The Vidboard: A Video Capture and Processing Peripheral for a Distributed Multimedia System

Joel F. Adam and David L. Tennenhouse

MIT Laboratory for Computer Science

Abstract

This paper describes a stand-alone network-based video capture and processing peripheral (the Vidboard) for a distributed multimedia system centered around a gigabit-per-second Asynchronous Transfer Mode (ATM) network. The Vidboard captures video from an analog NTSC television source and transmits it to devices within the system. Devices control the Vidboard through a set of ATM protocols. Whereas capture boards typically generate video streams having fixed frame rate characteristics, the Vidboard is capable of decoupling video from the real-time constraints of the television world. This allows easier integration of video into the software environment of computer systems. The Vidboard is based on a front-end frame-memory processor architecture that is also capable of generating full-motion video streams having a range of presentation (picture size, color space) and network (traffic, pixel packing) characteristics. A series of experiments are presented in which video is transmitted to a workstation for display. Frame rate performance and a remote video source control model are described.

Keywords: ATM, digital video, distributed systems, framegrabber, gigabit networks, video capture.

Authors' addresses: Joel Adam: MIT Lab for Computer Science, Room 503, 545 Technology Square, Cambridge, MA 02139; (617) 253-4731; jadam@tns.lcs.mit.edu

David Tennenhouse: MIT Lab for Computer Science, Room 508, 545 Technology Square, Cambridge, MA 02139; (617) 253-6005; dlt@tns.lcs.mit.edu

1 Introduction

Asynchronous Transfer Mode [5] (ATM) is a communication paradigm that offers support for seamless broadband communication across heterogeneous networking environments, from wide-area to the desk-area [8]. ATM has many properties which make it well suited to the transport of real-time (audio and video) information [1]. In turn, many research groups are designing distributed multimedia systems centered around ATM networks [12, 6, 4].

With the use of ATM, a new class of devices is possible in which devices act as shared network resources and communicate with each other through ATM-based protocols. In terms of multimedia, examples of such devices are video and audio capture boards, frame stores and video servers.

This paper describes an ATM network-based video capture and processing peripheral device called the Vidboard. The Vidboard was designed in the context of the ViewStation project [12]. The ViewStation system is an all-digital distributed multimedia system centered around a gigabit-per-second ATM network. The Vidboard was developed as a capture interface for the ViewStation environment. Full-motion video is captured from an analog television source and transmitted to other devices within the system. In terms of research, it was developed as a system level prototype for studying the properties of a network-based video source for a distributed ATM environment. These properties relate to functionality and ATM control mechanisms.

Video capture boards have been proposed/designed for other research-based distributed multimedia systems similar to the ViewStation. In the Pandora system [7], a multimedia box containing a video capture board is loosely coupled to a workstation. Black and white video in a variety of picture sizes is transmitted between boxes over a dedicated high-speed network. In the Desk Area Network Project [6], an ATM network-

based capture board with minimal functionality is described. At the University of Washington [4], a network capture board which generates video streams consisting of NTSC samples has been developed. Workstation-based capture boards have been developed for computing environments consisting of networked IBM RS6000 workstations [3, 13].

In these systems, video information is hidden from the user workstation's processor and network to varying degrees. In many cases, this hiding occurs by separating video from other types of information through the use of dedicated hardware, such as dedicated high-speed networks and display mixing cards. When video is mixed, it is usually transported in a non-pixel representation (eg. compressed) that is not directly usable by the workstation processor and is sent to dedicated hardware for processing. The rationale for hiding video has been performance - to avoid saturating workstation and network resources. One of the primary goals of the ViewStation project is to process video at the application level [12]. This software intensive approach will become feasible in a local environment given the trends in workstation and network performance, and allows for easy migration to higher performance workstations as they appear.

Operating in this software intensive environment places temporal demands on the video streams generated by video sources. Within this environment, video tends to be processed in bursts and processing speed depends on workstation resource availability. A mechanism needs to exist for adapting the frame rate to permit graceful degradation as system resources become scarce. The capture systems described above were designed to produce video streams having constant frame rate characteristics over the course of a video session. The Vidboard architecture is novel in that it allows the decoupling of video from the real-time/synchronous constraints of the television world. Through a closed-loop control mechanism, a destination device can dynamically vary the frame rate during the course of a video session. The Vidboard architecture also allows the generation of full-motion video streams having various presentation (picture size, color space, etc.) and network transport characteristics.

