NODE EMULATOR ARCHITECTURE FOR THE U.C. BERKELEY PROTOCOL WORKROOM FACILITY

by

Ayman Fawaz, Lester Ludwig and Mike Peck

Memorandum No. UCB/ERL M85/84

(Protocol Workroom Document No. 85-3, version 2)

15 November 1985

COVER PAGE

NODE EMULATOR ARCHITECTURE FOR THE U.C. BERKELEY PROTOCOL WORKROOM FACILITY

by

Ayman Fawaz, Lester Ludwig and Mike Peck

Memorandum No. UCB/ERL M85/84

(Protocol Workroom Document No. 85-3, version 2)

15 November 1985

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

NODE EMULATOR ARCHITECTURE FOR THE U.C. BERKELEY PROTOCOL WORKROOM FACILITY

Ayman Fawaz, Lester Ludwig and Mike Peck

U.C. Berkeley "Protocol Workroom" Development Group
Department of Electrical Engineering and Computer Sciences
University of California at Berkeley
Berkeley, California 94720

Protocol Workroom Document No. 85-3, version 2

ABSTRACT

The *Protocol Workroom* is a facility for studying the performance of communication protocols. It consists of several node emulators and a channel emulator. Each node emulator consists of a 68010-based single board computer and a 68020-based board that functions as a data link layer controller. The node emulators communicate over the channel emulator which can emulate various network topologies such as a broadcast channel or a ring. They are also connected to a backbone network along with a file server and a SUN workstation which provide control, monitoring, and analysis functions.

This document describes the node emulator that will be used in the *Protocol Workroom* facility.

NODE EMULATOR ARCHITECTURE FOR THE U.C. BERKELEY PROTOCOL WORKROOM FACILITY

Ayman Fawaz, Lester Ludwig and Mike Peck
U.C. Berkeley "Protocol Workroom" Development Group
Department of Electrical Engineering and Computer Sciences
University of California at Berkeley
Berkeley, California 94720

1. INTRODUCTION

The *Protocol Workroom* [1] is a facility for studying the performance of communication protocols. It consists of several node emulators and a channel emulator. The node emulators communicate over the channel emulator [2] which emulate various network topologies such as a broadcast channel or a ring. They are also connected to a backbone network along with a file server and a SUN workstation which provide control, monitoring, and analysis functions.

Each node emulator consists of two parts: an MC68010-based single board computer made by Pacific Micro Systems and a custom board based on the MC68020. This is shown in Figure 1. The MC68020-based board performs the functions of the data link layer. The Pacific board performs the functions of the higher level protocols, often lumped together and called the client layer.

Several tasks will be accomplished by this node emulator. The Pacific board will be used for traffic generation, data processing and applications software. The link layer controller board will execute the transmit and receive functions and will also implement monitoring functions for real-time evaluation of protocols.

Most of the tasks mentioned above will be executed concurrently. For the higher level protocols, response time constraints are not too stringent. This justifies the use of a single processor capable of task switching to implement these protocol functions. However, the response time at the link layer is a critical issue. This factor, and the high flexibility needed to implement a wide class of link layer protocols, require the use of dedicated hardware to run specialized tasks, in conjunction with the main processor.

2. EMULATING THE CLIENT LAYER

The Pacific board [3] can be used to emulate various digital devices such as a computer or a file server. The board is managed by the Motorola 68010 CPU running VRTX [4], a real time operating system that resides in ROM.

This operating system provides a multitasking environment for the implementation of higher level protocols, and for the procedures used to communicate with the data link layer controller. An event-driven, priority-based scheduling algorithm is used for CPU allocation. A process lock mechanism is used when running critical sections. This mechanism postpones scheduling decisions until a corresponding unlock is executed. Interprocess communication is realized by the use of mailboxes and the exchange of messages.

Message generation and most protocol features above level 2 (network, transport, etc.) required for the experiment are implemented on the Pacific 68010 system. In fact, each Pacific board contains a local message generation scheme implemented in software. The generation of data (e.g., packets, blocks, spurts, etc.) for transmission over the experimental network is programmable. Sample paths of a specified stochastic process modeling transmit times, destination addresses, and packet/block/spurt lengths can be realized with locally implemented pseudo-random generators. Table-driven deterministic processes can also be used, as well as combinations of these and generated sample paths.

