Institut Eurécom
2229, route des Crêtes
B.P. 193
06904 Sophia Antipolis Cédex
FRANCE

Research Report 94-005

Architecture of a Light-Weight Multimedia Adapter

Erich Rütsche

February 1, 1994

Erich Rütsche
E-mail: ruetsche@eurecom.fr
tel: +33 93 00 26 53
fax: +33 93 00 26 27

Abstract

This papers analyzes the particular requirements that multimedia communication imposes on the network adapter and on the I/O subsystem of a workstation. An adapter architecture is developed that is specially suited for multimedia communication. The key ideas are the separation of isochronous and asynchronous traffic and the execution of per-byte protocol operations on the adapter. The adapter does not perform protocol processing but provides only protocol processing support. Applications of the adapter and the performance gain for protocol implementations are discussed.

Architecture of a Light-Weight Multimedia Adapter

Erich Rütsche
Institut Eurécom, Sophia-Antipolis, France
ruetsche@eurecom.fr

1. Introduction

New types of networks such as FDDI and ATM provide a bandwidth in the range of 100 -155 Mb/s to a single workstation. Multimedia communication requires broadband communication of high-speed data, still images at high resolution, and moving pictures (video, animated graphics). The aggregation of these data generates not only high-volume continuous data streams but also dynamic bursts of data. An example of this is cooperative teleworking with video connections between multiple parties which also exchange high-resolution images, e.g., doctors holding a teleconference discussing X-ray images.

The communication of these data imposes a new type of load on the network, on the network adapter, and on the I/O subsystem of the workstation. Processing requirements of multimedia protocols are very different from the requirements of traditional transport protocols. Isochronous multimedia traffic requires the transmission of a large amount of data with low delay and bounded delay jitter but may accept relatively high bit-error or packet loss rates [1]. In a video, for example, bit errors or even lost frames are hardly visible, whereas a delay jitter of more than 10 ms is perceived as disturbing. Asynchronous traffic, for example file transfer or a remote procedure call, requires, in comparison, less throughput but tolerates no errors. In the conference example above, the video connection between the doctors could tolerate errors, whereas the X-ray images must be displayed error-free.

Conventional protocol processing on the workstation is limited by the available processing power and the I/O bottleneck [2]. Communication subsystems based on single processors such as Nectar [3] or based on multiprocessors, such as the Parallel Protocol Engine (PPE) [4] and similar architectures [5][6] were built to off-load protocol processing from the workstation. These systems were successful for transport protocols stacks at a network speed of up to 150 Mb/s and brought valuable insight into problems of parallel protocol processing. However, these approaches are too complex for commercial use and they are not suited for the quality of service requirements of real-time multi-media data streams. A successful multiprocessor subsystem for continuous multimedia at moderate speed was demonstrated in [7].

In this paper the new requirements generated by networked multimedia on the network adapter and the I/O system of the workstation are analyzed. A new light-weight architecture for an adapter specially suited for multimedia communication is developed. Application scenarios and the performance of the light-weight multimedia adapter (LWMA) are discussed.

In the next section we present the architectural concepts of the LWMA. In Section 3 we discuss the architecture and its key components. The performance gain of protocol implementations on the architecture is evaluated in the following section. Section 5 describes application scenarios for these new adapters. Section 6 is the conclusion.

2.2. Per-byte operations on the adapter

Per-byte operations such as copying and checksumming are too slow on a workstation and limit the protocol processing performance. Therefore these operations should be implemented in dedicated devices on the adapter.

3. The architecture

In this section we develop the architecture of a light-weight multimedia adapter for a 155 Mb/s ATM network. The architectural concept of the LWMA is shown Figure 1. The communication adapter is split into similar send and receive sides. Both sides provide separate host interfaces for asynchronous and isochronous traffic. The receive side demultiplexes incoming traffic to the isochronous and the asynchronous interface. On the sender side asynchronous and isochronous traffic is multiplexed on the network. The adapter is light-weight in the sense that it does not provide protocol processing but provides only protocol processing support. The coder/decoder offloads the per-byte operations that are too costly on the workstation.