The remainder of this paper describes the design and use of the Vidboard. Section 2 describes the ViewStation system and discusses the major functional objectives which motivated the design of the board. Section 3 describes the hardware and software components of the Vidboard. Section 4 describes a set of protocols for transmitting video from the Vidboard to a workstation. Section 5 discusses two systems experiments and presents performance results. Section 6 draws a number of conclusions from the Vidboard research and describes future work.

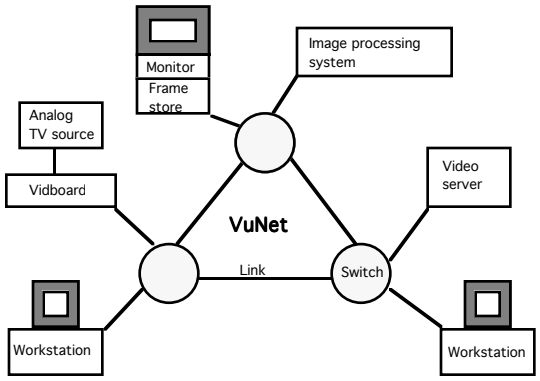

Figure 1: A typical ViewStation system

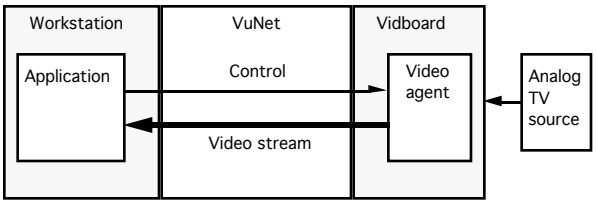

Figure 2: Application-friendly video source model

2 Background

2.1 The ViewStation system

As illustrated in Figure 1, a typical ViewStation system interconnects general purpose workstations and specialized video resources, such as Vidboards, image processing systems, frame stores and video servers. Communication within the system is supported by a gigabit-per-second ATM network called the VuNet[1]. The VuNet consists of switches interconnected by fiber links. Clients connect to the network through a switch port. See [12] for a more detailed description.

2.2 Objectives

As described previously, the software-intensive nature of the ViewStation environment places constraints on video sources. An application must have a mechanism for adapting the frame rate to system resources. The design of the Vidboard was motivated by the application-friendly video source model depicted in Figure 2. In this model, the application sends the Vidboard control information, which is processed by an intelligent *video agent*, and the Vidboard returns a video stream. Through

[1]Within the ViewStation system, the ATM protocol is modified by adding 3 bytes to the end of the header so that the overall length of a cell is 56 bytes, a length that is 4 byte aligned. The additional 3 bytes remain unused.

this closed-loop technique, the workstation regulates the characteristics, and in particular the frame rate, of the video stream.

This model led to the following design objectives for the Vidboard:

Temporal decoupling: The ability to process video at a rate which is disjoint from that imposed by television video. The Vidboard serves as an interface between the real-time world of television and the virtual-time ViewStation computing environment.

Uncompressed video: The Vidboard generates digital video in raw form. The reasons for using uncompressed video are twofold: network bandwidth is not *expensive* in the ViewStation system and video data is handled at the application level. The current generation of workstations does not have the processing power to handle the high data rates of uncompressed video, limiting the frame rate of video streams. However, the ViewStation environment can be easily ported to higher performance workstations as they appear, alleviating this problem. This argument does not mean that compression does not play a role in the handling of digital video. It simply states that compression is not appropriate for shipping video from the Vidboard to an application running on a workstation. Areas where we have found compression to be appropriate are storage and communication across low bandwidth networks attached to the ViewStation.

Distributed control: The ability to receive and execute commands from other devices within the system.

Network transport options: The ability to tailor the video stream to the needs of the destination device on a network transport level. Two factors are involved: the packing of video data into cells (transport protocol) and the generation of video streams having different traffic characteristics.

