The design of a token-ring to token-ring local area network bridge

by

Paul Michael Freeman

A Thesis Submitted to the

Graduate Faculty in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

Department: Electrical Engineering and Computer Engineering
Major: Computer Engineering

Approved:

Iowa State University
Ames, Iowa

1990

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.1 Introduction to Local Area Networks

During the last decade, the decreasing cost and increasing performance of VLSI components has brought about a change in the way that people use computers. Advances in VLSI technology have made possible the development of powerful microprocessor based workstations and personal computers. Rather than accessing a time-shared mainframe through an ASCII terminal, it is now often more economical and convenient for each user within an organization to be provided with his or her own personal computer. This distributed computing approach allows for easy expansion and flexible architectures. In addition, the use of powerful workstations with built-in high speed graphics provides the ability to execute programs such as window and icon based operating systems and high speed animation programs that aren't feasible with text only terminals connected to mainframes by low speed lines.

This trend toward distributed workstation based computing has created the need for local area networks (LANs) that provide high speed computer-to-computer communications within a limited geographic span. Users of a distributed system typically require the ability to access remote files, send electronic mail to other users, and access shared peripherals such as high quality printers. Standardized methods of providing this capability must exist in order that equipment from different vendors be interoperable. The Institute of Electrical and Electronic Engineers (IEEE) has recognized the need for such standardization and has created the IEEE

Standard 802, a family of standards for local area networks.

The IEEE 802 family consists of a set of six standards that specify the physical and data link layers for several local area networks. Figure 1.1 illustrates the relationship of these standards.

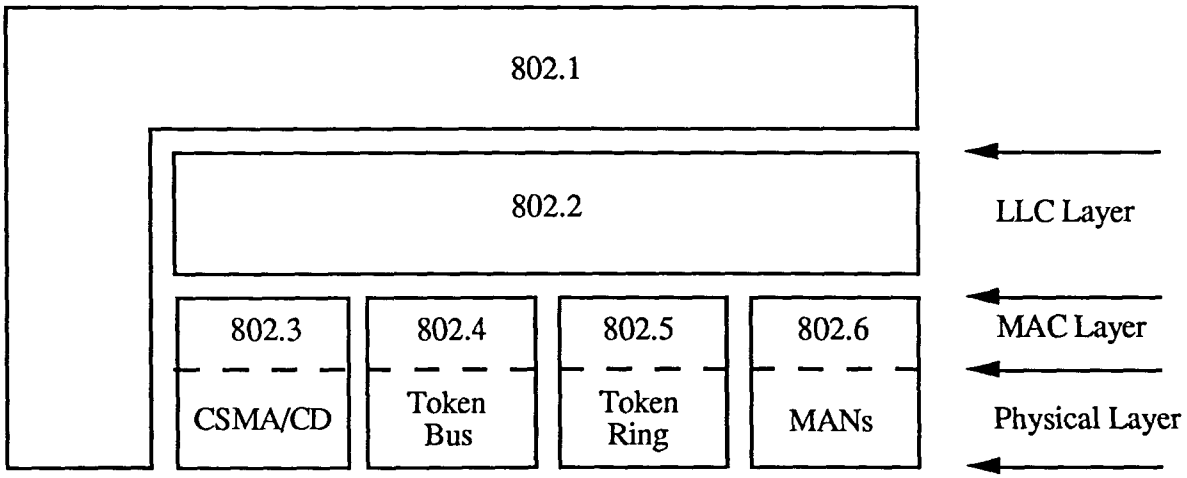


Figure 1.1 IEEE 802 local area network standards

The 802 standards have divided the data link layer into two sublayers: the media access control (MAC) sublayer and the logical link control (LLC) sublayer. Four separate standards exist for the MAC sublayer, corresponding to the four local area networks that have been specified by the IEEE. Each standard describes the physical interface and the method for controlling access to the channel for its specific network. The four networks defined by the IEEE 802 committee are described in standards 802.3 (CSMA/CD network), 802.4 (token bus network), 802.5 (token ring network), and 802.6 (metropolitan area network).

The LLC sublayer specifies procedures for connection oriented and connectionless data transfers between two stations on a local area network. These procedures are independent of the local area network being used, and are described in standard 802.2. Finally, standard 802.1 describes the relationship between the various 802 standards and specifies methods for

interconnecting two or more 802 local area networks together. This thesis is specifically concerned with the interconnection of two 802.5 local area networks.

1.2 Methods of Interconnecting Local Area Networks

Due to a variety of reasons, several local area networks may coexist within an organization. For example, any of the following may motivate the existence of multiple LANs within an establishment:

- Local area network specifications stipulate a maximum number of nodes that may be connected to the network and a maximum cable length for the network. When either of these limits is reached, it may be necessary to partition the original network into two or more smaller networks when expansion is desired.
- As a network grows, its maximum allowed bandwidth may become a bottleneck, limiting overall throughput. Separating the network into two or more smaller networks can provide a means of increasing the total available bandwidth, thus alleviating this problem.
- Different departments within an organization may wish to install and manage their own networks. This may be desirable for network security considerations, administrative reasons such as accounting purposes, or because the different departments wish to implement different types of networks. For example a network connecting robotic equipment on a factory floor may dictate the use of a LAN that guarantees a deterministic worst case access time, whereas an office environment of low cost personal computers may motivate the use of a relatively inexpensive network.

Even though separate networks may exist within an organization, it is still desirable to be

able to interconnect these networks in order to provide for communication between the nodes that reside on the different LANs. Three methods of LAN interconnection are commonly used. These are described below:

1.2.1 Repeaters

A repeater is a physical layer device used to link together two cable segments. The repeater amplifies the signals appearing on one cable segment and retransmits these onto the other cable segment. The amplification and retransmission are done for both directions, and thus with the repeater the two cable segments appear as one (longer) cable. Repeaters are used to increase the physical distance spanned by a network. However, they do not increase network throughput, nor do they allow dissimilar networks to be interconnected.

1.2.2 Bridges

A bridge is a device that operates within the MAC sublayer of the data link layer to connect individual local area networks into a "bridged local area network". A bridge connects two or more local area networks by capturing and buffering complete frames from one network before forwarding these frames to another network. In a bridged local area network, a station transmitting a frame addresses the frame to the desired destination station rather than to the bridge. The bridge examines the destination address of each frame captured, and only forwards those frames which are addressed to a node on another network that the bridge is connected to.

The store and forward operation of a bridge is necessary to allow the bridge to gain access to the transmission medium of the destination network before forwarding the frame. Thus a bridge introduces extra end-to-end delay over a repeater. However, since each network attached to the bridge operates in parallel, a bridged local area network can offer greater total

throughput than a set of cable segments connected by repeaters. In addition, a bridge can be used to connect local area networks with different media access control procedures. In this case the bridge may need to manipulate the received frames so that they are compatible with the second network before retransmission. However, it is not always possible to map frames between the different media access control protocols in a completely transparent manner. For example different MAC layer protocols specify different values for the maximum frame length which may be transmitted. A bridge linking LANs with different MAC sublayers may have to reject frames which are too large to forward. Thus bridges operate best when all the networks in the bridged environment are compatible at the MAC sublayer.

Bridges do not implement any form of flow control, and are therefore subject to congestion. Since a bridge has only a finite buffer space, it may lose frames which it cannot store. The recovery of these lost frames is the responsibility of the end stations.

1.2.3 Routers

Routers are internetworking devices that operate at the network layer. Use of a router requires a transmitting station to address its frame to the router rather than to the desired destination station. Information in the layer three header of the frame specifies the final end station. The router uses this information to forward the frame to the appropriate network. Thus, as opposed to most bridging algorithms, the use of a router is not transparent to the end stations in the network.

Connecting local area networks with routers requires that all nodes in the internetworked environment execute the same layer three protocol, whereas the layer three protocol used in a bridged local area network is transparent to the bridges and may therefore be different for different stations. However, routers are better able to connect networks with incompatible MAC layers than bridges are. For example, protocols typically used with routers allow a

router to fragment frames that are too large to forward into several smaller frames to be forwarded.

Since bridges do not operate at the LLC or network layers, a bridge should in general perform better than a router.

1.3 Scope of the Thesis

This thesis describes the development of a local area network bridge designed to link together two IEEE 802.5 token ring networks. The work performed for this project includes both the hardware and software design for the bridge and a study of the performance characteristics of the bridge. This bridge design is intended to be used as part of a larger project for which the goal is to develop and implement protocols for the 802.5 token ring network for transmitting data, digitized voice, and digitized still-frame video pictures. The transmission of a digitized voice must be performed in such a manner that the end-to-end delay and end-to-end jitter between consecutive frames are small enough to not introduce noticeable distortion in the played-back voice.

The 802.5 token ring network was originally developed by IBM and was eventually chosen as one of the three LAN protocols standardized by the IEEE 802 committee. This standard is described in Chapter 2.

Chapter 3 outlines the important design factors considered in the development of this bridge. These include factors that should be considered in any bridge design, as well as factors specific to bridges designed for 802.5 networks.

Two alternative routing algorithms have been proposed to allow arbitrary topologies of bridges and LANs to be constructed. The hardware designed for this thesis is capable of operating with either algorithm. Both algorithms are discussed in Chapter 4.

The hardware design for this bridge uses an IBM PC as a base for which 802.5 network interface expansion cards were designed and constructed. The network interface hardware design is presented in Chapter 5.

The bridge software that executes on the IBM PC is described in Chapter 6. This software implements the source routing bridge algorithm described in Chapter 4.

2. DESCRIPTION OF THE IEEE 802.5 TOKEN RING NETWORK

This chapter describes the IEEE 802.5 token ring local area network. Section 2.1 provides an overview of the network. The frame format encoding is described in Section 2.2. Section 2.3 describes the ring priority mechanism, and Section 2.4 describes the ring maintenance procedures.

2.1 Introduction to the 802.5 Token Ring Network

The IEEE 802.5 token ring network (ANSI/IEEE 1985) consists of a series of stations serially connected with shielded twisted-pair wiring in a star shaped ring topology (Figure 2.1). Logically, the nodes in this form of network are wired in a ring configuration, with each node having connections to its successor and predecessor. Physically, however, the nodes are connected to wiring concentrators by a cable containing two twisted pairs, one for transmit data and one for receive data. The wiring concentrators are themselves connected by point-to-point links in a ring topology. The wiring concentrators contain relays used to insert or remove stations from the logical ring. These relays are energized by current from the stations. If the cable to a station breaks or the station is powered down, the loss of drive current will release the relay and bypass the station.