Depending on the experiment that the user is interested in running, one could have different schemes of operations. One scheme is to generate packets and to directly transfer them to the controller board for link layer processing and transmission. This scheme may be used for the study of link layer protocols. Another scheme is to generate messages, process them by executing software procedures implementing high levels protocols, and transfer them to the controller board where they go through link layer processing and transmission. In this case, the interaction of link layer protocols with higher level protocols can be investigated as well as the performance of the higher level protocols themselves.

3. DESCRIPTION OF THE PROTOCOL WORKROOM LINK LAYER CONTROLLER

The data link layer controller is a custom designed board based on the MC68020 [5]. The board serves as an interface between the Pacific board and the channel emulator. The 68020 supervises the operation of the other components on the controller board, and it performs computations associated with the protocol being implemented. It is also responsible for communication with the 68010 host and the collection of monitoring data.

Besides the 68020 microprocessor, the board consists of programmable state machines used for pattern recognition and fast decision making, and of dedicated hardware for packet transmission, packet reception, and event recording. The main idea behind the link layer controller architecture is the use of dedicated hardware to relieve the board processor from real time constraints. Figure 2 gives a general overview of the board architecture.

3.1. Communication Between the Pacific Board and the Data Link Controller

This operation is controlled by interrupt signals used to inform the destination processor about the presence of valid data. All control messages and data exchanges between the two boards take place through mailbox memory realized with dual-port RAM. Any processor desiring to send information to the other one stores the data in the dual-port RAM and sends an interrupt signal to the other processor to notify it.

The dual-port RAM is divided into different mailboxes. Two are used to buffer packets: the first is dedicated to the packets to be transmitted, and the second to the received ones. Another mailbox contains monitoring information that is associated with the transmitted and the received packets. Other mailboxes are also used to exchange control messages. These messages will contain information that is pertinent to the interrupt signal which follows. This partition is taken into consideration by the service routines of the different interrupt signals and this fact will prevent any consistency problem in the shared memory.

The 68010 controls the reset signal of the 68020. This feature is implemented because during the initialization phase, the 68010 modifies the configuration of the link layer controller board and therefore must control it.

The transfer of information takes place through a private bus connecting the two boards, the P2 bus. This bus is similar to a system bus connecting the CPU to its peripherals.

3.2. Channel Emulator Interface

The controller board is designed to interface to the channel emulator. The interface consists of several lines in parallel. Two of these are data in/out lines, and the rest are used for control and timing signals. Among these, there are:

- signals to indicate valid data on the data line
- clock signals that are used by the controller hardware for transmission and reception
- global clock signals used for a global time stamp used for monitoring.

3.3. Programmable Finite-State Machines

There are programmable state machines to control transmission, reception, and event recording. They monitor the channel and perform pattern recognition and fast decision making. All these functions are highly dependent on the communication protocol used in the network and are user definable. These functions regulate the access to the channel by following the media access control (MAC) protocol implemented. They consist of looking at signals from the channel emulator, identifying them, and making decisions regarding packet transmission and reception.

The rest of the board communicates with the state machines by setting flags that indicate state information of the link layer controller. The state machines send interrupt signals to the other subsystems with some encoded information describing the events that are occurring. From these signals, the subsystems determine what action should be taken.

There are two types of state machines. The first one identifies the control signals or the patterns sent on the channel, while the second one makes the correct decision associated with the current state of the node and the result of the pattern search performed by the first one.

3.4. Transmission

Transmission is controlled by the 68020 and the state machines. A transmission occurs in the following way. A packet is generated and stored in the buffer by the 68010. An interrupt signal from the 68010 to the 68020 informs the latter about the presence of the packet. A dedicated piece of hardware is used to transfer data from the memory to the channel. This piece of hardware is controlled by the 68020 and the state machines. After the transfer of the last byte in the packet, the data transfer hardware sends a signal to the state machines to indicate this event. The state machines put the required signals on the channel, and inform the 68020 about the result of the transmission attempt. In case of an error, for example, a packet collision in an Ethernet emulation, the state machines stop the data transfer, take the necessary actions, and inform the 68020. In response, the 68020 resets the dedicated hardware to start a new transmission attempt. The 68010 is notified of the outcome of a transmission attempt and later collects the monitoring data associated with it.

A CRC calculation can be performed in hardware while transmitting the packet, and the resulting checksum can be appended to it.