Another set of objectives relates to digital video applications:

Full-motion video: The ability to generate full-motion (30 frames/s) video streams. Full-motion video streams are important from the point of view of applications.

Presentation options: One level upon which video streams can be tailored concerns the presentation characteristics of video, which determine how the video is displayed. Presentation options include picture size and color space. Devices will have different needs in terms of presentation options for application and bandwidth reasons.

A third set of objectives relates to the network technology adopted for the ViewStation system:

ATM communication: Physical connection to the VuNet switch and adherence to the ATM protocol.

3 The Vidboard

The Vidboard is based on a front-end frame-memory processor architecture as shown in Figure 3. The architecture is centered around a Texas Instruments TMS320C30 digital signal processor (DSP). In turn, the Vidboard is actually a system having both hardware and software components. Video is captured by the Front End and the resultant pixel information is stored in the Frame Memory. The organization of the pixel information within the Frame Memory is controlled by the Format Convert circuitry. The pixels are then read by the DSP, packed into ATM cells and sent to the network. The DSP is also responsible for receiving and interpreting commands from other devices within the ViewStation system. On a research level, the DSP is important because, through software, it provides the flexibility needed to implement different sets of functionality.

In this section, the hardware and software components of the Vidboard are described. The rationale behind the design of the components is summarized.

3.1 Hardware architecture

The Front End executes the actual video capture operation on the board. It consists of a digital video chipset from Philips. The chipset supports a wide range of features for generating digital video streams with different characteristics. The chipset captures one of three NTSC television signals[2] to a resolution of 640x480 pixels. The pixels are quantized to 24 bits in either the RGB or YUV (luminance-chrominance) color space.

During transmission, the packing of digital video components into a data stream is an important issue. The packing format determines the ease with which a receiving device can use the video data. The Vidboard

[2]The chipset is also capable of capturing PAL/SECAM video, however, the board is designed to capture only NTSC video efficiently.

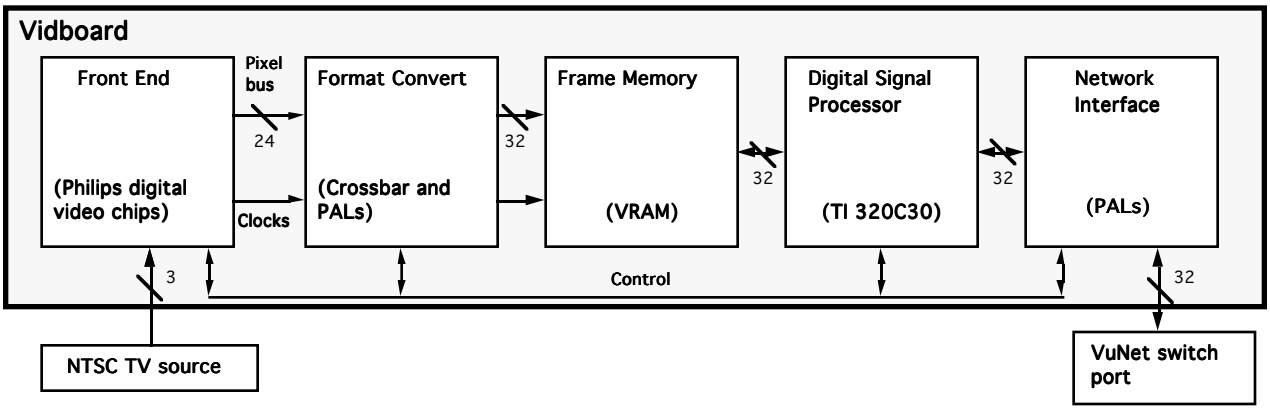

Figure 3: Block diagram of the Vidboard

provides two methods for implementing different packing formats. The first method consists of modifying the format in the DSP as the pixel stream passes through it. This method has the critical disadvantage of preventing real-time performance since format conversion, when executed over an entire video frame, represents a computation intensive task. Therefore, the Vidboard provides real-time support for different packing formats in hardware. The Format Convert circuitry acts as a pipeline stage between the Front End and the Frame Memory. It takes the three component streams produced by the Front End and organizes their writing into the 32-bit word Frame Memory. The manner in which the components are stored in the memory determines how they appear within the DSP word, and ultimately influences the packing format of the transmitted video stream if the pixel words are transferred directly from the Frame Memory to the network. For certain video streams, it is most efficient to use a combination of the two methods.