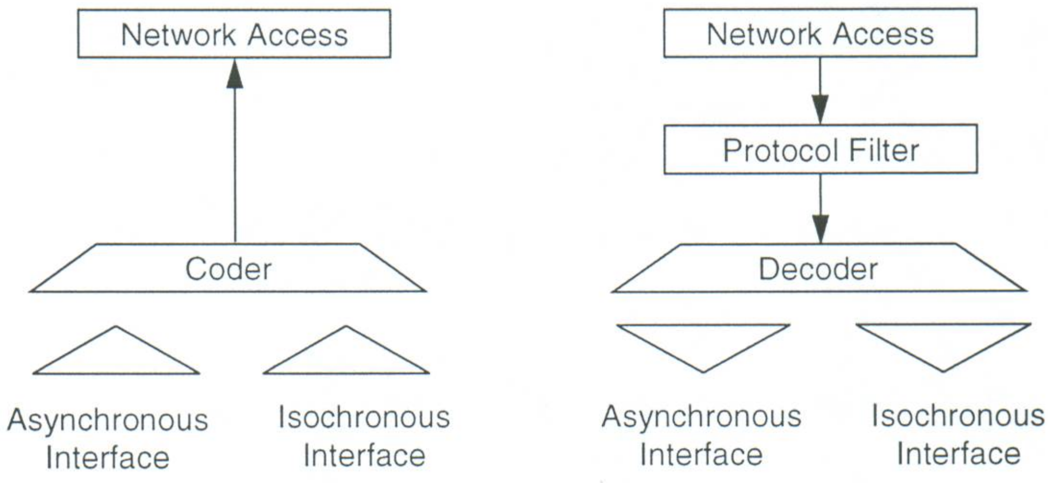

Figure 1. Architecture Overview

On the receiver side, the network access unit implements the medium access control (MAC) functions and provides a frame interface to the next pipeline step. The coder separates asynchronous and isochronous packets arriving on the network. The protocol filter, described in detail below, is the key device that supports this demultiplexing function. It parses protocol headers and generates a unique connection number (CN) for a known connection. The CN determines to which interface the traffic is forwarded. The decoder uses the CN to calculate the appropriate checksumm/CRC[1] required by the given connection. For multimedia data the coder can perform decompression. The decoder then forwards the received packet, its CN, and its check sequence over the isochronous or the asynchronous interface to the workstation.

On the sender side the coder multiplexes the asynchronous and isochronous traffic coming from the separated interfaces to the network access unit. The coder calculates the check

1. In the remainder of this paper we will use the more generic term ***check sequence*** whenever we speak of a checksum as in TCP/IP or a Cyclic Redundancy Check sequence (CRC) as in OSI TP4.

sequence for asynchronous traffic or a compression algorithm for isochronous traffic and finally forwards the data to the frame interface of the network access unit.

The architecture of a LWMA with the ATM network interface is shown in Figure 2. The frame memory of the network access unit is connected to the protocol filter and the DMA unit. The coder/decoder acts also as a controller for the network access unit. For an ATM network, the network access unit would implement the ATM Adaptation Layer (AAL) protocols, and the coder/decoder processes the signalling protocol.