3.5. Reception

Packet reception is performed in the following way. Whenever the state machines detect an incoming packet, they send a signal to a dedicated piece of hardware that is used to transfer data from the channel to the buffer memory. After storing the received packet in memory, the state machines send a signal to the 68020 informing it about the event that took place. The 68020 checks the packet, collects the monitoring information associated with it, and sends an interrupt signal to the 68010, so that the latter can read the packet and the monitoring data.

A CRC check can be done in hardware while reading the packet from the channel, and the result is stored with the other monitoring information.

The use of a separate piece of hardware to perform packet reception is essential, so that a node can transmit and receive simultaneously at 10 Mb/s. This feature is required for some network architectures.

3.6. Experimental Data Collection

Since the channel is operating at 10 Mb/s, the experimental data collection requires a fast implementation. Therefore, many subsystem blocks cooperate with the processor to perform the event recording in our monitoring architecture. The state machines identify the occurrence of a MAC event and then inform a hardware unit consisting of a FIFO and a time stamp circuit. The state machines provide this hardware unit with an event code. This event code is stored in the FIFO with a time stamp. After the completion of a packet transmission or a packet reception, the 68020 collects the event information from the FIFO, processes it, and stores it in the mailbox memory, where the 68010 on the Pacific board can read it.

3.7. Other Protocol Functions

Many other protocol functions can also be implemented on the link layer controller board. Some of these functions belong to the data link layer, and other higher layer functions can also be implemented to relieve the 68010 CPU board of its burdens.

An example of data link layer functions that can be performed is the truncated binary exponential backoff retransmission control policy used in Ethernet whenever a collision occurs. Examples of higher layer functions that can be implemented are packet formation, packet reception acknowledgement, flow control, and even some routing functions in the case of a gateway. All these additional functions are realized by software procedures executed by the 68020 processor.

4. INTER-PROCESSOR COMMUNICATION

An important feature of the node emulator is the communication scheme between the two processors: the 68010 host and the 68020. This function is based on a request/reply message exchange scheme, using dedicated mailboxes. A request or a reply message consists of a hardware interrupt signal and the associated control data for the processor which services the interrupt.

4.1. The Case of Data Transmission

To begin a transmission, the 68010 host sends a transmit request message to the link layer controller board processor. This message contains information related to the packet that was stored in the buffer by the host itself. This information is used by the transmission procedure implemented on the controller. By doing so, the host gives the 68020 full control over the buffer containing the packet, and therefore cannot use the buffer until it receives a reply message. The reply message is sent by the 68020 upon servicing the transmission request. In communication protocol terminology, this reply message corresponds to the indication or the confirmation message. Only then can the host processor access the buffer for additional data transmission requests.

There is a pool of four buffers that are used for this type of data transfer. Two of these buffers are reserved for the 68020 for link layer messages, and the other two are used by the 68010 host. The link layer controller handles one outgoing packet at a time. The 68010 host can store a second packet to be transmitted in the second buffer and request its transmission. This request is queued in the controller and is serviced upon finishing the transmission of the current packet and if there are no link layer packets waiting to be transmitted. All the other packet transmission requests are queued in the 68010 board.

The use of two buffers in the scheme described above reduces the delay introduced by the hardware between packet generation and packet transmission. If a packet is generated while a previous one is being transmitted, the new packet is stored in the second buffer where it waits for transmission. By following this method, the whole inter-board transfer time of the second packet has overlapped with the transmission time of the previous one. The use of more than two buffers does not affect this delay at all since when the 68020 is servicing the second packet, the first buffer is available to the 68010 and there is no need for more buffers. This scheme reduces the memory used to buffer outgoing packets.

4.2. The Case of Data Reception

In the case of the received packets, the following takes place. The 68020 sends a receive request message to the host processor. This message contains the starting address and the length of the data block that has to be read. By doing so, the 68020 gives the host processor full control over the buffer. After reading the data from the buffer, the 68010 host updates the value of a hardware "shadow" pointer that stores the address of the last buffer location read by the 68010. Then, it sends a reply message to the 68020 giving back the control over the buffer.

The buffer used to store received packets is a cyclic queue, and can store 16K words of data. Notice that link layer messages received by the node will be stored in this buffer but will not be read by the host processor. If the received packets buffer is full, a special control message is sent to the 68010 to notify it.