The Format Convert circuitry consists of a crossbar switch to route the Front End component streams to the proper video RAM (VRAM) chips making up the Frame Memory and a state machine made for control. Two formats are currently supported:

- Component/frame format: Each of the three pixel components is grouped together across the frame and sent in the order of the pixels, one component after another. For example, in RGB space, the R components for the entire frame are sent, followed by the G components and finally the B components. This is the preferred format for storage of RGB video and, in YUV space, the display of black and white video.
- Component/pixel format: The components that belong to each pixel are grouped together and transmitted in the order of the pixels. For example; for a given pixel, the R component is sent first, followed by the G and B components. Then, the next pixel is sent in the same manner and this process is repeated over the whole frame's worth of data. This is the preferred format for color display on 24-bit frame buffers.

The Frame Memory consists of 3 MBytes of VRAM logically organized into two banks, each bank capable of storing one video frame. The two memory banks are used to buffer the video so that one frame can be transmitted while the other is being captured in real-time. Digital video data is written into the memory through the VRAM serial access memory port on a scan line basis and read by the DSP through the conventional DRAM port. The DSP has random access to the video data which facilitates many of the operations involved in video processing such as spatial subsampling.

The Frame Memory serves as a mechanism for decoupling video from the real-time requirements of television. Since the capture of a video frame into the Frame Memory is controlled by the DSP and the Frame Memory is double-buffered, the Frame Memory can be used as a rate adapter between the television rate of the Front End digital video stream and the rate at which the DSP processes a frame of video.

The Vidboard is centered around the DSP[3] which acts as a pixel engine, a command dispatcher and the board controller. During video transmission, the DSP acts as a pixel engine. Digital video data is read from the Frame Memory, possibly processed in some way, packaged into ATM cells and written to the network. The DSP's dual bus structure allows the video data to *flow* through the processor as it is moved from the memory to the network.

[3]The C30 is a 32-bit floating point, 16.6 MIPs digital signal processor that has two independent parallel buses.

The Vidboard design was fabricated into a printed-circuit board in January 1992. The board consists of four signal layers and its dimensions are 13.85in by 9.15in. Cost of a complete board is $1400 in prototype quantities.

3.2 Software framework

A Vidboard will typically contain a software library of the tasks it supports. Devices within the ViewStation system make the Vidboard execute tasks by sending commands over the network. The DSP runs a simple dispatcher for receiving and interpreting these commands. The dispatcher relies on three mechanisms for supporting distributed control: command cells, echo back cells and command tables.

A command cell is an ATM cell generated by the controlling device and sent to the Vidboard which contains information about the task the board is to execute. Table 1 describes the general format of a command cell. Examples of command cells which have been created for remote debugging are READ and WRITE cells for respectively reading and writing the Vidboard address space.

field	# of bytes
ATM cell header	5
unused portion of header	3
echo VCI/tport	4
command code	4
command parameters	40

Table 1: General command cell format

The echo back mechanism is used to prove to a controlling device that the command it sent was received correctly. When a command is received, a copy of the command cell (echo back cell) is sent back to the controlling device, where a check is performed. The echo VCI/tport field of the command cell is used to generate the echo cell header for routing.

The dispatcher determines which command to execute through the command code field. A number, or command code, is associated with each task the Vidboard supports. When interpreting a command cell, the dispatcher uses the command table to map from the code to the corresponding Vidboard routine.

Besides those for video capture, processing and transmission, library routines have been developed for remote debugging and network diagnostics. Time-critical routines, such as those involved in video transmission, are written in assembler language, while non-time-critical routines are written in C.