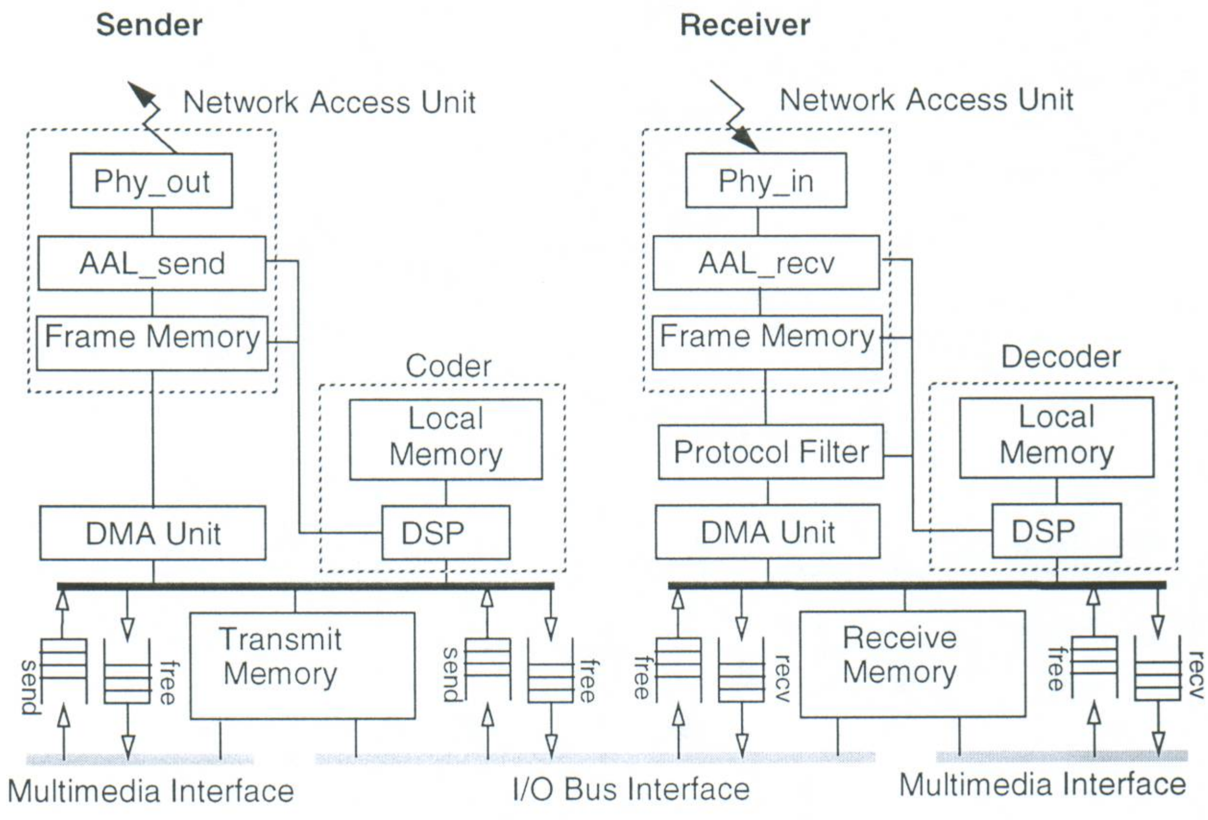

Figure 2. Architecture of the Light-weight Multimedia Adapter

The host interface is split into an asynchronous and an isochronous interface. It looks similar to the Afterburner [14]: the access to the memory is controlled by FIFOs. The isochronous and asynchronous interfaces provide separate FIFOs to control access to the memory of the sender and of the receiver. Thus asynchronous and isochronous data reside in the same memory. The memory must have sufficient bandwidth to provide concurrent access from the interfaces, the DMA Unit and the coder.

To send data an application gets a pointer to a free buffer from the free FIFO. It writes the data to the buffer in the transmit memory and writes the buffer pointer to the send queue. On reception the decoder gets a receive memory buffer from the free queue and writes the pointer to the processed buffer to the receive queue. Each active multimedia connection has its dedicated send and receive FIFO.

3.1. Protocol Filter

The protocol filter (PF) is a central device of the architecture that is needed to separate isochronous and asynchronous traffic. It scans each incoming packet header and extracts the information defining a connection. If a known connection is found, the protocol type and the connection number are returned. The connection number is a unique number that can be used as a pointer to the connection control information.