4.3. Other Types of Messages

In addition to the transmit and the receive requests, many command requests are initiated by the host processor. Among them is the command telling the link layer controller board to respond only to certain address classes. Other commands are initiated by the 68020. One example is the control message that requests from the 68010 the collection of the monitoring information gathered by the link layer controller.

5. THE MONITORING FUNCTION

To investigate and compare the performance of different protocols and architectures, a monitoring and analysis system is required. For this reason, a highly flexible monitor is implemented in the facility. Figure 3 gives a general overview of the Protocol Workroom monitoring system. This section focuses on the monitoring functions of the node emulator.

MAC event monitoring is done in the way described in 3.6. Any event occurring on the board is associated with an interrupt signal initiated by one of the subsystems. These interrupts are the key elements in the implementation of the monitoring function, since they are used to control the hardware responsible for storing the events codes and time stamps in the FIFO. These same interrupt signals, if sent to the 68020, start the execution of the monitoring data handling routine. An important feature of this function is that these events are user defined. In fact, the user's definition of the events to be monitored is based on two parameters: the MAC protocol implemented and the experiment that will be running to determine a specific performance measure. This feature is common to the event monitoring function performed at any level, but is noteworthy at the MAC level because it is implemented in hardware.

The 68020 collects all events codes stored in the FIFO and save them in the monitoring information mailbox. A specific data structure is used to save this information for the 68010. A data record is associated with each packet serviced by the controller board. This record contains the packet identification number, the packet length, the packet source address, the packet destination address, and all the events codes (and time stamps) that are related to it. Each record has a flag that is reset by the 68010 when the latter finishes reading it. A record also contains its length. This information is used by the 68010 when it reads the record. This data consolidation process have a lower priority than other processes that are more crucial for the normal operation of the controller, for example processes that service packets.

After reading the monitoring information, the 68010 host adds to it the information gathered on the Pacific board, consolidates it, and stores it waiting for the SUN master computer to collect it.

Another important feature of the system is the programmable record filtering function that is performed by both processors. This function is specified by the user and depends on the experiments that will be selected. It allows both, the 68010 and the 68020, to discard information when processing it before the transfer to the higher layer. The information that is left out and not transferred to the master computer is not relevant to the performance measure that the user is trying to estimate by running his experiment.

6. DESCRIPTION OF THE STATE MACHINES

The state machines are intended to orchestrate all actions and decisions related to the MAC protocol functions.

The high level features for the state machines are:

* Simple programming;
* Independent and simultaneous handling of transmission and reception events;
* Response capability within one bit period.

To accomplish these goals, the state machines control a specific set of operations. These operations are defined in terms of special control commands to and from peripheral hardware subsystems on the controller board.

The operations performed by the state machines ensure the monitoring and control of:

1. Pattern recognition:
   a. Support of patterns with length less or equal to 64 bits;
   b. Support of "don't care" fields;
   c. Support of 32 consecutive different patterns;
   d. Simultaneous search for 4 patterns;
2. Receive-port to Transmit-port coupling with flexible overwrite capability;
3. Interaction with hardware for data transfer between shift registers and packet buffers;
4. Operation of event FIFO subsystem upon MAC events occurrences;
5. Interaction with 68020:
   a. From 68020 (via status flags);
   b. To 68020 (via 68020 interrupts);
6. Provision of required signals to the Channel Emulator interface;
7. Delay operations to defer actions as required.

The peripheral subsystems include supporting circuitry to assist the state machines in ensuring these tasks. For example, delays and timing functions are implemented with programmable timers.

Because of the need for autonomy of the transmission and reception functions, the state machines are decomposed into two essentially decoupled parts. Provisions for coupling are included to support the interaction between these two functions, depending on the MAC protocol implemented.

7. A MULTIPORT CONFIGURATION

The multiport configuration is illustrated in Figure 4. Up to 8 controller boards can share a single Pacific board through the P2 bus. There are also two other buses as shown in Figure 4. In the multiport configuration the P2 bus is used for centralized control, employing the Pacific board as the central controller. The P3 bus is used for high speed interconnection between the controllers in order to exchange data. The P4 bus is included to provide peer communication among the controllers 68020s. This permits the use of decentralized algorithms to control a multiport configuration under study.

The multiport configuration can act as a circuit, packet, cut-through, or integrated switch. Since the rest of the network controllers are free to implement any compatible protocol, the multiport can also function as an integrated services gateway in addition to more traditional gateway functions.