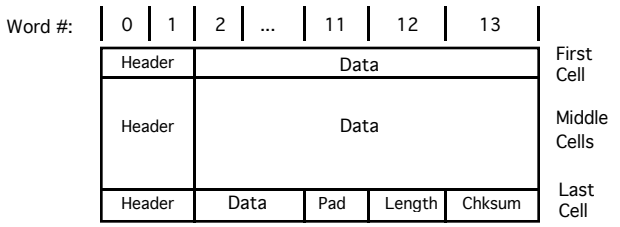

Figure 4: Pseudo-AAL5 transmission frame format

4 Video to the workstation protocols

A set of protocols have been developed for transmitting video from the Vidboard to a workstation. These protocols fall into two categories: a command suite for controlling the Vidboard and network transport/traffic protocols for end-end transport.

Four types of command cells are used to control the Vidboard in a closed-loop control model. They are:

GRAB: The GRAB cell puts the Vidboard into a free-running video capture loop. The next three cells control the functioning of the board within this loop.

TRANSMIT: The TRANSMIT cell causes transmission of the last frame to be captured.

FE_INIT: The FE_INIT cell initializes the front-end chipset and crossbar chip according to the desired presentation options.

FE_OFF: The FE_OFF cell disables the front-end circuitry, terminates the grab loop and returns the Vidboard to an idle state.

In a typical video session, the workstation application places the Vidboard into a free-running capture loop with the GRAB cell and then selects the desired presentation options with the FE_INIT cell. TRANSMIT cells are then sent every time a frame is desired. The selected presentation options can be changed during the course of a session by sending additional FE_INIT cells. An FE_OFF cell is sent to end the session.

To move video data over the VuNet, the Vidboard implements a network transport protocol that is similar to that specified by the ATM Adaptation Layer 5 (AAL5) standard [2]. As shown in Figure 4, cells are

grouped into larger packets called transmission frames (t-frames)[4]. Two levels of segmentation occur during video transmission. Each video frame is segmented into a number of t-frames and each t-frame is further segmented into a number of ATM cells.

word #	field	description
0-1	header	ATM cell header
2	echo VCI/tport	for echo cell
3	command code	grab and transmit
4	dest VCI/tport	for video traffic
5	IBD	interburst delay
6	IFD	interframe delay
7	CB	cells per burst
8	LF	scan lines per t-frame
9	SUB	spatial subsampling
10	COLOR	color space
11-13	not used	

Table 2: TRANSMIT command cell format

To avoid overwhelming a workstation with network data, the Vidboard implements a network traffic protocol. Video data is sent in small bursts since unequal transmission and reception rates can be tolerated over small numbers of cells because the VuNet switch buffers incoming traffic. A burst is followed by a delay during which the workstation drains the buffer of the burst. Video traffic characteristics are defined by four parameters: interburst delay, inter t-frame (interframe) delay, cells per burst and scan lines per t-frame. These parameters are specified in the TRANSMIT command cell, as shown in Table 2. The parameters are used to shape the video traffic to a pattern which can be tolerated by the workstation.

5 Workstation experiments

The mechanisms described in the previous subsection have been used for transmitting video from the Vidboard to an Alpha workstation for display in a small-scale ViewStation system. The system consists of two Vidboards and two Alpha workstations connected to one VuNet switch[5]. The Alphas are interfaced to the switch through a simple programmed-I/O interface board and are equipped with 8-bit frame buffers.

On the Alphas, several routines are used to receive and display the video stream. The *vsdemo* program is an X-windows based video viewer application. It consists of two windows: a view panel for displaying the video and a control panel for selecting the presentation characteristics of the video stream. The *cell reception* device driver reads cells from the switch, unpacks them according to the pseudo-AAL5 transport protocol and stores the contents into memory buffers.

For color video, since the Alpha frame buffers have 8 planes, the Vidboard reduces the 24-bit RGB values to 8 bits. Two techniques for the reduction are explored. The first simply consists of quantizing the pixel value to 8 bits. The second technique implements a dither algorithm to improve the quality of the video picture. This technique uses four sets of quantization look-up tables which are offset to give the viewer the impression of color half-tones. The sets of look-up tables form a two-by-two window which is scanned across the video frame.

This section describes two experiments which focus on frame rate performance and concludes with a discussion of video source control models.