Information is transmitted sequentially from station to station at a data rate of 4 Mbps. A transmitting station transfers a frame onto the ring, where it circulates from one station to the

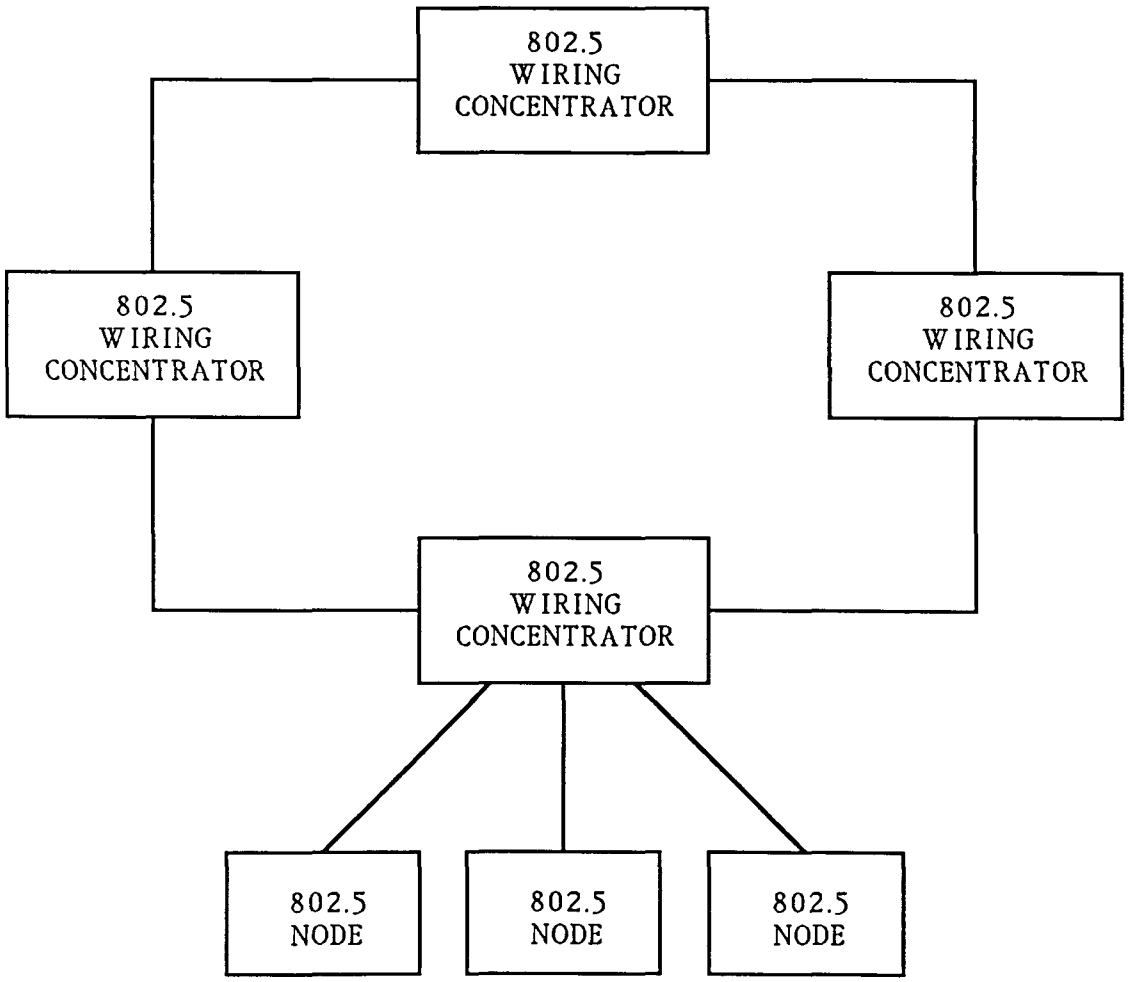


Figure 2.1 IEEE 802.5 network topology

next. All stations but the transmitting station repeat each bit as it passes. In addition to repeating the incoming bit pattern, the addressed destination station copies the passing information into local memory and sets the value of two acknowledgement bits at the tail of the frame to indicate receipt. Finally, the station that transmitted the frame removes it from the ring.

A station gains the right to transmit when it detects a unique signaling pattern, known as a token, passing on the medium. As explained in Section 2.3, tokens contain a priority field that can be used to give higher priority access to stations that must transmit frames with real-time timing constraints. Upon detecting a token with the appropriate priority, a station with data to transmit may capture the token by modifying it to a start-of-frame sequence and appending a frame. A transmitting station may continue to transmit frames until a token holding timer at the transmitting station expires or the station has no more frames queued to transmit with the appropriate priority. When a station has completed its information transfer and has seen that the start of its first transmitted frame has completely circulated the ring, it transmits a new token onto the ring to provide other stations with the opportunity to gain access to the ring.

Error detection and recovery in the token ring are performed by a station known as the active monitor. Every station has the capability to serve as the active monitor; a negotiation phase is performed to elect the monitor upon initialization of the ring or upon power down or failure of the current monitor. The active monitor is responsible for performing such functions as recovering from a lost token, stripping persistently circulating frames, and removing partial or garbled frames. The operation of the active monitor is explained further in Section 2.4.

2.2 Frame Encoding

Two basic frame formats are used on the token ring network: information frames and token frames. Both formats are made up of a sequence of octets (bytes), each octet containing eight symbols. A symbol can encode one of the following four values:

0 = binary zero

1 = binary one

J = non-data J

K = non-data K

The binary zero and binary one symbols are used to encode standard binary data. The J and K symbols provide an out-of-band signaling mechanism to uniquely encode the starting and ending delimiters of a frame or token. A fill pattern consisting of an arbitrary number of zero or one symbols is used to separate the trailing end of one frame or token from the starting end of the next. A station can distinguish fill bits from non-fill bits by keeping track of the starting and ending delimiters as they pass.

The 802.5 token frame format is shown in Figure 2.2 and the information frame format in Figure 2.3. The fields of these frames are described in the following subsections.

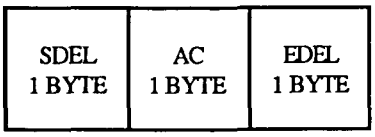


Figure 2.2 IEEE 802.5 token frame format

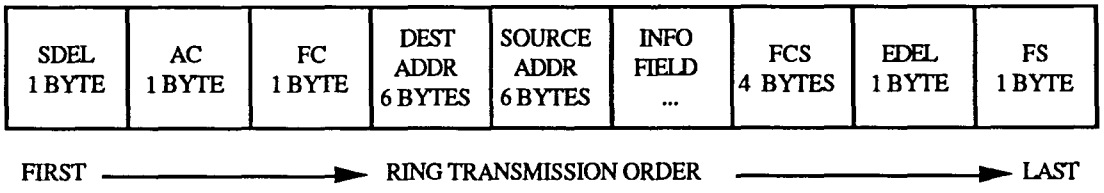


Figure 2.3 IEEE 802.5 information frame format

2.2.1 Starting delimiter

The starting delimiter field is one byte in length as shown in Figure 2.4. A station uses this field to synchronize to the bit stream of the frame. The use of the J and K symbols in the starting delimiter uniquely distinguishes it from the other fields in the frame for synchroniztion purposes.

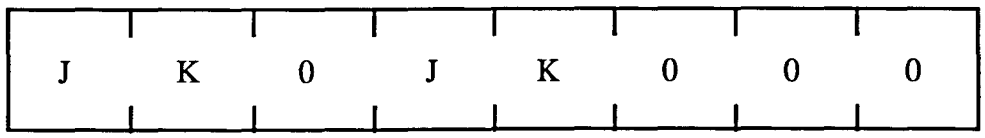


Figure 2.4 IEEE 802.5 starting delimiter

2.2.2 Access control byte

The access control byte is shown in Figure 2.5. This byte is used to control access to the network. The bits marked PPP and RRR are used to implement the token priority mechanism. Up to eight priority levels, encoded as 0 - 7, are available for transmission, priority level 7 being the highest. The priority of the current token is encoded in the PPP bits. The RRR bits are used by stations to request that a future token be released at the requested priority level. The priority scheme is explained further in Section 2.3.

The T bit is used to distinguish a token from an information frame. This bit is encoded as a zero in a token and as a one in an information frame. A station waiting to transmit can flip

this bit in a passing token to a one to change the token into an information frame and append a queued frame which has a priority greater than or equal to the token priority encoded in the PPP bits.

Finally, the M bit is used by the active monitor station to delete frames which should be removed from the ring. When a station transmits a frame it sets the M bit to zero. As the frame passes the active monitor, the monitor flips this bit to a one. Since the transmitting station should delete the frame from the network once it has traversed the ring, the active monitor should never see the M bit set to a one. However, if the transmitting station malfunctions or is powered down before stripping its frame, the frame will return to the monitor with the M bit set. The monitor will see that this bit is a one and remove the frame.

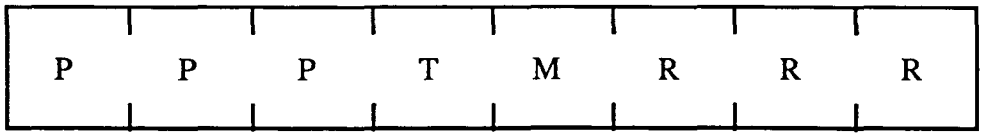


Figure 2.5 IEEE 802.5 access control byte

2.2.3 Frame control byte

The frame control byte is shown in Figure 2.6. The two most significant bits are used to distinguish between information frames destined only for the MAC sublayer (encoded as 00) and information frames that should be passed to the LLC sublayer (encoded as 01). The other two possible encodings of these bits are reserved for future use.

There are six MAC sublayer frames defined. These are used for ring maintenance purposes. For these frames, the Z bits designate which type of maintenance frame is present in the data field. For the LLC frames three of the Z bits can be used to encode the priority of the frame from the source LLC entity to the destination LLC entity. Note that the PPP bits of the access control byte do not necessarily designate the frame priority since a station may transmit a

frame using a token for which the priority level is less than the priority of the frame.