8. CONCLUSION

The functionality of the Protocol Workroom node emulator can be programmed to simulate different types of traffic generations by using different methods and to implement different protocols. The major characteristic of the link layer controller is the use of separate blocks to perform the different tasks ordinarily associated with the data link layer. This is the key to its architecture's flexibility.

9. ACKNOWLEDGEMENTS

The authors are grateful to Professors P. Varaiya and J. Walrand for their suggestions, encouragement, and support. Many thanks are also extended to Randy Cieslak for his valuable comments on this paper. The research is supported in part by NSF grants ECS-8118213/85-08337, ONR contract N00014-80-C-0507, JSEP contract F49620-79-C-0178, and grants from Bell Communications Research and the MICRO program.

10. REFERENCES

[1] A. Fawaz, D. Giralt, L. Ludwig, "The Protocol Workroom: An Experimental Protocol and Distributed System Research Facility for U.C. Berkeley", Protocol Workroom Document No. 84-1, version 2, June, 1985, UCB, EECS Dept.

[2] L. Ludwig, "Channel Emulator for the U.C. Berkeley Protocol Workroom Facility", Protocol Workroom Document No. 85-1, February, 1985, UCB, EECS Dept.

[3] The Pacific Board User's Manual.

[4] The VRTX User's Manual.

[5] The Motorola 68020 User's Manual.