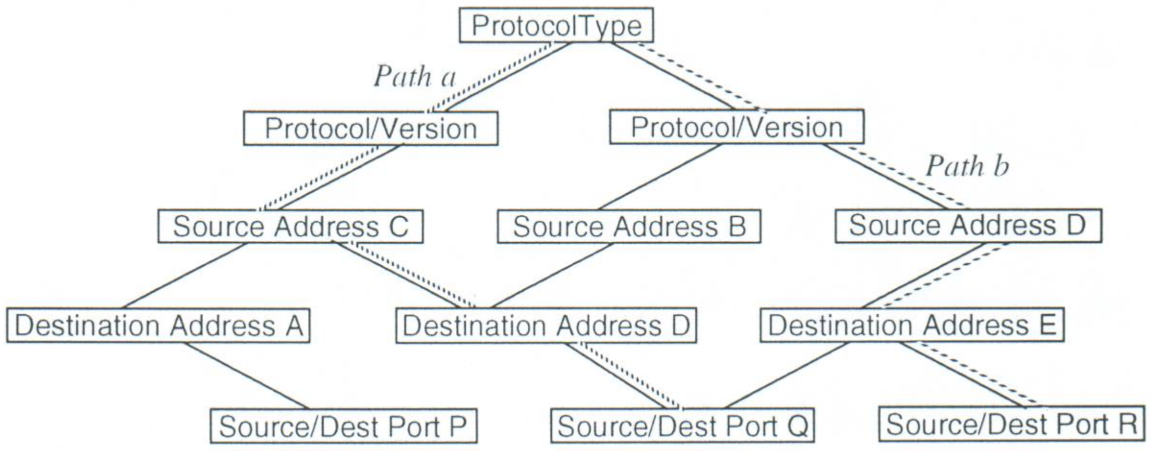

Figure 3. Protocol Address Tree Structure (Internet Protocols)

The addresses and protocol specific information used in the various layers of a protocol stack form a tree. A connection is defined by the path through the tree (see Figure 3). This path is stored in a content addressable memory (CAM). A CAM row contains the information of the tree level, the protocol type and the address information. For each branch of the tree, the CAM returns the address of the matched word. These addresses are concatenated to form a unique connection number.

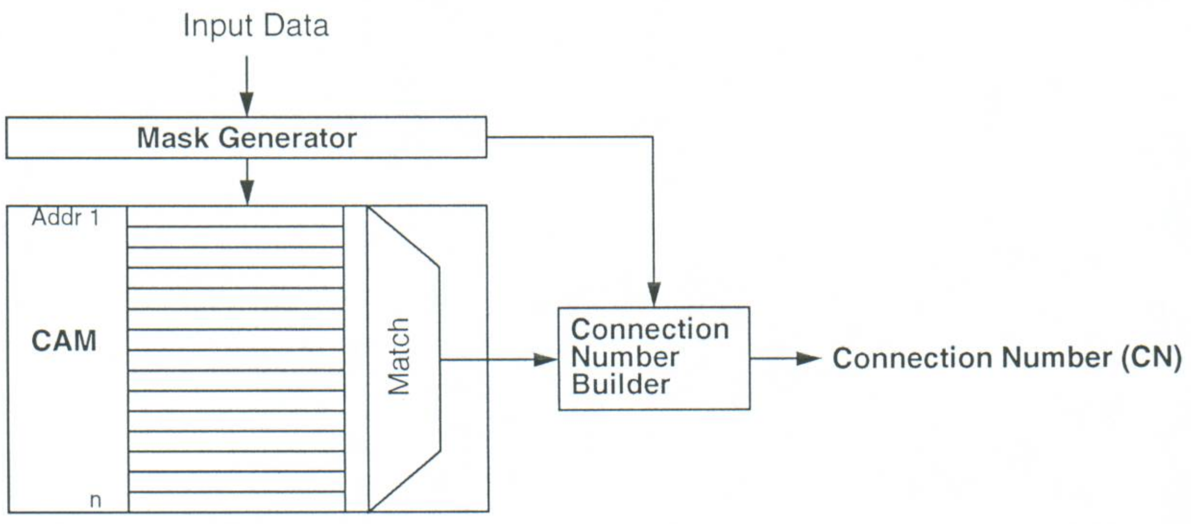

Figure 4. Protocol Filter

The architecture of the PF is shown in Figure 4. The PF filter consists of two hardware state machines built around the central CAM. The Mask Generator extracts the relevant header fields from the protocol header and generates a mask that is compared in the CAM. The

Connection Number Builder reads the addresses given by the CAM and builds a unique CN. As the connection detection works in O(1) time, the PF can be built to run at network speed. A detailed description of the PF is given in [15].