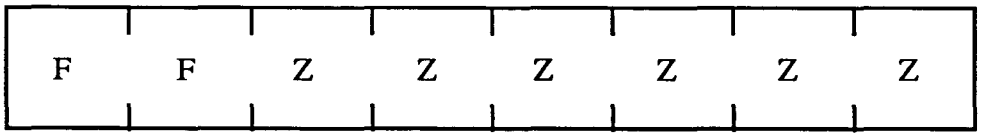


Figure 2.6 IEEE 802.5 frame control byte

2.2.4 Destination address

The destination address field is six bytes long as shown in Figure 2.7. These six bytes encode the destination address for the frame. The two upper bits of the most significant byte have a special purpose. Bit seven of the byte is used to distinguish frames addressed to an individual station from frames addressed to a group of stations. A value of zero indicates an individual destination address and a value of one indicates a group destination address. Bit six of the most significant byte is used to distinguish addresses assigned globally from addresses assigned by the local network administrator. A value of zero indicates a globally administered address and a value of one indicates a locally administered address. This address encoding scheme has been adopted for all 802 local area network MAC layer standards.

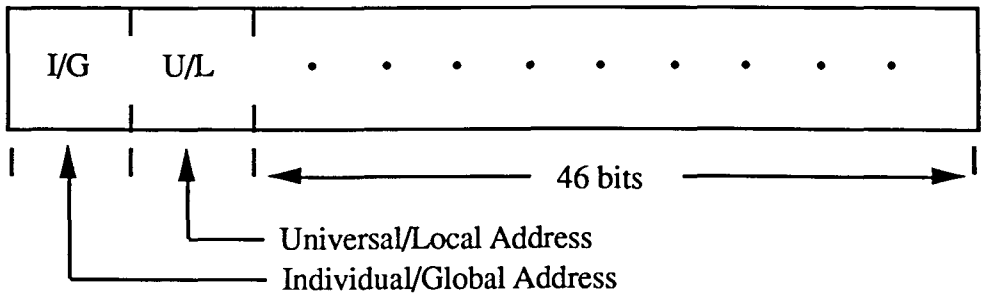


Figure 2.7 IEEE 802.5 address format

2.2.5 Source address

The source address field is six bytes long as shown in Figure 2.7. The source address is encoded with the same format as that used to encode the destination address; however, the source address must always be an individual station address.

2.2.6 Information field

The information field contains the LLC data or MAC data for the frame. No maximum frame length is specified by the 802.5 standard; however in practice the length is limited by the maximum time a transmitting station may hold the token. This value is a system configurable parameter.

2.2.7 Frame check sequence

The frame check sequence field is a 32-bit checksum used by the destination station to detect transmission errors. The checksum covers the frame control field through the frame check sequence fields inclusive.

2.2.8 Ending delimiter

The ending delimiter is one byte long as shown in Figure 2.8. This byte encodes the ending delimiter pattern for the frame. As in the starting delimiter, the use of the J and K symbols in the field uniquely distinguishes it from other fields in the frame. The I bit is used to distinguish intermediate frames of a multiple frame transmission from the last frame transmitted by a station while holding the token. The transmitting station checks for this bit as it recieves the frames it has transmitted after they have circulated the ring. It stops stripping the circulated frames when it sees a frame with the I bit indicating the last frame. The E bit is used to indicate a frame check sequence error to the transmitting station. This bit is initially

transmitted as a zero; if any station on the ring determines that the passing frame has a frame check sequence error, it sets this bit to a one as it passes.

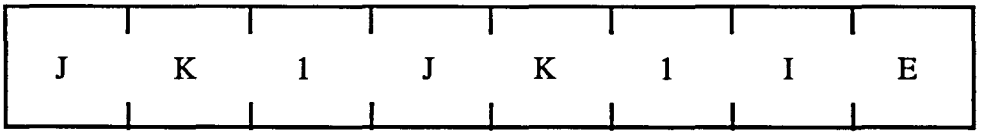


Figure 2.8 IEEE 802.5 ending delimiter

2.2.9 Frame status

The frame status field is one byte long as shown in Figure 2.9. This byte is used to indicate to the transmitting station whether the destination station exists on the ring and whether or not it was able to copy the frame. The byte is initialized to all zeros by the transmitting station. The destination station writes a one into the A (Address Recognized) bits to indicate to the transmitting station that the destination address was recognized. If the frame passed the destination station without a frame check sequence error and the destination station was able to copy the frame into local memory, it sets the C (Copied) bits. Since the frame status field is not covered by the frame check sequence, the A and C bits are duplicated to provide a means of error detection.

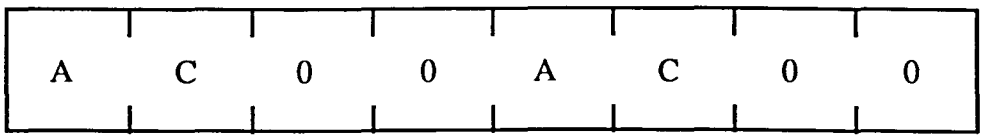


Figure 2.9 IEEE 802.5 frame status byte

2.3 Priority Operation

The access control byte of each frame contains a three bit priority field (PPP) which encodes the priority of the frame or token being circulated. The priority of the initial token on the ring is always 000, the lowest priority. A station can capture a passing token to transmit information frames as long as these frames have a queued priority greater than or equal to the priority of the passing token. These frames are transmitted with the priority bits in the access control byte set to the priority of the captured token, regardless of the queued priorities of the frames.

A station with frames queued to transmit can request that the next token released by the current transmitting station by transmitted at a certain priority. This request is accomplished by setting the RRR bits of the access control byte of a passing token or frame to the requested priority level. However, if a priority greater than the desired priority has already been requested in the RRR bits, a station is not allowed to lower the requested priority level.

When the station currently holding the token completes its transmission, it releases a new token with the priority set to the maximum of the requested priority level in the RRR bits of the frame it last circulated, the highest priority in any frame still waiting in its transmit queue, and the current ring priority. If this value is greater than the current ring priority, the release of the new token will raise the ring priority. In this case the transmitting station must remember that it has raised the priority, and must store the values of the old priority and the new priority on a stack. When the station sees a token return at this new priority, it examines the request bits of the access control field to determine whether to raise, maintain, or lower the current priority. If the priority should be lowered, the station pops the old priority from the top of the stack and releases a new token with this priority. If the priority should be raised, the station releases a new token with the proper priority and pushes the corresponding values onto the top of its

stack.

2.4 Ring Maintenance

The station designated as the active monitor continuously monitors the ring. The active monitor is responsible for regenerating lost tokens and stripping frames that continue to circulate. If a period of time elapses with no token being detected, the monitor assumes that the token has been lost and issues a new token. If a frame passes the active monitor, the monitor sets the M bit in the access control byte. If the frame returns to the monitor with this bit set, the monitor knows that the frame has been seen before, and strips it from the ring. This may occur either because a transmission error garbled the frame so that the sending station did not recognize it and therefore did not strip it, or because of a failure or power-down of the sending station.

All other stations on the ring are known as passive monitors. The passive monitors check the operation of the active monitor. If the active monitor fails, the passive monitors use a procedure to determine which station should take over as the active monitor.

3. BRIDGE DESIGN OBJECTIVES

The inclusion of one or more bridges in the route between two end stations should be as transparent as possible to the end stations. For example, bridges should strive to maintain the high throughput, low delay, and low error rate characteristics of the LANs which they connect. To achieve this transparency, bridge designs must incorporate certain features. Section 3.1 outlines the characteristics that an ideal bridge or network of bridges should posses in order to minimize the impact on the communicating end stations, and lists the features that a bridge should possess in order to achieve these characteristics. Section 3.2 compares the bridge designed for this research against these goals.

3.1 Desired Characteristics for Bridges

3.1.1 Minimal frame loss

Due to the absence of any flow control mechanism, a bridge may temporarily run out of buffer space during times of peak activity. When this occurs, the bridge will not be able to capture frames that should be forwarded. In order to minimize this possibility, a bridge should have a large enough buffer space to handle heavy loads. Support for packet filtering by the bridge hardware rather than relying only on software filtering can also help minimize the number of lost frames.

3.1.2 Sequential delivery

Most, if not all, local area networks deliver frames between nodes in the order in which they were sent. If the communication protocols used by the stations on a LAN expect such sequential delivery of frames, then bridges must preserve this characteristic. Note that for LANs such as the 802.5 network that implement a priority scheme, the requirement for sequential deliver may prevent bridges from reordering the frames queued for retransmission based on the priorities of the queued frames.

3.1.3 Minimal delay

Stations on a LAN typically expect a low end-to-end delay when transferring frames. To retain this property, a bridge should attempt to minimize the processing overhead for forwarded frames. Packet filtering by the bridge hardware can be used to reduce the interrupt load on the bridge's processor, thus allowing the processor to respond quicker to those frames that are to be forwarded.

3.1.4 Source address overlay

A bridge must be able to transfer a frame with a source address other than its own in order that forwarded frames retain the original source address rather than the address of the bridge. This capability is not possible with all commercial VLSI local area network controllers.

3.1.5 End-to-end frame check sequence coverage

A bridge must not introduce any undetectable errors in frames which are retransmitted. Allowing a bridge with the ability to retransmit frames using the original frame check sequence rather than regenerating this field provides a method for the end stations to detect whether the

data have been corrupted while stored in the bridge's memory. This capability is also not possible with all existing VLSI LAN controllers.

3.1.6 High priority transmission

A bridge should have the ability to transmit frames with a higher priority than other nodes. This helps to reduce congestion at the bridge and to minimize end-to-end delay of frames that traverse one or more bridges. The 802.5 network supports up to eight levels of priority; however, current implementations use only the lowest four levels (zero to three). It is desirable that 802.5 bridges be able to transmit frames at any priority. It may also be desirable to determine the priority of each frame at transmission time based upon current network load.

3.1.7 Setting of the frame status field

A transmitting station may rely on the values of the address recognized and frame copied bits of the 802.5 frame status field. Therefore, a bridge for 802.5 networks must be able to set these bits if it has determined that it should forward the frame. This must be done in real-time as these bits pass through the bridge while circulating the ring.