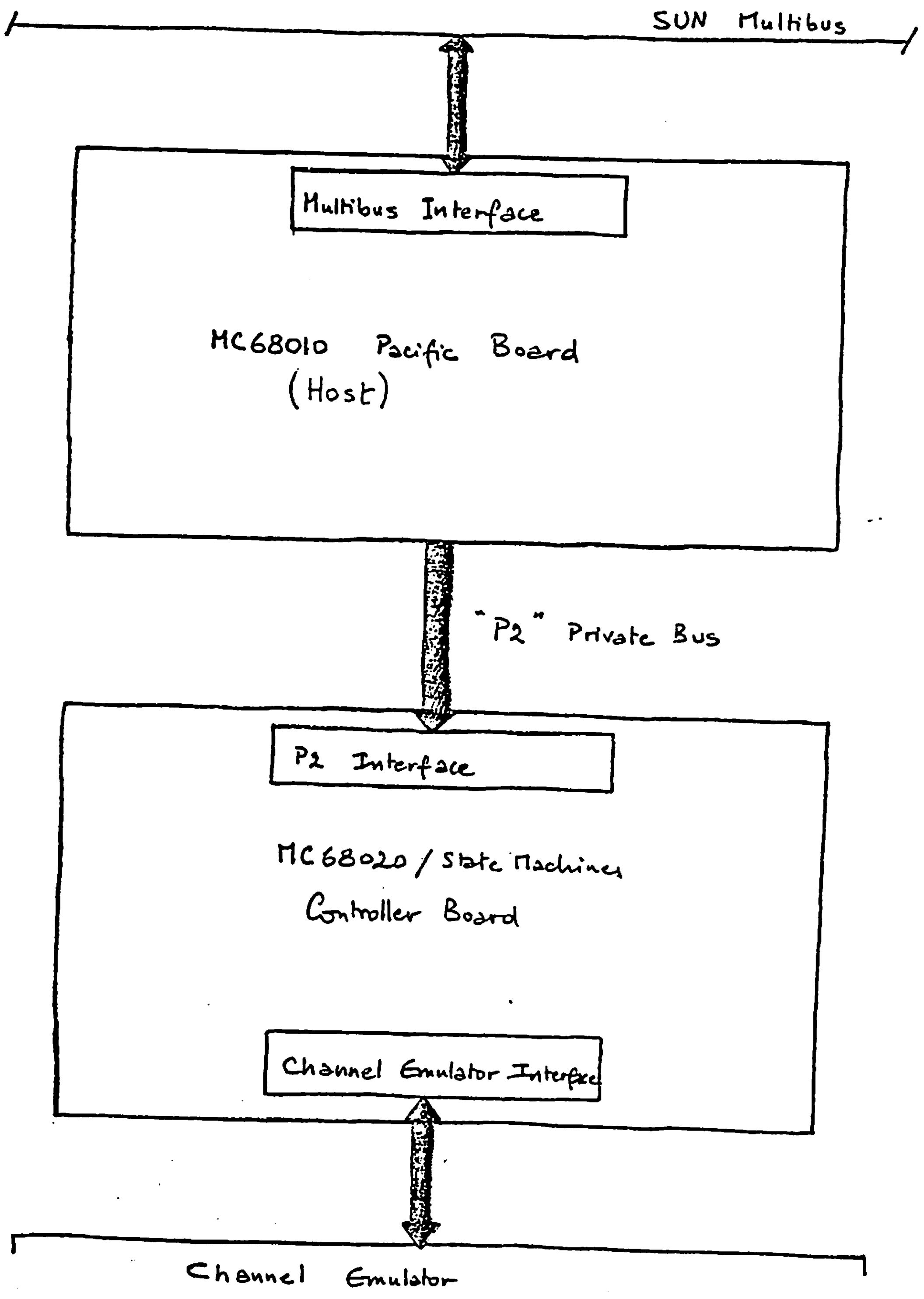

Figure 1. General Overview of the P.W. Node Emulator

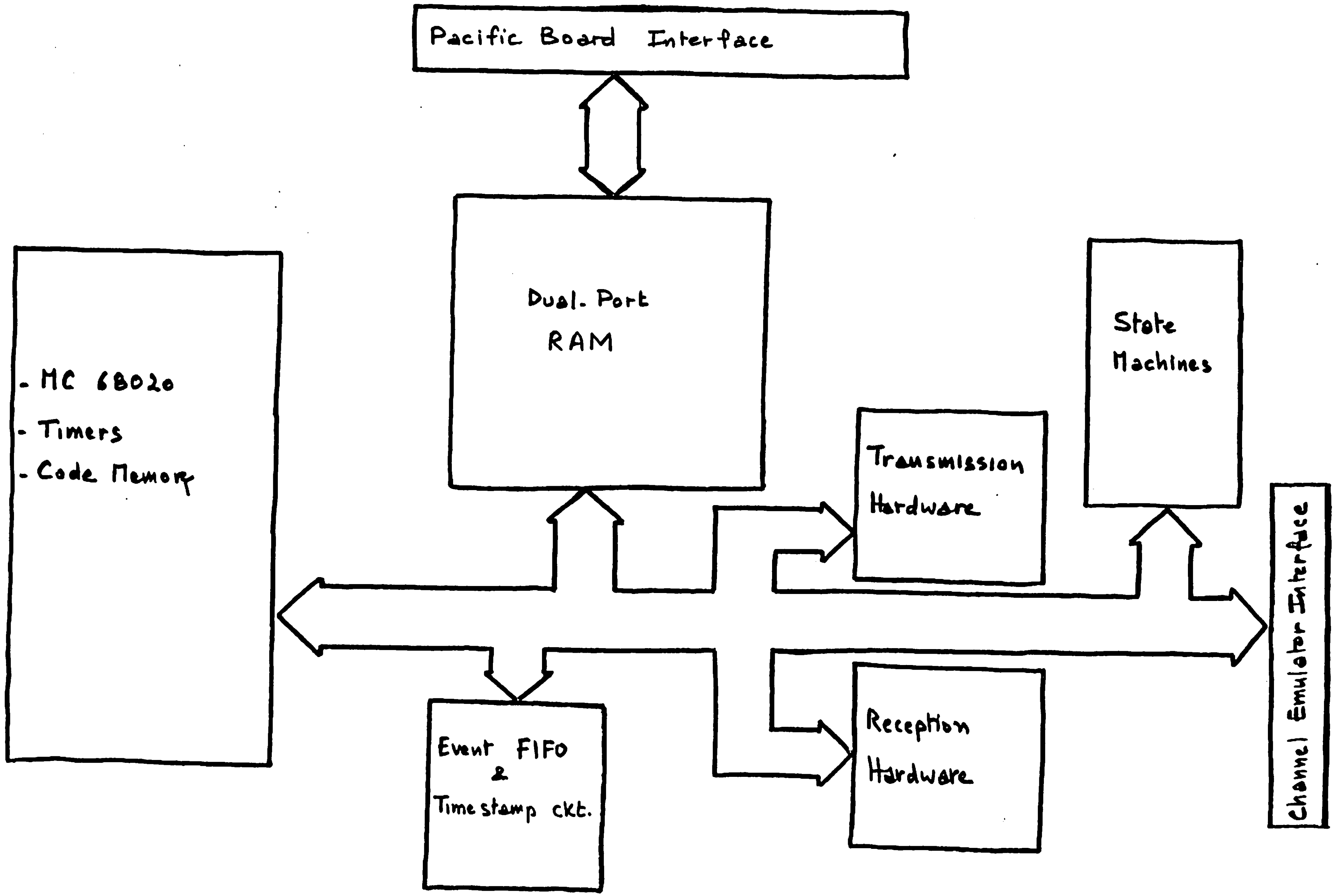

Figure 2. General Overview of the P.W. Controller Architecture

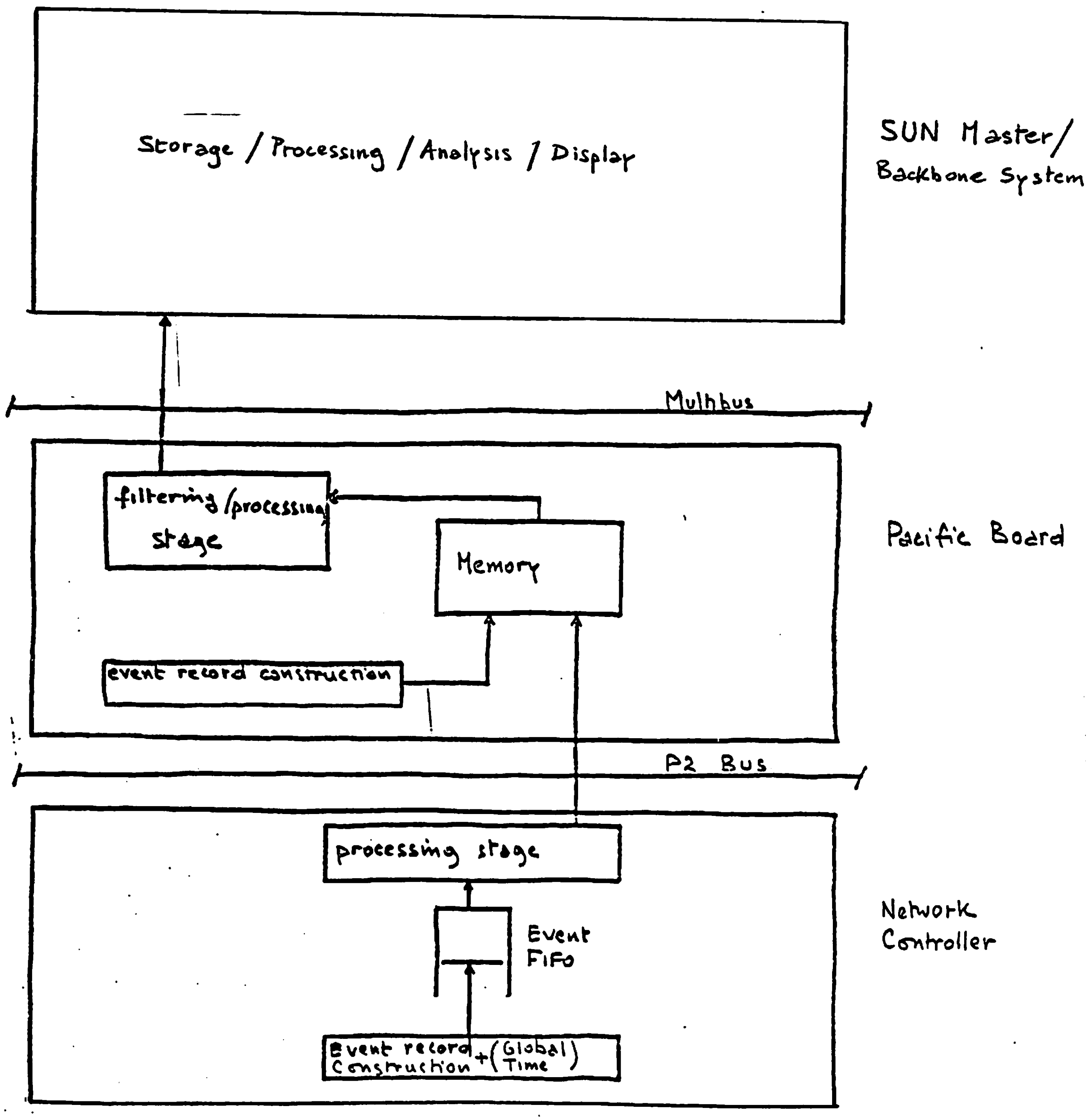
Storage / Processing / Analysis / Display
SUN Master/ Backbone System
Multibus
filtering/processing stage
Memory
Pacific Board
event record construction
P2 Bus
processing stage
Event FIFO
Network Controller
Event record Construction + (Global Time)