3.2. Coder/Decoder

The coder/decoder implements all operations on the content of a packet. Even though these operations could be implemented in dedicated hardware we chose a processor based solution because a processor can provide sufficient performance but adds more flexibility and functionality. The processor acts as coder/decoder and as controller. The coder controls the transmit side of the LWMA, the decoder controls the receive side. The coder/decoder implements the coding functions requiring reading and possibly changing the data of a packet, e.g., compression/decompression, check sequence calculation, and encryption/decryption. The CN is used to trigger the coder/decoder to perform the functions required by the protocol. For asynchronous traffic, the calculation of a check sequence is required. Encryption functions might be used for asynchronous transport connections requiring privacy. For isochronous traffic, audio coding and video compression functions are often necessary. The coder/decoder performs this function on the packet data in the transmit/receiver memory. For coding/ decoding functions that change the content and the size of a packet, the coder/decoder can move the data between the frame memory and the transmit/receiver memory while performing its function.

The coder/decoder can be implemented with a fast digital signal processor (DSP), which is optimized to perform the typical operations in coding such as e.g., MULTIPLY-ADD. DSPs are suited for this task because of their short interrupt latency. Their instruction set and cycle time allows for a frame check sequence calculation. An Mwave System [16] could be used to implement the coder and the decoder. The Mwave system consist of a fast DSP and an operating system that provides audio and imaging services. For real-time video coding additional VLSI implementing MPEG would be required.

3.3. Network Access Unit

The LWMA requires a network access unit (NAU) that provides a frame interface. In the presented architecture the NAU connects to a 155 Mb/s ATM network and implements the AAL (ATM Adaptation Layer) protocols. The NAU consists of a physical interface, an AAL unit, and a frame memory. The physical interface transforms between the analog network signals and the internal digital data representation. The AAL unit of the receive side (AAL_recv) receives ATM cells from the physical interface (Phy_in), reassembles AAL frames in the frame memory, and processes the AAL protocol. The AAL unit of the send side (AAL_send) processes the AAL and segments the AAL frame in the frame memory into ATM cells. The physical interface (Phy_out) sends the cells to the network. The coder/decoder controls the NAU and implements the ATM signalling protocol. The SARA chipset [17] could be used to implement the NAU.

3.4. Memory and DMA Unit

The memory is split into a transmit and a receive memory to reduce memory contention. The memory of the LWMA is Video RAM (VRAM). The DMAU accesses the serial port. The coder/decoder, the I/O bus interface, and the multimedia interface arbitrate on the parallel port.

The memory is clocked by the multimedia interface to provide multimedia devices optimal access. VRAM with 60ns access time provides a sufficient bandwidth for virtually parallel access of the coder/decoder and the two interfaces.

The DMA units move data between the parallel frame memory and the serial VRAM port. The DMA unit on the sender side provides gather DMA, whereas the DMA unit on the receiver side provides simple memory movement.

4. Performance Considerations

Protocol implementation on a workstation must be adapted to take advantage of the support functions of the LWMA. The LWMA is designed such that it can process all traffic in real-time.

The main function performed on the LWMA is header analysis. The CN generated can be used for 'protocol bypassing' [18]. I.e., a dedicated, optimized processing path is executed for a known connection. The most important application of protocol bypassing is the separated processing path for isochronous and asynchronous traffic on the LWMA. The coder/decoder offloads per-byte operations from the workstation that make up a considerable part of the protocol processing.

In the following we explain protocol processing of isochronous protocols on the example of ST-II and of asynchronous protocols on the example of TCP/IP.

4.1. Processing of ST-II

ST-II is a protocol that was developed for multimedia data streams. In the normal data transfer only the connection must be analyzed. ST-II provides an HID (Hop IDentifier) that defines a connection (a stream). Once the protocol type and the connection are identified by the CN the packets can be forwarded to its recv FIFO of the multimedia interface. If it is required by the connection the decoder decompresses audio or even video. The FIFO generates an interrupt on the multimedia bus and a multimedia device reads the data from the FIFO. The dedicated FIFOs for each multimedia connection allow for prioritized data transfer between adapter and multimedia device.