3.1.8 Access for network management purposes

For large local area networks it is important to be able to control the operation of a bridge from another node for network management purposes. The IEEE has recommended five categories on network management functions that an ideal bridge should support (Currie 1988). These five categories include configuration management (e.g., resetting, enabling, or disabling a bridge or a port on a bridge), fault management (e.g., detection and reporting of failures), performance management (e.g., collection and reporting of performance and traffic statistics), security management (e.g., access control and encryption), and accounting

management (e.g., recording of any costs for the use of a bridge's resources).

3.1.9 Minimal traffic overhead

The algorithms used by bridges should minimize traffic by avoiding complex bridge-to-bridge protocols used to compute routes.

3.1.10 Allow arbitrary topologies

Most bridging algorithms work to eliminate the existence of multiple routes between any given pair of hosts in a bridged network. These bridging algorithms exchange protocol messages among the bridges to detect the existence of multiple routes in the network. If such routes are found a negotiation phase is executed to disable certain bridges or ports within a bridge so that a single route exists between any two nodes. This is necessary to ensure that duplicate frames aren't delivered to a destination station. Two alternative bridge routing protocols have been developed for this purpose. These protocols are discussed in Chapter 4.

3.1.11 Quick response to changes in the environment

A bridged local area network environment is in general more dynamic than that of a single LAN due to the possibility of bridge of LAN failures or the movement of hosts between LAN segments. The routing protocol used by bridges should be responsive to these changes. For example, if a bridge fails during operation, a topology update algorithm should be executed by the other bridges to enable another path to be established between the LANs connected by the failed bridge is such a route exists.

3.2 Accomplishment of Design Goals

The bridge designed for this thesis attempts to achieve most of the goals listed in Section 3.1. Frame filtering is performed in hardware to minimize processor interrupts and lower the frequency of lost frames. Extra frame buffering RAM is included on the LAN adapters to also help reduce the frequency of lost frames. The bridge software maintains sequential delivery of frames forwarded by the bridge. The 802.5 VLSI chip set used by the bridge provides the ability for source address overlay and maintains the original frame check sequence for forwarding frames. The chip set is also capable of using all eight priorities defined by the 802.5 standard, and the bridge software forwards frames at a high priority to help minimize congestion. The chip set also sets the frame status bits on frames which it captures for forwarding purposes.

Although no remote network management capability was designed into the bridge, error reporting and performance measurements are printed to the local screen of the IBM-PC serving as the bridge. Finally, the source routing protocol, one of the two bridge routing protocols described in Chapter 4, was implemented for the bridge. This protocol uses minimal overhead to compute routes and allows arbitrary topologies of bridges and LANs to be constructed. Dynamic reconfiguration is the responsibility of the end stations under the source routing protocol.

4. BRIDGE ROUTING PROTOCOLS

Due to the growing demand for methods to interconnect local area network segments together, the IEEE 802 committee has been developing standards for local area network bridges. Two alternative bridge routing protocols have been proposed. These two protocols are referred to as the "spanning tree algorithm" and the "source routing algorithm". The IEEE 802.1 committee has evaluated both bridging schemes and has selected the spanning tree algorithms as the single standard for interconnecting 802 local area networks together (Backes 1988). However, the 802.1 committee has allowed the 802.5 Token Ring committee to extend its standard to include source routing bridges. This chapter examines and compares the two algorithms. The spanning tree algorithm is described in Section 4.1 and the source routing algorithm in Section 4.2. Finally, Section 4.3 compares and contrasts the two algorithms.

4.1 Spanning Tree Algorithm

The spanning tree algorithm (Perlman 1985, Backes 1988) is designed to allow bridges to be transparently integrated into existing local area network environments. Transparent integration means that no modifications are required to existing stations to allow these stations to communicate with each other through a bridge. Thus transparent bridges are backwards compatible with existing 802 compliant implementations.

Under the spanning tree algorithm, bridges perform three basic operations:

1. Frame forwarding
2. Learning of station addresses
3. Participating in an algorithm to resolve topology loops.

These functions are described in the following subsections.

4.1.1 Frame forwarding

In the spanning tree algorithm bridges operate in a "promiscuous mode", receiving all frames transmitted on each LAN segment to which they connect. Each bridge maintains a forwarding data base by observing and recording the source address of all frames received from each of its ports. The data base consists of a list of node addresses and a bridge port for each address. When a frame is received, the bridge compares the destination address to the information contained in the forwarding data base. If the destination address is found in the forwarding data base, and the port identifier for that entry is the same as the identifier for the port on which the frame was received, the frame is not forwarded to any bridge port. If the port identifiers are different, the frame is forwarded to the bridge port contained in the data base. Finally, if the destination address is not found in the data base, the frame is transmitted on all bridge ports except the port from which it was received. Likewise, frames addressed to a group address are also transmitted on all bridge ports except the one from which they were received. The upper box of Figure 4.1 presents the flow chart for the frame forwarding process.

4.1.2 Learning of station addresses

Initially the forwarding data base is empty. To add an entry to the data base, the bridge compares the source address of each frame received without error against the entries currently in the data base. If the source address is not found, it is added along with the identifier of the

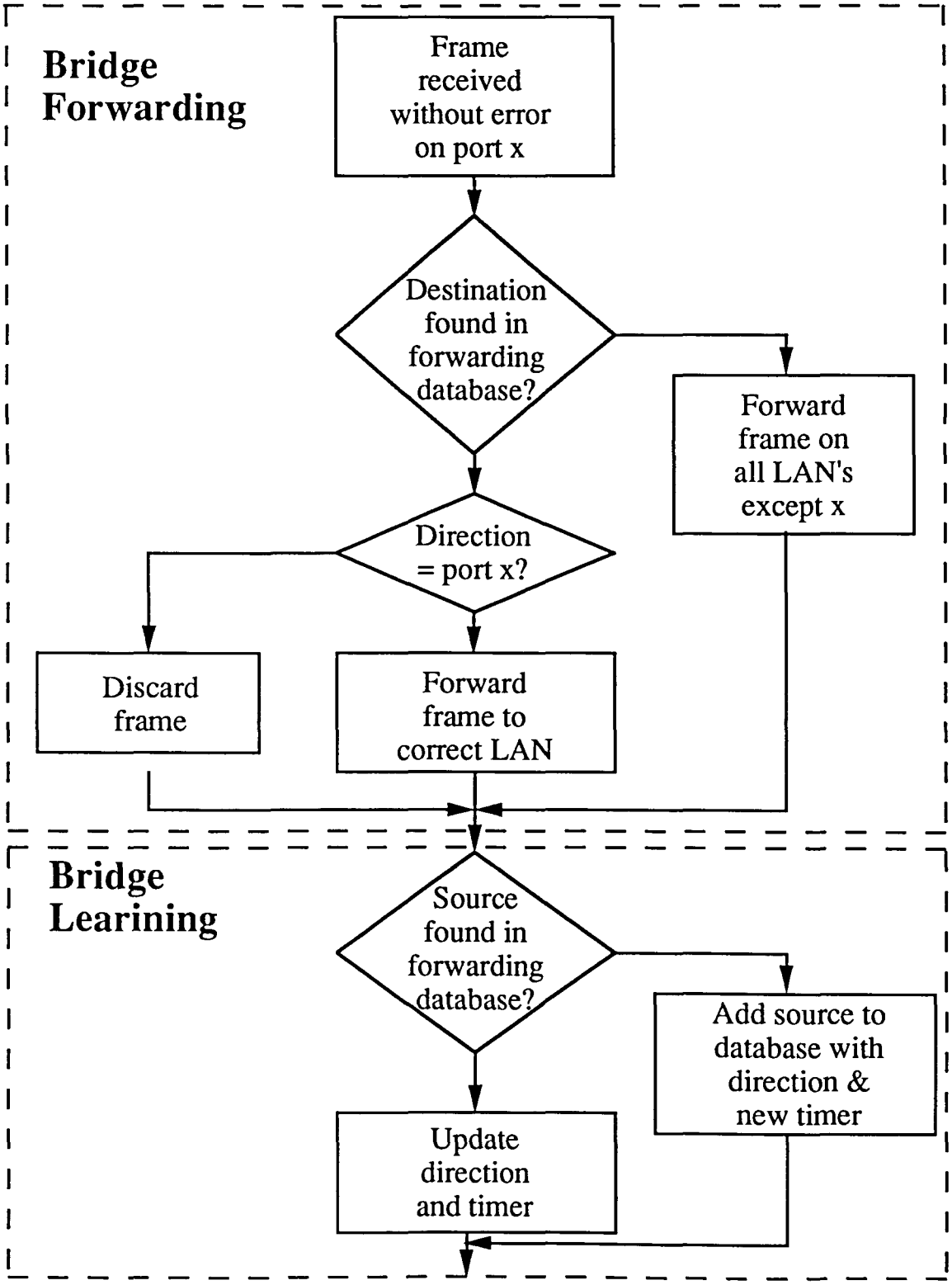


Figure 4.1 Spanning tree algorithm flow chart

port on which the frame was received. A timer value for this data base entry is reset to indicate that this is a current entry. The timer is used to keep track of the age of each entry. Entries are removed from the data base when their time field indicates that the data has not been updated for a specified period of time. Removing old entries allows the data base to adjust to changes such as the movement of a station from one LAN segment to another, or the power down or failure of a station.

If the source address of a received frame is found to exist in the data base and the port identifier does not match the entry in the data base, the port identifier in the data base is changed to reflect this new information. If the port identifier is the same it is not changed. In either case the timer field for the entry is reset. The lower box of the flow chart in Figure 4.1 illustrates the bridge learning process.

4.1.3 Spanning tree calculation

Difficulties can arise in bridged LANs if the network is configured such that multiple routes exist between any given pair of hosts. In such cases a frame could take each possible route to the destination, thereby generating duplicates. Additionally, frames would be continuously circulating in the system. Therefore, in addition to providing frame relaying functions, bridges must also overcome multiple route difficulties.

