic to be able to send different QOS streams.

11 Measurements

The first generation video board has been completed. It does not include on-board compression, and only does byte-wide transfers. We are using a video camera as the NTSC video source and the digitized video stream is displayed on the RS/6000 console using a simple X program. Initially, we modified a still-picture display program (*Xv* by John Bradley of the University of Pennsylvania) to display the video stream. Because of the complexity of the program, we could only manage to display a frame every three seconds.

We then wrote a small X program that resulted in less overhead. A simple X window is created and the stream is displayed by continuously looping the *XPutImage* and *LoadFrameToMemory* functions. We are currently able to display 3.3 frames per second of 8-bit grayscale video with a resolution of 320 x 240 pixels. We are also able to display a slightly slower image on two separate screens simultaneously. A 160 x 120 pixel, 1 bit image can be displayed at almost 10 frames per second. The transactions are performed byte-wide and using discrete I/O calls. At the present moment, we are unable to measure the upper bound of the frame rate as we are software-limited. Ideally, a device driver should be able to increase the number of frames per second up to approximately 15-20 frames for a 8 bit, 320 x 240 picture.

In addition, we are able to transfer byte-wide video data from the Video Capture board to the ATM Host Interface every 400ns. These transfers are done using discrete I/O

transactions. A byte-wide transfer every 400ns is equal to a bandwidth of 20 Mbps, or approximately 15 frames per second. This seemingly low rate scales up quite nicely when we move from byte-wide transfers to 32-bit transfers (increases bandwidth four-fold) and also streaming at a rate of 32-bits every 100ns. The bandwidth is then increased to

$$\frac{32\ bits}{100\ ns} = 320Mbps\ (minus\ arbitration\ and\ congestion)$$

This rate approaches the theoretical bandwidth limit of the Micro Channel bus, and we are confident of reaching close to that rate in direct transfers between the Host Interface and the second Video Capture Board as our measurements are based on actual transfers, not simulations.

The second video board is currently being completed. The second board will include all the features mentioned above, including streaming support and a 32 bit wide data path (i.e. 4 pixels). A picture of the actual board is shown in Figure 9 and the control flow diagram is shown in Figure 10. JPEG compression capability is being included on board, and once implemented will give compression rates of at least 30:1. At that compression rate, a 30 frame per second, 320 x 240 pixel video stream will require a bandwidth of

$$30\ frames\ x\ \frac{1}{30}\ compression\ x\ (320\ x\ 240)\ pixels\ x\ 8\ bits = 0.61Mbps.$$

This is well within the limits of almost any network. If a bigger picture (640 x 480) is desired, the bandwidth doubles to 1.2 Mbps.

12 Conclusion

This thesis examined the architecture of a videoconferencing board for a networked system. We looked at the individual components of digitized video transmission and discussed the issues and problems involved. A survey of commercial and research videoconferencing products was also included. We also looked at the issue of quality of service and the bandwidth requirements associated with it.

We then presented the experimental results of the Penn Video Board. Future designs of the board should include a universal video interface that accepts other standards such as SECAM, PAL and S-Video as inputs. A new chipset from Signetics/Philips [20] has such capabilities. As mentioned earlier, color should be incorporated as a switchable feature.

Figure 9: Picture of the Video Board (Actual size)

References

[1] P.H. Ang, P.A. Ruez, D. Auld, *Video Compression Makes Big Gains*, IEEE Spectrum, pp. 16-19, October 1991.

[2] *C-Cube CL* 550^{TM} *JPEG Image Compression Processor*, C-Cube Microsystems Data Book, August 1991.

[3] D.D. Clark, B.S. Davie, D.J. Farber, I.S. Gopal, B.K. Kadaba, W.D. Sincoskie, J.M. Smith, and D.L. Tennenhouse, *An Overview of the AURORA Gigabit Testbed*, Proceedings of the 1992 IEEE Infocom Conference, Florence, Italy, 1992.

[4] Computer Staff, Gigabit Network Testbeds, IEEE Computer 23(9), pp. 77-80, September 1990.

[5] *DECSpin Digital Sound Picture Information Network Product Overview, Version 1*, Digital Equipment Corporation, January 13, 1992.

[6] D. Delisle, L. Pelamorgues, *B-ISDN and how it works*, IEEE Spectrum, pp. 39-42, August 1991.