5.1 Vidboard-workstation frame rates

Experiments were conducted to determine the video frame rate which could be achieved between the Vidboard and a Alpha if the latter's only task is the display of video. The frame rates, and corresponding, bit rates for various video streams are listed in Table 3. Certain of the frame rates are limited by the Alpha, while others are limited by the Vidboard. For example, the frame rate of the 640x480 black and white video stream is limited by the rate at which the Alpha can read from the network and display video. The main bottleneck to video traffic between the Vidboard and the Alpha is the simple programmed-I/O VuNet interface[6]. As the table shows, the maximum video bit rate which can be achieved in the context of this system is about 20 Mbits/s. The Vidboard is the limiting factor in the case of the color streams since the quantization and dither algorithms are computation intensive.

5.2 Full-motion experiments

Full-motion frame rates are not achieved when displaying video on the Alpha for a variety of reasons. In terms of the Alpha, the main issue is the slow speed of the I/O bus where the network interface and frame buffer are located. In terms of the Vidboard, the main

[4]The word size in the context of the Vidboard is that of the DSP, 32 bits.

[5]The current implementation of the VuNet switch has four ports, and can therefore connect up to four devices.

[6]A DMA interface is being developed that will increase the network data rate by several factors.

	Video type		
Picture size	Black and white frames/s (Mbits/s)	Color (quantization) frames/s (Mbits/s)	Color (dither) frames/s (Mbits/s)
640x480	9.3 (22.8)	1.8 (4.4)	1.6 (3.9)
320x240	28.0 (17.2)	7.7 (4.7)	6.0 (3.7)
212x160	30.0 (8.2)	16.6 (4.5)	10.0 (2.7)
160x120	30.0 (4.6)	20.0 (3.0)	15.0 (2.3)

Table 3: Vidboard to Alpha frame rate performance

Color space	Picture size	Packing format	Frame rate (frames/s)	Video data rate (Mbits/s)
black and white	640x480	comp/frame	30	74
	320x240	comp/frame	30	18
24-bit RGB	640x480	comp/frame	20	147
	640x480	comp/pixel	20	147
	320x240	comp/pixel	30	55

Table 4: Maximal Vidboard frame rates

issue is the computation intensive nature of the color video algorithms.

A series of experiments were conducted to determine the maximum frame rate the Vidboard could generate assuming that a ViewStation device existed that could receive high-bandwidth video streams. These experiments consisted of having the Vidboard transmit video as fast as possible, with a minimal amount of framing information, to an empty VuNet switch port. Frame rates were determined by counting the number of cycles it took to transmit one frame. The frame rates achieved for various combinations of color space, picture size and pixel packing format are reported in Table 4. The results show that full-motion frame rates are achievable in all cases except for full-resolution color[7].

5.3 Video source control models

As was mentioned previously, a primary objective of the Vidboard is to decouple video from the hard time constraints present in the television world. In the experiments presented in the previous subsections, a closed-loop control was used to adapt the frame rate of the Vidboard to the computational resources of the workstation. From a historical perspective, it is of interest to analyze the control models developed for requesting video.

In our initial experiments, we implemented a connection-based model in which the workstation sent a video service command cell and the Vidboard returned a constant bit-rate video stream. This model had the disadvantage that the workstation was forced to accommodate a constant bit-rate stream no matter what system resources were at its disposal. An example of the type of problem which occurred was the partial loss of frames at the network interface as the workstation devoted resources to tracking a mouse movement or to a disk access.

Since the connection-based model was not well adapted to the ViewStation environment, the closed-loop model, which was our initial design goal, was implemented. The frame rate adaptation mechanism of this model was especially apparent when two vsdemo programs were executed concurrently, each controlling a different Vidboard. The allocation of system resources could be seen to lean towards the servicing of one Vidboard or the other as different frame rates and picture sizes were selected.

6 Conclusion

This paper described the Vidboard, an experimental network-based video capture board for the ViewStation distributed multimedia system. The Vidboard's front-end frame-memory processor architecture allows the temporal decoupling of video from the real-time constraints of the television world as well as the generation of full-motion video streams having variable presentation and network characteristics.

[7] A faster version (by 20%) of the C30 DSP will be produced shortly. This will improve frame rate performance.