The spanning tree algorithm is designed to overcome these difficulties by pruning an arbitrarily connected graph of networks and bridges to a single tree structure which spans all of the LAN segments. The nodes of the graph consist of the bridges and LAN segments in the system. An edge connects a bridge node to a LAN segment node if and only if the bridge is directly connected to the LAN segment and is in a state to forward frames to or from the LAN segment. The algorithm is implemented by a complex protocol in which bridges exchange protocol messages using a connectionless mode data transfer. The exchange of messages

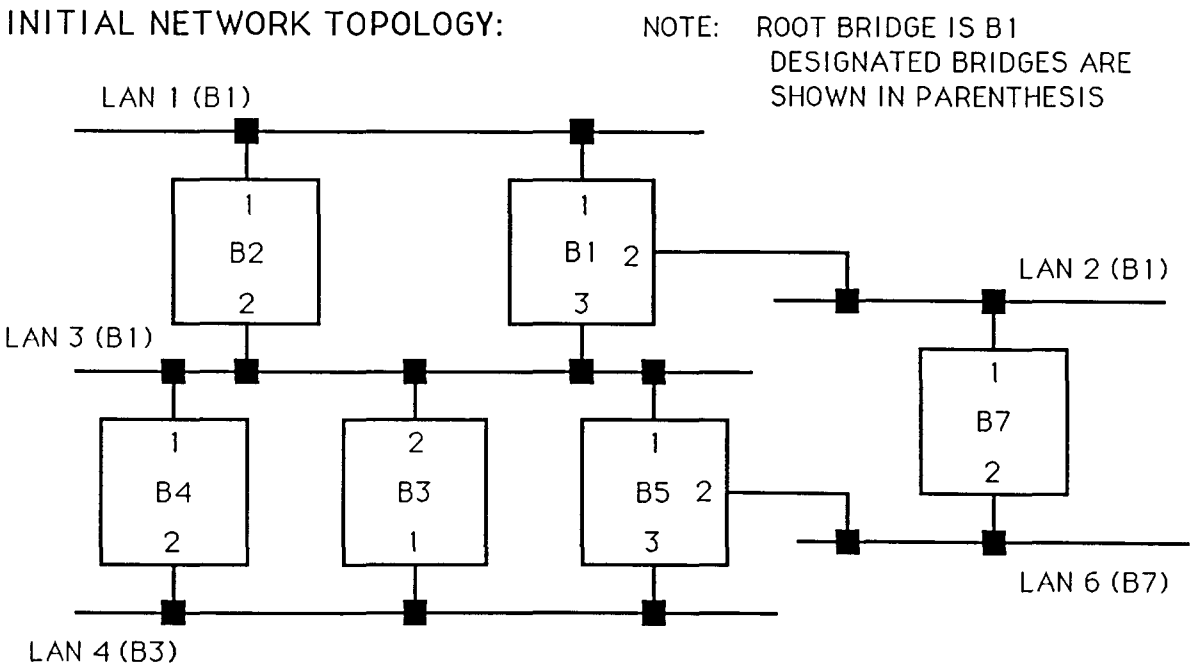
results in certain bridges disabling some or all of their ports from forwarding frames so that the resulting topology is a spanning tree. The algorithm is briefly described below.

Initially, each bridge and each port within a bridge are assigned a unique identifier. The bridges exchange messages using multicast addressing to a well-known bridge group address in order to determine the bridge with the lowest numbered identifier. This bridge becomes the root bridge. Then each bridge selects a port through which a least cost path to the root is found. This port is called the "root port". Next, a unique bridge is selected for each LAN. This bridge is termed the "designated bridge", and the port connection to the LAN is termed the designated port. The designated bridge for a LAN is selected to be the bridge with the lowest numbered bridge identifier among all bridges on that LAN. Finally, each bridge puts its root port and all bridge ports to LANs for which it is the designated bridge into a forwarding state. The other bridge ports are put into a blocked state. The end result is the desired spanning tree topology.

An example network topology and the result after the spanning tree algorithm has been performed are shown in Figure 4.2. In the resulting network each LAN has a unique path up-tree towards the root through the designated bridge for that LAN. A designated bridge will forward frame traveling up-tree onto its root port. When a frame has traveled up-tree as far as is necessary, it will then travel down-tree along a different set of branches going in through the root port of the proper bridge and out through that bridge's bridge port until it has reached the destination LAN.

4.2 Source Routing Algorithm

This section examines an alternative to the spanning tree bridge algorithm known as the source routing algorithm. The source routing algorithm has been developed by the IEEE 802.5



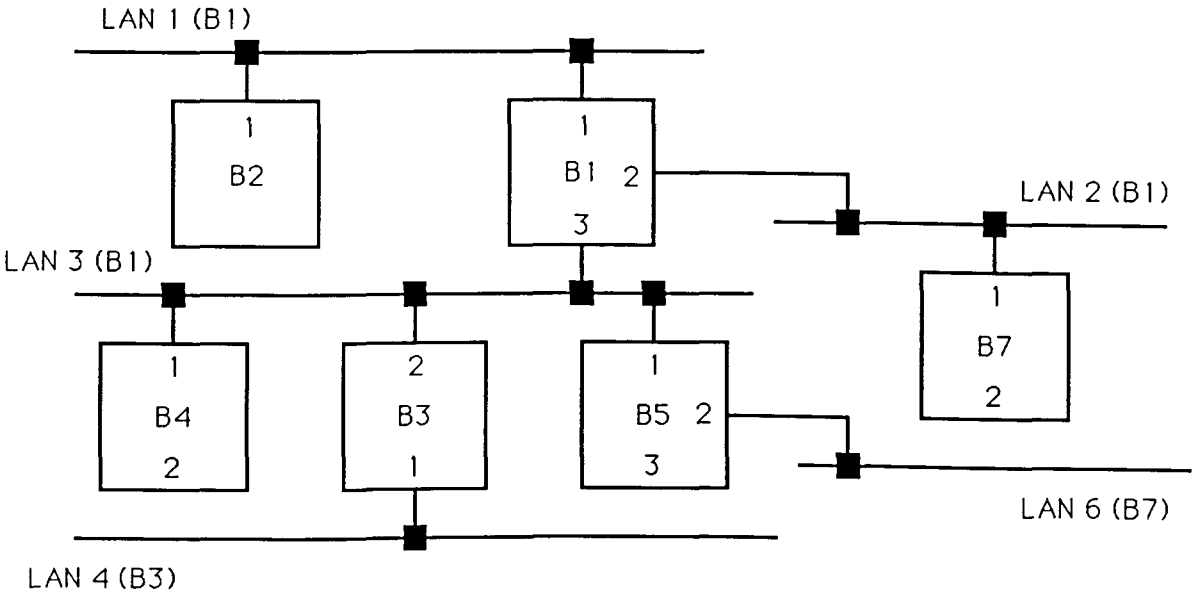


Figure 4.2 Example of spanning tree algorithm

committee to allow stations to communicate across multiple token ring networks (Dixon and Pitt 1988). The frame forwarding operation of the algorithm is explained in Section 4.2.1. The route discovery process used with source routing is explained in Section 4.2.2.

4.2.1 Frame forwarding under source routing

In the source routing algorithm, a transmitting station is responsible for specifying the route that a frame should follow to reach the destination station. In order to determine such a route, the transmitting station must first execute a route discovery algorithm as explained in Section 4.2.2. If this algorithm determines that the destination station does not reside on the same LAN segment, the transmitting station inserts a routing information field immediately following the source address in every frame sent to the destination station. The bridges connecting the LAN segments use this routing information field to forward the frames across the correct LAN segments.

The presence of a routing information field is indicated by a one in the most significant bit of the 6-byte source address in the MAC layer header. In the format for LAN addresses specified by the IEEE, this bit is defined to distinguish between an individual station address and a group address. However, the 802.5 specification states that a transmitting station must always use its individual station address, rather than a group address, for the source address field. Therefore, the source routing protocol redefines this bit to indicate the presence or absence of a routing information field.

The format of the routing information field is illustrated in Figure 4.3. The field can vary in size from 4 to 18 bytes. The first two bytes are control bytes. The encoding of these control bytes is shown in Figure 4.4. Their use will be described later. The remaining bytes consist of a sequence of route designators that describe the route that the frame should traverse in order to reach the addressed destination station.

802.5 ROUTING INFORMATION FIELD:

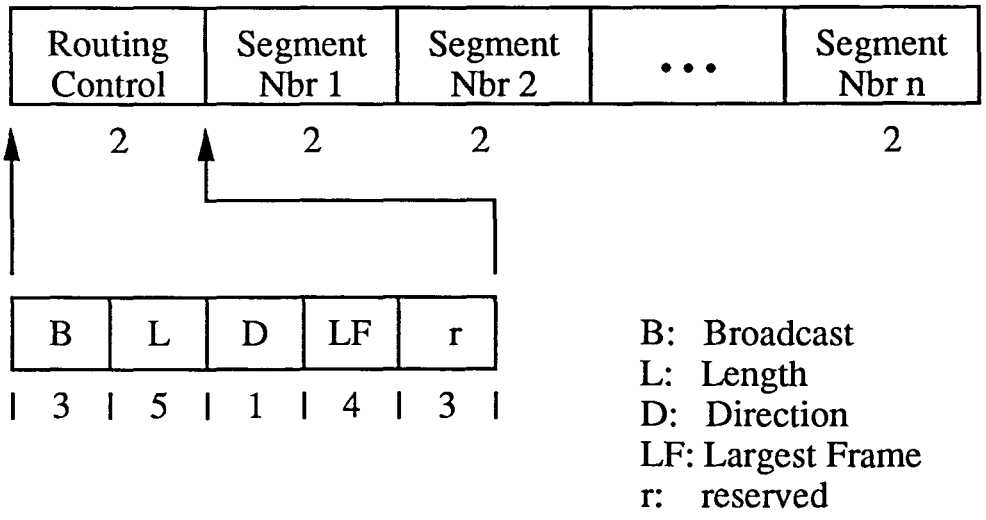
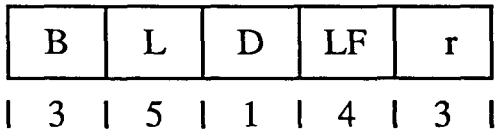


Figure 4.3 Routing information field for source routing algoritm

802.5 ROUTING INFORMATION FIELD CONTROL BYTE ENCODING:



B: Broadcast
L: Length
D: Direction
LF: Largest Frame
r: reserved

B: Broadcast bits
000 = Non broadcast
100 = All-routes broadcast

L: Length bit
- indicates the length of the routing information field

D: Direction bit
- indcates whether the frame is traveling from the originating station to the target or the other way around

r: Reserved bits

SEGMENT NUMBER:

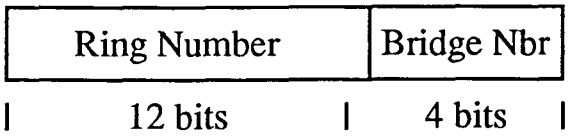


Figure 4.4 Routing information field control byte encoding

Each route designator consists of a LAN segment number and a bridge number. Every LAN segment in the bridged LAN environment is assigned a 12-bit number and every bridge is assigned a 4-bit number. The LAN segment numbers must be unique for each LAN segment. A bridge number needs only be unique among all bridges that link the same two LAN segments together. A LAN segment number followed by a bridge number constitutes a route designator, and thus each route designator is 16-bits long. The sequence of these route designators denotes the order of LAN segments and bridges that a frame must traverse to reach the addressed destination. A routing information field can contain up to eight route designators. The number actually present is encoded in the control bytes of the routing information field.