[7] G. Estes, Product Manager, Silicon Graphics, Inc., 1992. *Electronic Mail Communication.*

[8] A. Hać, *Synchronous Optical Network and Broadband ISDN Protocols*, IEEE Computer, November 1989.

[9] M. Hayter and D. McAuley, *The Desk Area Network*, Communications of the ACM, pp. 14-21, October 1991.

[10] A. Hopper, *Pandora – An Experimental System for Multimedia Applications*, Olivetti Research Laboratory report, 1990.

[11] D. Le Gall, *MPEG: A Compression Standard for Multimedia Applications*, Communications of the ACM, vol. 34, No. 4, pp. 47-58, April 1991.

[12] M. Liou, *Overview of the p*64 kbit/s Video Coding Standard*, Communications of the ACM, vol. 34, No. 4, pp. 47-58, April 1991.

[13] M.L. Liou, *Visual Telephony as an ISDN Application*, IEEE Communications Magazine, pp. 30-38, February 1990.

[14] B. Neidecker-Lutz, Multimedia Engineering, CEC Karlsruhe, Software Motion Pictures, Digital Equipment Corporation, 1992. *Electronic Mail Communication.*

[15] K. Niwa, T. Araseki, T. Nishitani, *Digital Signal Processing for Video*, IEEE Circuits and Devices Magazine, pp. 27-33, January 1990.

[16] G.V. Siclen, Workstations Multimedia Marketing, Digital Equipment Corporation, 1992. *Electronic Mail Communication.*

[17] M.T. Singer, VideoPix Design Team, Sun Microsystems, 1992. *Electronic Mail Communication.*

[18] C. B. S. Traw and J. M. Smith, *A High-Performance Host Interface for ATM Networks*, Proceedings ACM SIGCOMM '91, Zurich, September 1991.

[19] *Using VideoPix*, Video Frame Capture Card manual from Sun Microsystems, February 1991.

[20] *Video Data Handbook*, Signetics/Philips Semiconductors, 1991.

[21] G.K. Wallace, *The JPEG Still Picture Compression Standard*, Communications of the ACM, vol. 34, No. 4, pp. 31-44, April 1991.

A Glossary of Terms

ADC Analog to Digital Converter.

AIX Advanced Interactive Executive, IBM's operating system on the RISC System 6000 workstation.

ATM Asynchronous Transfer Mode. A possible standard for future networks. Data is transferred in fixed-size cells of 53 bytes (48 bytes of data and 5 byte header)

CCITT Comite Consultatif Internationale de Telegraphique et Telephonique.

Composite Sync A signal extracted from a video source. It signals the beginning of each line of video.

EPLD Erasable Programmable Logic Device.

frame In motion video, a single image (every $\frac{1}{30}$ second). Two fields form a frame in interlaced video.

fps frames per second.

Huffman Coding Static set of minimum redundancy integral-length bit strings.

IOCC I/O Channel Controller on the Micro Channel Bus.

(B)ISDN (Broadband) Integrated Services Digital Network.

JPEG compression Joint Photographic Experts Group compression standard. A compression technique using DCT and Huffman coding to perform intraframe compression. Originally intended for still pictures, but is now also used for motion video. Produces a variable bit rate output.

Mbps Million bits per second.

Micro Channel The bus architecture of the IBM PS/2 and RISC System 6000 workstation.

MPEG compression Moving Picture coding Experts Group compression standard. New standard for motion video, uses interframe coding to produce high compression rates. This standard is still in the settling stages.

NTSC National Television System Committee. TV standard used in the US and Japan.

OC-3 Optical Carrier (155.4 Mbps).

p*64 CCITT adopted video compression standard; *p* (integer)*64 kbps. Also called H.261.

PAL Phase Alternating Line. TV standard used in much of Europe.

PIO Programmed Input/Output. Discrete I/O calls made to access data on the Video Board. Significantly slower than Streaming.

QOS Quality Of Service. Can be defined in many ways - picture resolution, size, frames per second, color and jitter.

Resolution Size and fineness of picture in pixels. Usually given as (width in pixels) x (height in pixels). Common sizes are 640 x 480 and 320 x 240 pixels.

SECAM Sequentiel Couleur avec Memoire. TV standard developed in France.