To send data the host processor sets up a connection and generates once a protocol header template. In subsequent send operation the multimedia device writes data to its send FIFO. The coder compresses the data and generates network packets by concatenating the header and the data. The processing speed of the isochronous traffic is limited by the compression/ decompression. ST-II can be processed at network speed because the header analysis is performed in hardware by the PF.

4.2. Processing of TCP/IP

The processing of asynchronous protocols like TCP/IP can be improved considerably by the LWMA. The parameters of TCP and IP should be set such that IP segmentation is prevented. The LWMA analyzes the header of an incoming packet and generates its CN. The CN is used to select and execute the check sequence algorithm. The CN, the calculated check sequence, and the pointer to the packet in the receive memory are written to the recv FIFO.

The workstation reads this FIFO and processes TCP/IP. The PF checks the validity of part of the header (TCP *and* IP) already, therefore an optimized processing path can be chosen.

IP processing is not required because the header analysis of the PF already translates the IP header information to a CN. A faulty or unknown header would result in a null CN and special error handling takes place.

The CN also represents the address header information of TCP. Therefore the CN can be used as a pointer to the TCP control block. The TCP header can be analyzed as with a header prediction algorithm without searching for the control block. The received TCP segments are appended to a socket buffer. The data are copied in the recv socket call to the application buffers.

For transmission of TCP/IP segments the host generates a header template that already contains the IP header and the static fields of the TCP header. The remaining fields are added for each packet to be sent. Header and Data are written to the transmit memory. The pointers to these buffers and the CN are written to the send FIFO. The coder calculates the check sequence and writes it to its field in the protocol header. Then the coder triggers the DMA unit to copy the data to the frame memory and signals the NAU over which AAL connection to send the data.

Of the 300 TCP/IP instructions[1] the LWMA replaces IP processing (57 instructions), plus the instructions to find the TCP control block. Thus, the LWMA support saves about 20% of the per packet cost of TCP/IP processing. The FIFO interface reduces considerably the operations of the buffer layer and the number of interrupt between host and adapter..

The performance of asynchronous protocols like TCP/IP can further be improved by using a *Single-Copy Implementation* [14]. The per-byte overhead of a TCP/IP implementation can be reduced by up to 66% compared to a conventional implementation by a single-copy implementation and by the implementation of the checksum on the LWMA[2].

5. Application Scenario

Distributed multimedia applications change the requirements on a computer. As these requirements are so different, it is better to talk about a multimedia station whose task is not primarily computing but multimedia communication between a user and a high-speed network. For this purpose the architecture of the multimedia station must provide sufficient communication channels between the multimedia devices and the networking devices. Röthlisberger, for example, suggests the use of a dedicated multimedia bus that connects the network adapter to the multimedia devices [19]. The LWMA architecture proposed above would well integrate into such a system (see Figure 7).

The Lancaster Multimedia Network Interface (MNI) [7] uses a similar setup by connecting multimedia I/O devices directly to the communication subsystem. Protocols are executed on the subsystem. The workstation acts as a controller and is not bothered with the handling of the high-bandwidth multimedia streams.

1. For this analysis we assume a TCP/IP implementation as described in [11]
2. We assume a two-copy implementation and similar overheads for copying and checksumming