A source routing bridge examines the most-significant bit in the source address field of each frame to determine if the frame should be forwarded to another LAN. If this bit is a one, the bridge scans the contents of the routing information field to determine if the segment numbers for the segments to which it is connected and its own bridge number are grouped together and in the correct order. If so, the bridge forwards the frame to the desired destination ring. If not, the bridge discards the frame.

4.2.2 Route discovery

In order to communicate with a station on another ring, a sending station must first execute a route discovery algorithm. In the route discovery algorithm, a station wishing to determine a route to another station first transmits a route locating frame. The route locating frame is addressed to the desired destination station, but contains a routing information field for which the route designators are empty. The broadcast bits in the control bytes of the routing information field are set to indicate that this is a frame that should be forwarded by each bridge on all outgoing ports. The largest frame bits in the control byte are initially set to '111'.

Each time the route locating frame is received by a bridge, the bridge scans the route

designator list to determine if a route designator corresponding to its LAN segment number and bridge number are already present in the list. This check is to verify that the frame has not previously been forwarded by that bridge. If the correct route designator does not appear in the list, the bridge will add it to the end of the list and forward the frame on all ports except the one on which it was received. If the maximum frame size that can be forwarded by the bridge is smaller than that indicated by the LF bits in the control field, the bridge will modify these bits to indicate the maximum length frame which it can forward before retransmitting the route locating frame.

When the frame has reached the destination station, the destination responds by sending a frame back to the original sender with the broadcast bits in the control bytes set to non-broadcast, the direction bit inverted, and the identical segment number list which was received. Once the original station receives the response frame it has a route to the destination.

4.3 Routing Protocol Comparison

The main advantage of the spanning tree algorithm over the source routing algorithm is that the spanning tree algorithm is completely transparent to the end stations. Source routing requires functionality in the end stations, and there are existing 802 products that do not provide this functionality. However source routing does offer several advantages over the spanning tree algorithm. For example, source routing nodes may be able to choose a route based upon considerations such as security or hop count. It is also possible to use parallel bridges in source routing to gain an increase in performance by allowing different source stations to select different routes. Note that it is not in general a good idea for a single station to alternate between parallel bridges since this approach does not guarantee sequential delivery. Finally, source routing bridges are more able to perform some level of frame filtering in

hardware since the filtering decision for source based on comparison of a single bit in the passing frame.

5. BRIDGE HARDWARE DESIGN

The token ring to token ring bridge described in this thesis was designed to use a dedicated IBM Personal Computer (IBM-PC) as a hardware base. Two token ring bridge adapter boards were constructed and placed into expansion slots on the IBM-PC backplane. Software was written to execute on the IBM-PC to control these two boards. Each of the circuit boards provides an interface between the PC and a single token-ring network as shown in Figure 5.1. The two circuit boards are identical in design, and the implementation could easily be generalized to an "n-port" bridge by adding additional network interface boards to the PC.

The bridge design is based on the Texas Instruments TMS380 chip set. This chip set is described in Section 5.1. The schematic circuit diagrams are illustrated in Figure 5.2. These schematics are described in Sections 5.2 through 5.4. In the description of the schematic diagrams, a component is identified by its location on the circuit board. A two-dimensional grid is used for this identification, where each row is labeled with a number from one through seven, and each column with a letter from 'A' through 'I'.

5.1 TMS380 Chip Set

The bridge's network interface boards are based in the Texas Instruments TMS380 chip set (Texas Instruments 1986). This chip set consists of a family of five chips that together