Streaming Transfer of contiguous words in the address space every 100 ns, without the necessity of passing the address again.

T1 H11 in ISDN. A 1.544 Mbps communication channel

T3 H22 in ISDN. A 45 Mbps communication channel.

Vertical Sync Signal that identifies the start of a new field in a video signal.

VRAM Video Random Access Memory. RAM that has auto-incrementing addressing.

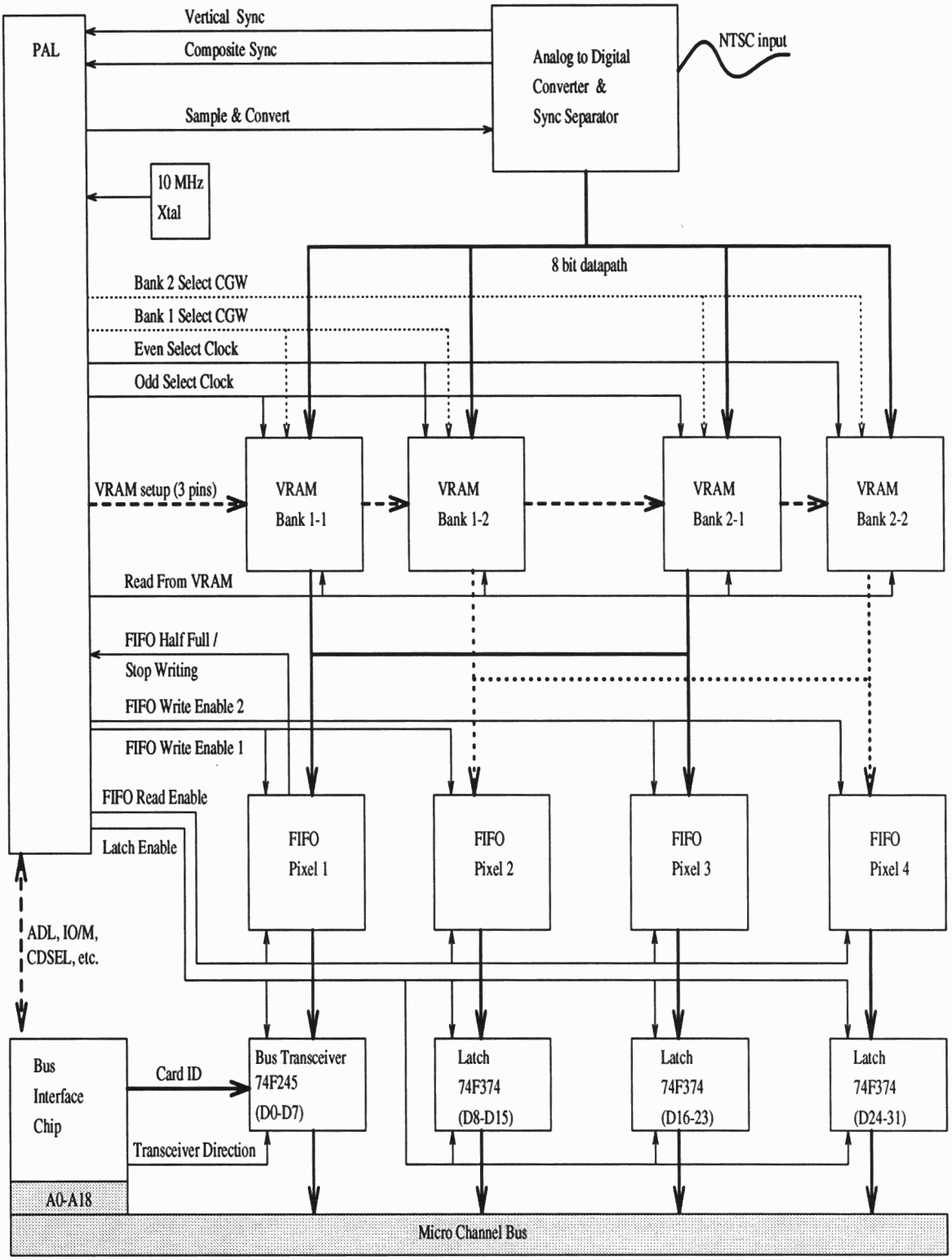


Figure 10: Video Board Data/Control Flow Diagram