Figure 3. General Overview of the P.W. Monitoring System

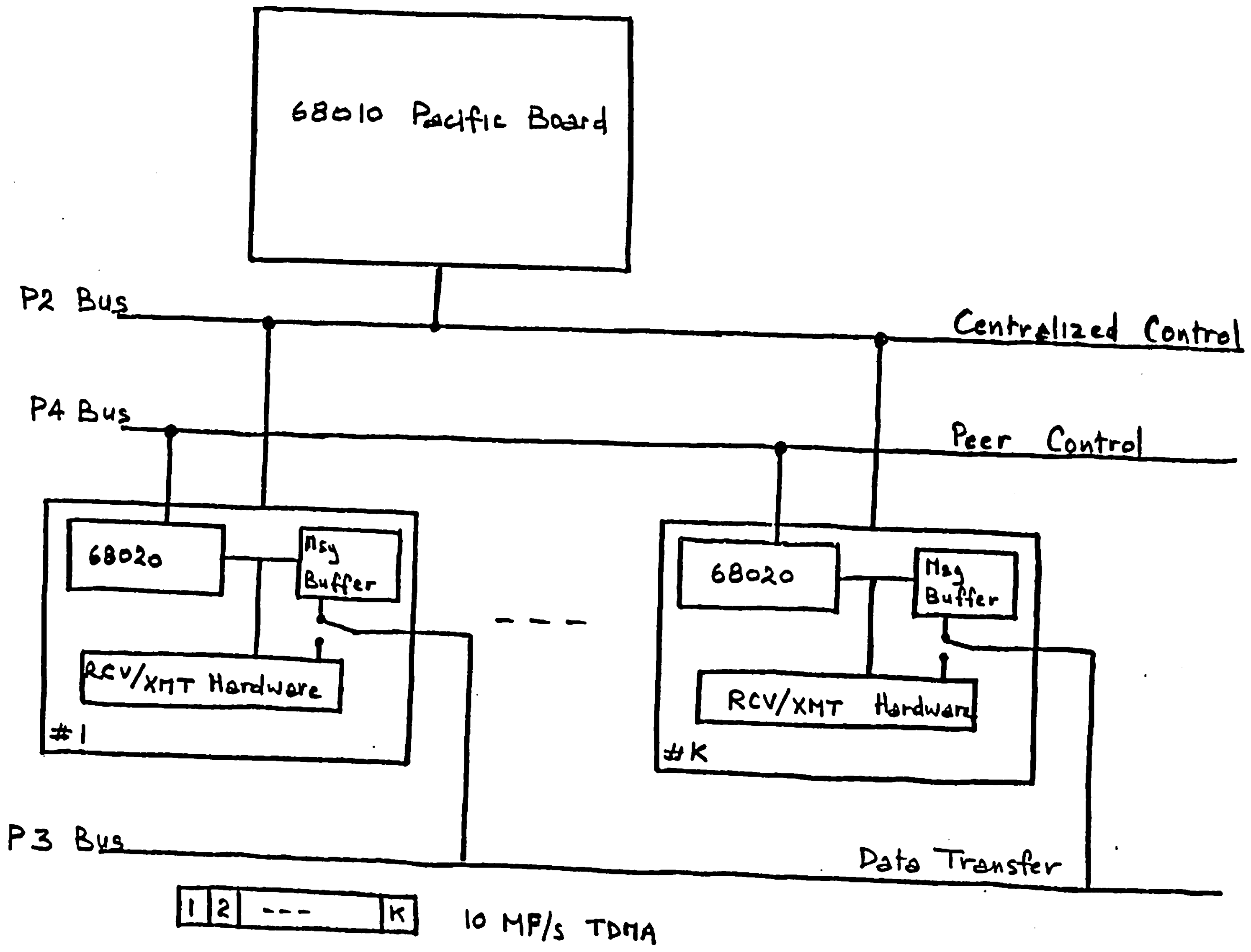

Figure 4. K-port version of the Node Emulator