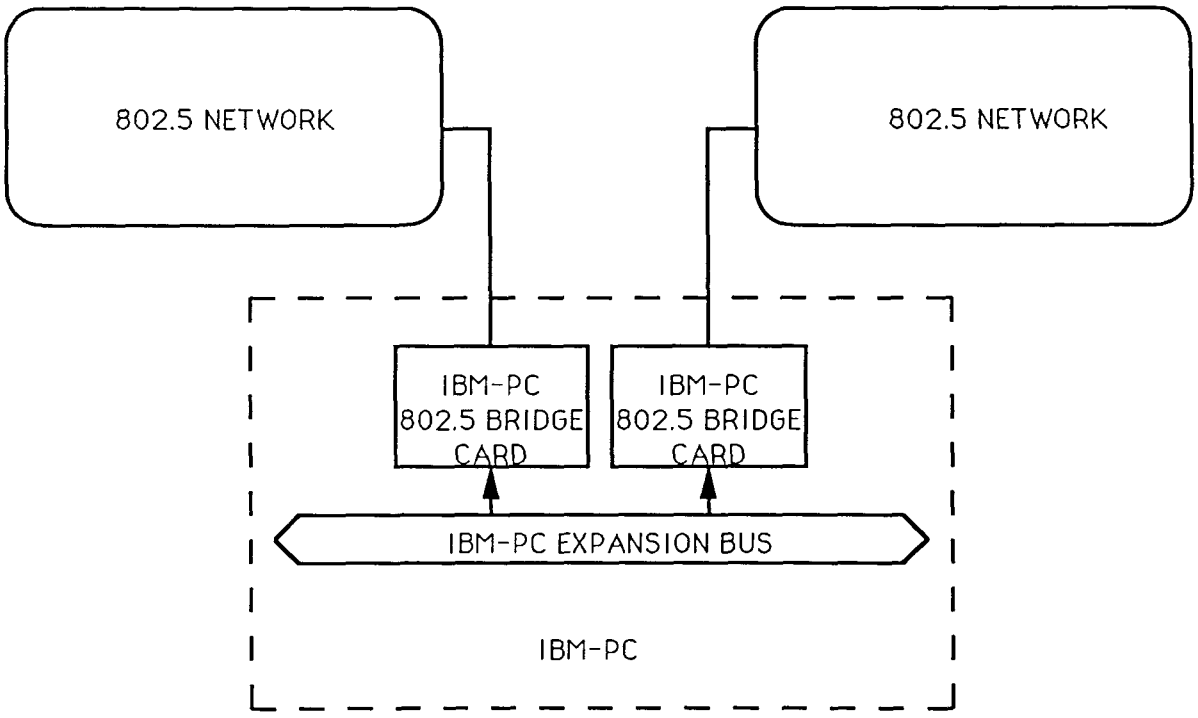


Figure 5.1 Token ring bridge configuration

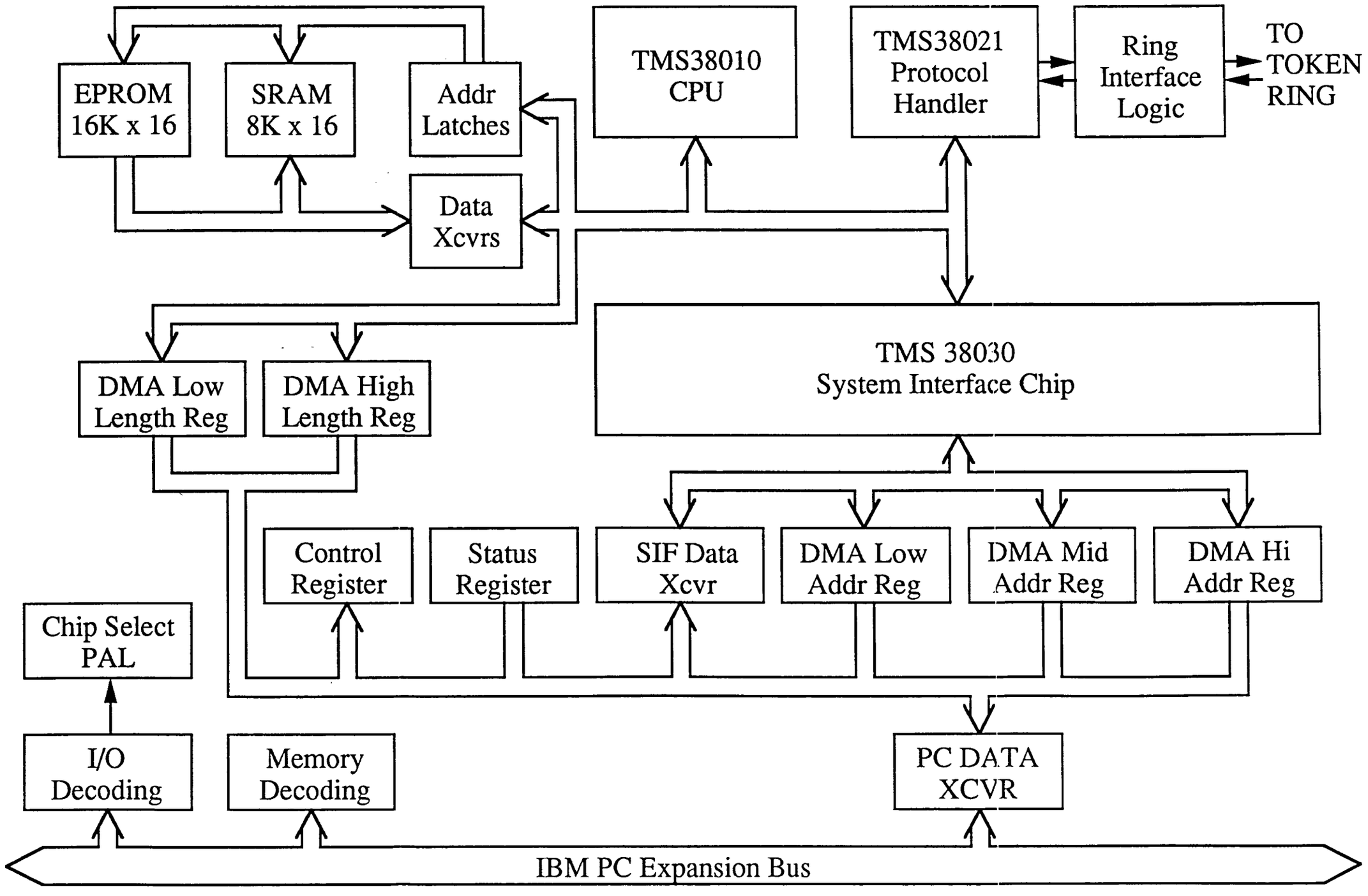


Figure 5.2 Bridge schematics

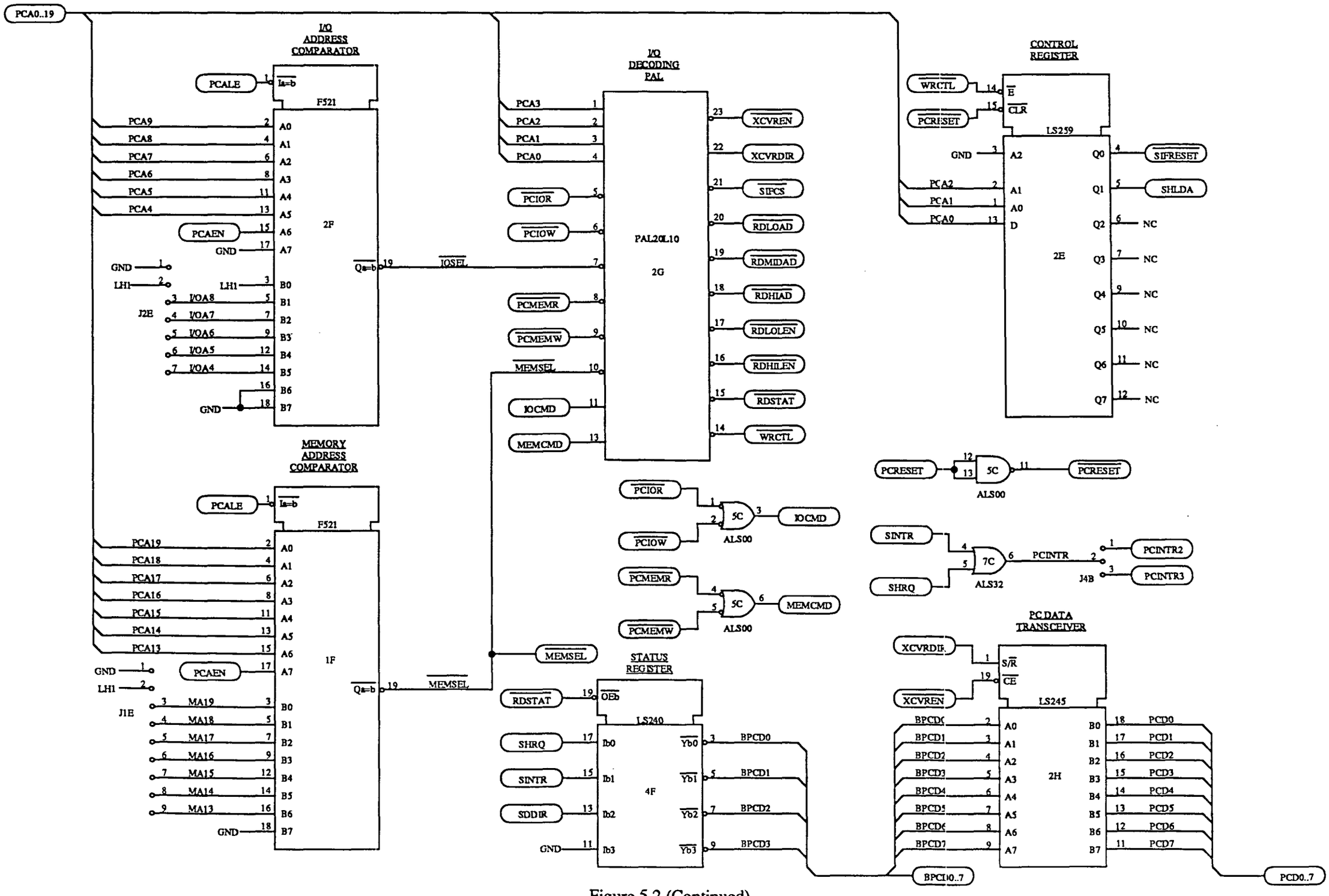


Figure 5.2 (Continued)

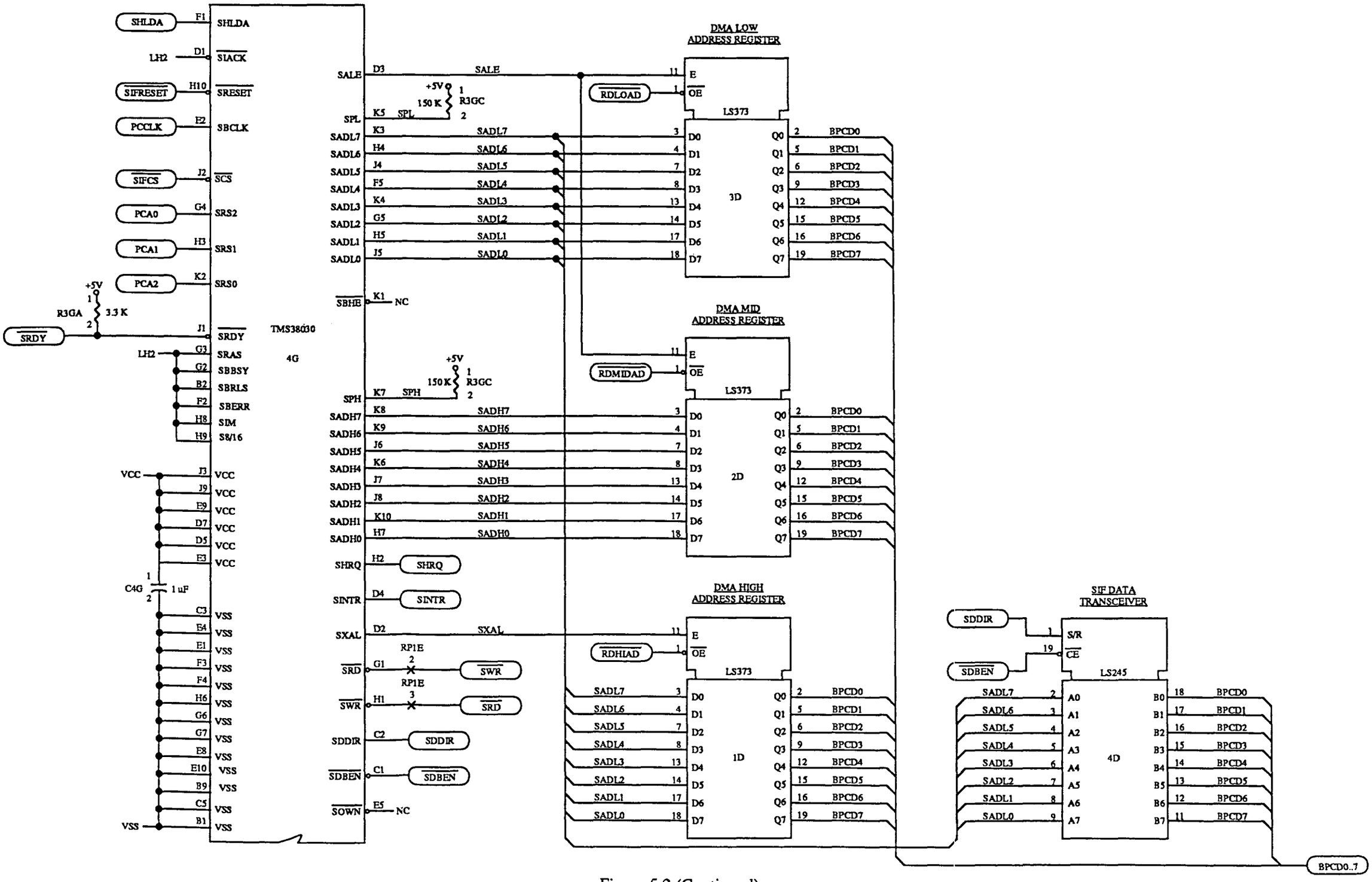


Figure 5.2 (Continued)