A number of important implications, concerning the properties of a network-based video capture module, can be drawn from the Vidboard research. The ability to temporally decouple video from its television source combined with a closed-loop control model facilitates the integration of video into the virtual-time computing world. Video service can be adapted to the available system resources and degrades gracefully as resources become scarce. A second implication is that a capture module needs to be able to accomplish tasks related to the distributed nature of the environment and not only capture/transmit video. Task examples are network transport, network traffic and distributed control.

With the trend towards integration of video into the digital domain, research groups are developing cameras which produce video in a digital format [9, 10] that will come to replace television camera/capture board systems for live video applications. Similarly, with the trend towards distributed computing, video capture systems which connect directly to a network are becoming desirable. A digital network camera would consist of a digital camera with added circuitry for network interfacing, distributed control and frame buffering. It is our hope that a subset of the Vidboard architecture and functionality could serve as a basis for the digital network camera of the future. Important architectural features are hardware support for pixel packing formats and dual frame buffering for rate adaptation.

In addition to the basic tasks of video capture and transmission, the programmability of the Vidboard provides support for other tasks related to digital video and distributed computing. The following tasks have been envisaged or are being implemented:

- The extraction and decoding of closed-caption information from a television signal.
- A daughterboard for color dithering.
- Video compression using the JPEG [14] standard.
- Experimenting with the video header/descriptor formats under development by SMPTE [11].

Five Vidboards are currently in operation. Future plans call for a revision of the prototype board in 1993 and the deployment of approximately twenty boards in a large-scale ViewStation system by 1994.

Acknowledgements

This research was supported by the Advanced Research Projects Agency of the Department of Defense, monitored by the United States Air Force (AFSC, Rome Laboratory) under contract No. F30602-92-C-0019, and by a grant from Nynex.

References

[1] Ambrüster, H. and Wimmer, K. Broadband Multimedia Applications Using ATM Networks: High-performance Computing, High-capacity Storage, and High-speed Communication. *IEEE JSAC 10*, 9 (Dec. 1992), 1382-1396.

[2] ANSI T1S1.5. AAL5 - A New High Speed Data Transfer AAL. Stds Proj. T1S1.5/91-449 AAL-ATM, Nov. 1991.

[3] Chen, M. et al. A Multimedia Desktop Collaboration System. IBM Research Report RC 17951 (#78882), May 1, 1992.

[4] Cox, J., Gaddis, M. and Turner, J. Project Zeus. *IEEE Network 7*, 2 (March 1993), 20-30.

[5] de Prycker, M. *Asynchronous Transfer Mode: Solution for Broadband ISDN*. Ellis Horwood, 1991.

[6] Hayer, M. and McCauley, D. The Desk Area Network. *ACM Operating Systems Review 25*, 4 (Oct. 1991), 14-21.

[7] Hopper, A. Pandora - An Experimental System for Multimedia Applications. *ACM Operating Systems Review 24*, 4 (Apr. 1990), 19-34.

[8] Leslie, I., McAuley, D. and Tennenhouse, D. ATM Everywhere. *IEEE Network 7*, 2 (March 1993), 40-46.

[9] Nikoh, H. and Kuwajima, T. The Full Digital Video Camera System and Simulation of its Essentail Parameters. In *Proceedings of the IEEE International Conference on Consumer Electronics* (1989), 48-49.

[10] Parulski, K.A. et al. A Digital Color CCD Imaging System Using Custom VLSI Circuits. In *Proceedings of the IEEE International Conference on Consumer Electronics* (1989), 182-183.

[11] SMPTE Header/Descriptor Task Force: Final Report. *SMPTE Journal*, June 1992, 411-430.

[12] Tennenhouse, D. Telemedia, Networks and Systems: Group Annual Report. MIT LCS Internal Report, Aug. 1992.

[13] Udani, S.K. Design, Implementation and Experimental Evaluation of a Video Capture Board. Master's thesis, University of Pennsylvania School of Engineering and Applied Science, 1992.

[14] Wallace, G.K. The JPEG Still Picture Compression Standard. *Communications of the ACM 34*, 4 (April 1991), 31-44.