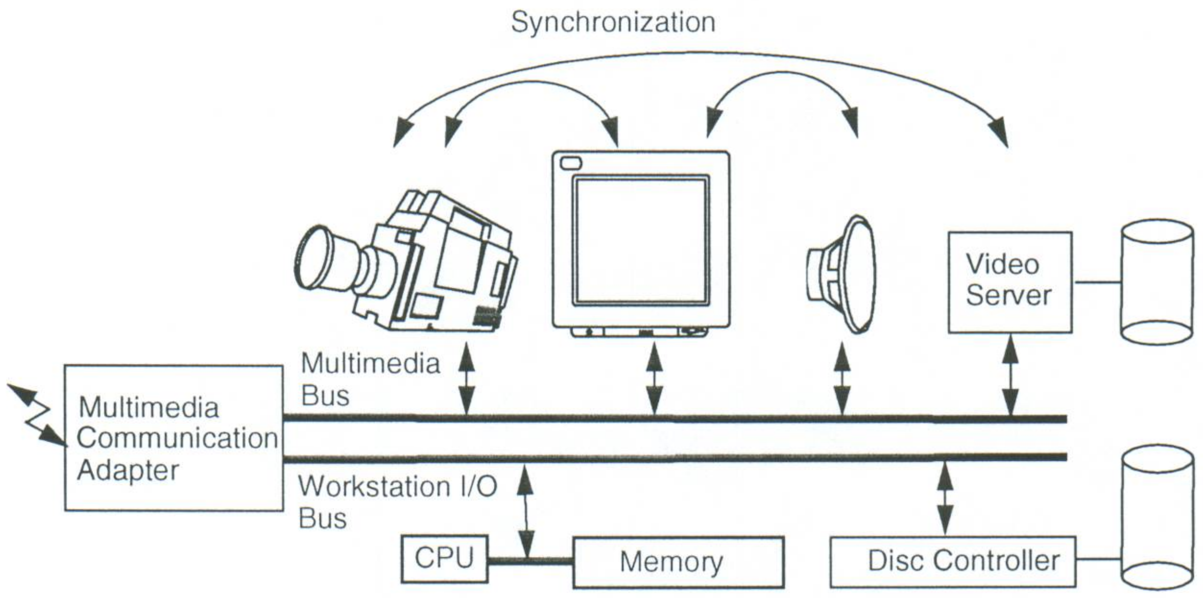

Figure 5. Application of a Multimedia Adapter in a Multimedia Station

While the MNI is a multimedia terminal, separated from the workstation, the presented multimedia communication adapter supports a close integration of real-time multimedia into a multimedia station. Figure 5 shows the application scenario of a multimedia adapter. The separated host interfaces connect to the I/O bus and the multimedia bus of a multimedia station. The multimedia bus connects all the multimedia end-devices and adapters. The multimedia adapter can provide coding and decoding functions that then need not to be implemented on the device adapter. With the addition of coding/decoding hardware the multimedia adapter could be enhanced to a multimedia communication desktop collaboration system [20].

The multimedia communication adapter can synchronize networked data streams. The adapter processors delay the reception or transmission of data until a predefined synchronization point between the data streams is reached. However, for efficient real-time synchronization at high speed, hardware support would be useful. The synchronization with workstation-internal data streams may be supported by the multimedia bus, e.g., with a timing signal.

Processing of multimedia data that require a reliable transport service is performed on the asynchronous transport path. The potentially necessary retransmissions to make a service reliable do not allow an isochronous service. In the example of teleradiology, the multimedia adapter would directly move the isochronous video between the network and the devices (digital camera or display memory). For X-ray data a reliable transport protocol, e.g., TCP/IP, would be processed before the image is written to the memory of a high-resolution display.

For applications, where price is a major argument, the LWMA provides inexpensive multimedia support. The additional hardware required to enhance an *intelligent* ATM adapter to a multimedia adapter is not too complex and costly. If only isochronous multimedia support is required, the architecture of the LWMA can even be scaled to Gb/s networks.

6. Conclusion

User programmable communication subsystems were so far not overly successful because they were too expensive and complicated for widespread use. Careful protocol implementation on the workstation was sufficient for most applications. However, for new classes of applications relying on multimedia data transported over high-speed networks, the current shallow subsystems are not sufficient. For high-speed multimedia communication the adapter must provide support to guarantee a certain quality of service.

The LWMA provides inexpensive support of multimedia and traditional data traffic. The header analysis in hardware is the key function that allows for the early separation of isochronous and asynchronous traffic on the adapter. The two types of traffic use different host interfaces: the standard I/O interface and a dedicated multimedia interface.

Instead of implementing the protocols on the subsystem, the LWMA performs CPU-intensive operations for coding and checksumming packet data. Protocol processing on the workstation is improved by the offloading of these operation and by the CN. The CN can be used for protocol bypassing. For multimedia traffic the multimedia FIFOs provide an interface for multimedia data streams at the full network speed of 155 Mb/s. The connection to the multimedia bus and the multimedia device needs further investigation, because these interfaces are not yet fully understood.

The protocol processing support of the LWMA for protocol filtering and for coding/decoding can be used already in traditional workstation architectures that provide a single I/O bus.

I would like to thank Matthias Kaiserswerth for his valuable comments.

7. Bibliography

[1] Hehmann, D., Salmony, M., Stüttgen, H., "High-Speed Transport System for Multimedia Applications," in Protocols for High-Speed Networks, Elsevier/North-Holland, Amsterdam, 1989.

[2] Ramakrishnan, K.K, "Performance Considerations in Designing Network Interfaces," IEEE Journal on Selected Areas in Communications, Vol. 11, No. 2, February 1993.

[3] Steenkiste, P., "Analysis of the Nectar Communication Processor," Proc. IEEE Workshop on the Architecture and Implementation of High Performance Communication Subsystems, Tucson, Az, Febr. 1992.

[4] Kaiserswerth, M., "The Parallel Protocol Engine," IEEE/ACM Transactions on Networking, Vol 1, No. 6., Dec. 1993.

[5] Braun, T., Zitterbart, M., "Parallel Transport System Design," IFIP Conference on High Performance Networking, Liege (Belgium), 1992.

[6] Ulrich, R., "Untersuchungen an einem OSI-Kommunikationswerk auf Transputer-basis," Dissertation, Arbeitsberichte des Institus für Mathematische Maschinen und Datenverarbeitung der Universität Erlangen-Nürnberg, Band 26, Nr. 11, Sept. 1993.

[7] Blair, G., Campbell, A., Coulson, G., Garcia, F., Hutchinson, D., Scott, A., Shepherd, D., "A Network Interface Unit to Support Continuous Media," IEEE Journal on Selected Areas in Communication, Vol. 11, No. 2, Febr. 1993.

[8] Lumley, J., "A High-Throughput Network Interface to a RISC Workstation," Proc. IEEE Workshop on the Architecture and Implementation of High Performances Communication Subsystems, Tucson, AZ, Feb. 17-19, 1992.

[9] Rütsche, E., "The Architecture of a Gb/s Multimedia Adapter," ACM Computer Communication Review, Vol. 23, No. 3, July 1993.

[10] Inmos Limited, "The T9000 Transputer Products Overview Manual," Inmos 1991.

[11] Clark, D., Lambert, M.I., Romkey. J., Salwen, H., "An Analysis of the TCP REptocessing Overhead," IEEE Commmunications Magazine, Vol. 27. No. 6, (June 1989).

[12] Topolcic, C. (Editor), "Experimental Internet Stream Protocol, Version 2 (ST-II)," RFC 1190, October 1990.

[13] Steenkiste, P., Zill, B., Kung, H., Schlick, S., "A Host Interface Architecture for High-Speed Networks," IFIP Conference on High Performance Networking, Liege (Belgium), Dec. 1992.

[14] Dalton, C., Watson, G., Banks, D., Calamvokis, C., Edwards, A., Lumley. J., "Afterburner," IEEE Network, July 1993.

[15] Rütsche, E., "Multimedia Communication Subsystems: Architectures, Interfaces and Implementations", Ph.D Thesis ETH Zürich, Nr. 10228, VDI Verlag, Reihe 10, Nr. 257, Düsseldorf 1993.

[16] Texas Instruments, "MWave Multimedia System," Technical Brief, Texas Instruments,1992.

[17] Transwitch Corporation, "SARA Chipset Technical Manual," Document Number TXC-95000, Edition 1, June 1992.

[18] Thia, Y.H., Woodside, C.M., "High-Speed OSI Protocol Bypass Algorithm with Window Flow Control," 3rd. Int. IFIP WG.6.1. Workshop on Protocols for High-Speed Networks, Stockholm, May 1992.

[19] Röthlisberger, U., "A Network for a Multimedia Communication System," IEEE Workshop on the Architecture and Implementation of High Performance Communication Subsystems, Tucson, Arizona, Febr. 1992.

[20] Chen, M.-S., Shae, Z.-Y., Kandlur, D.D., Barzilai, T.P., Vin, H.M., "A Multimedia Desktop Collaboration System," in Proceedings GLOBECOM 92, IEEE, Dec. 1992.