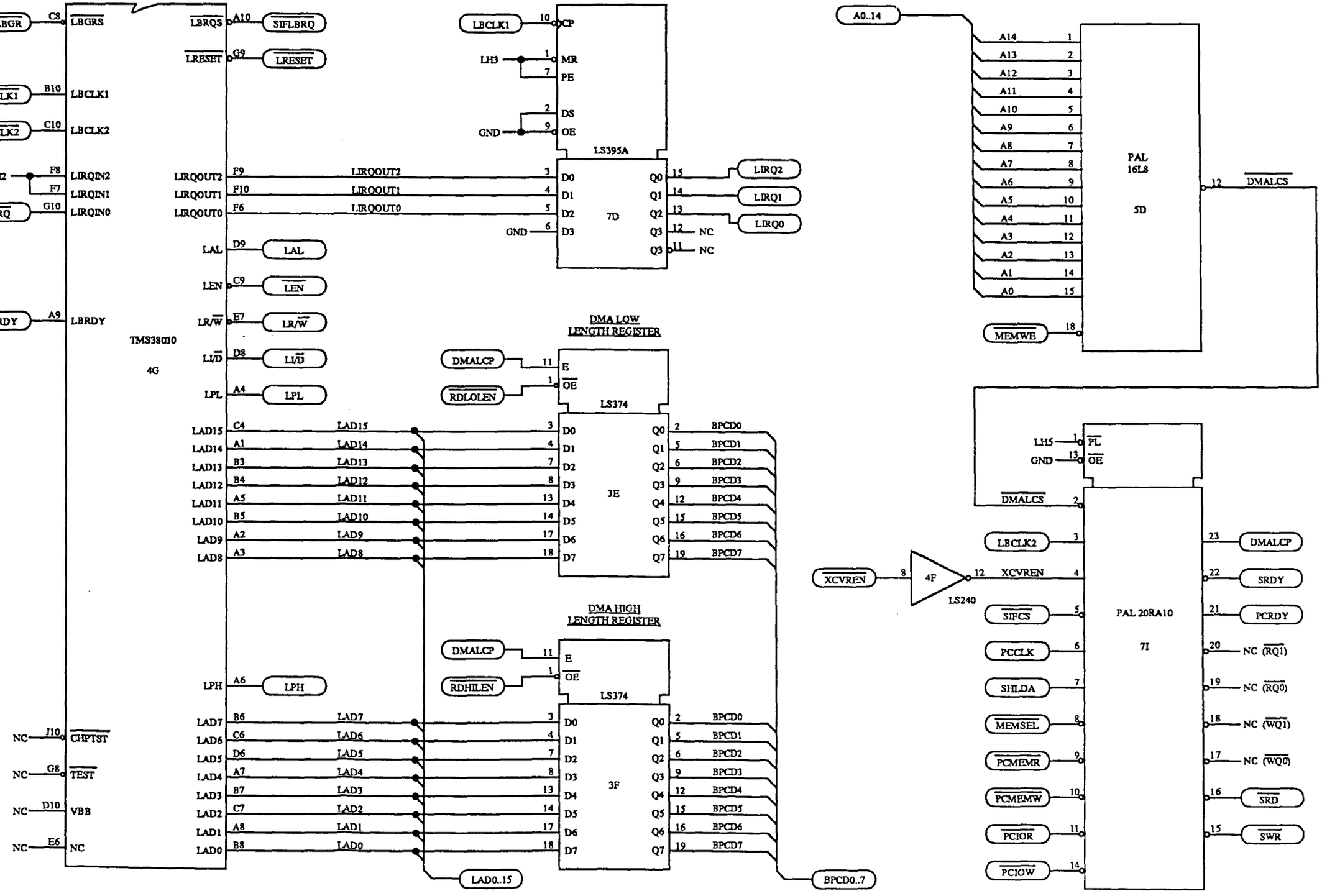


Figure 5.2 (Continued)

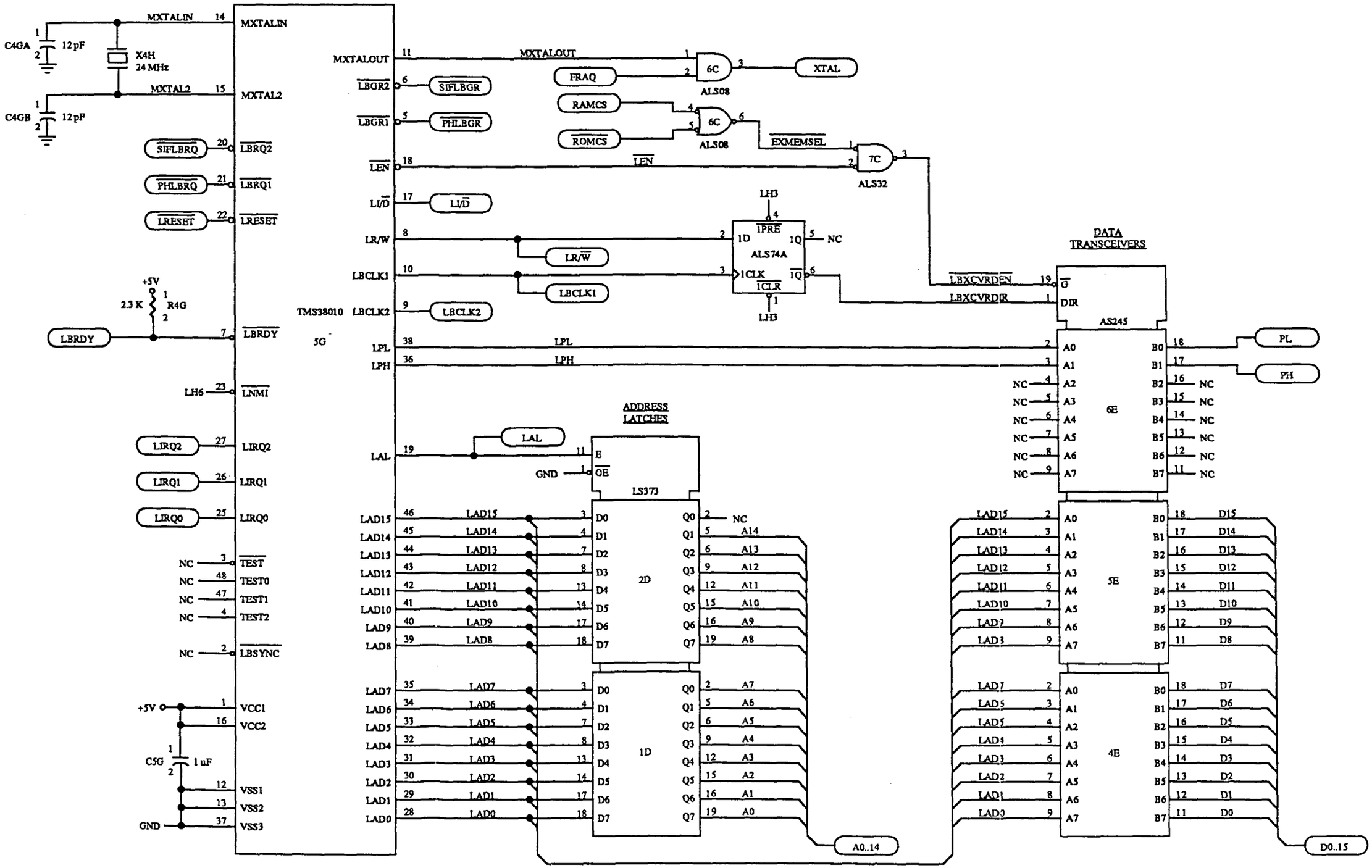


Figure 5.2 (Continued)

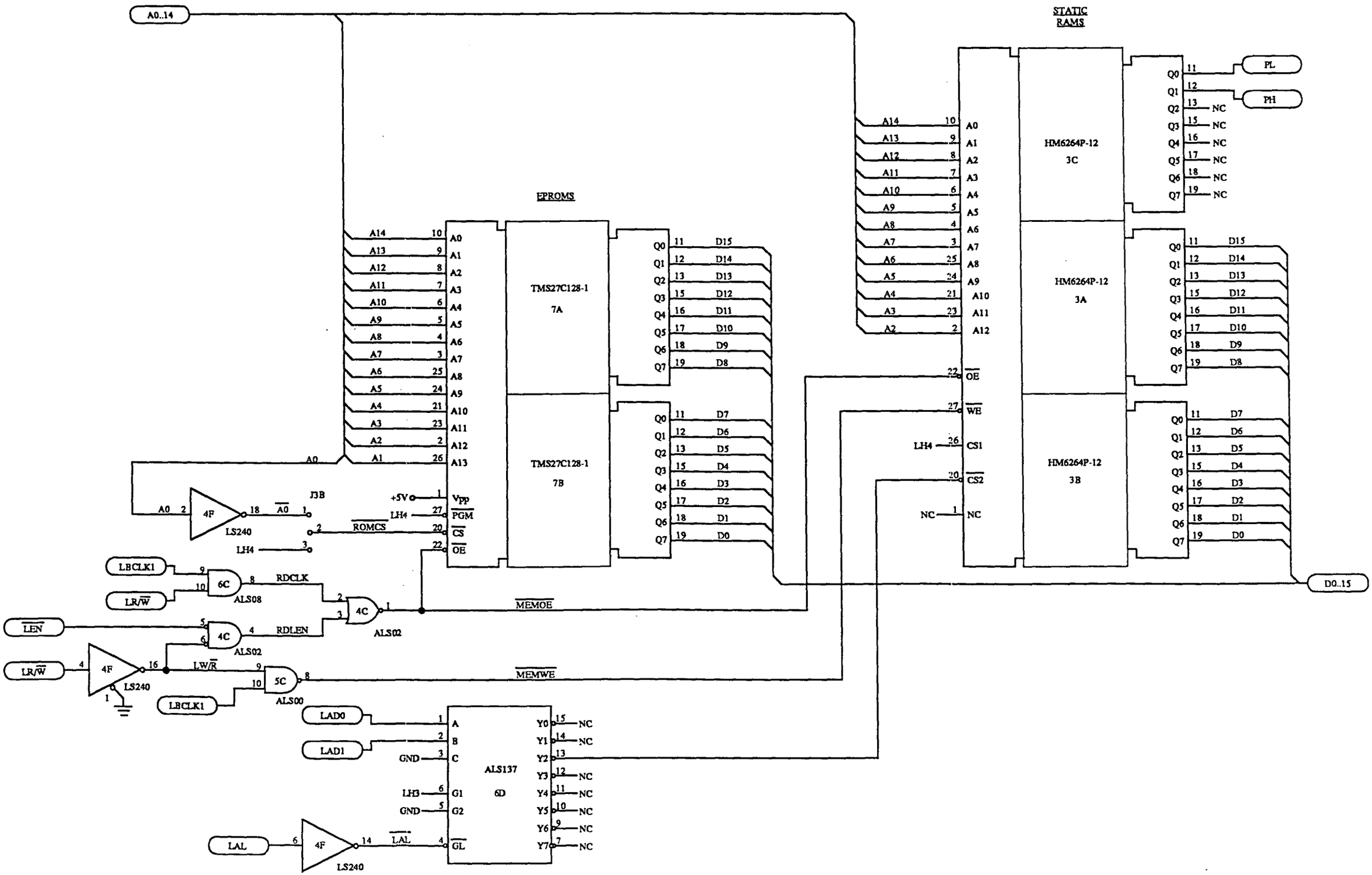


Figure 5.2 (Continued)

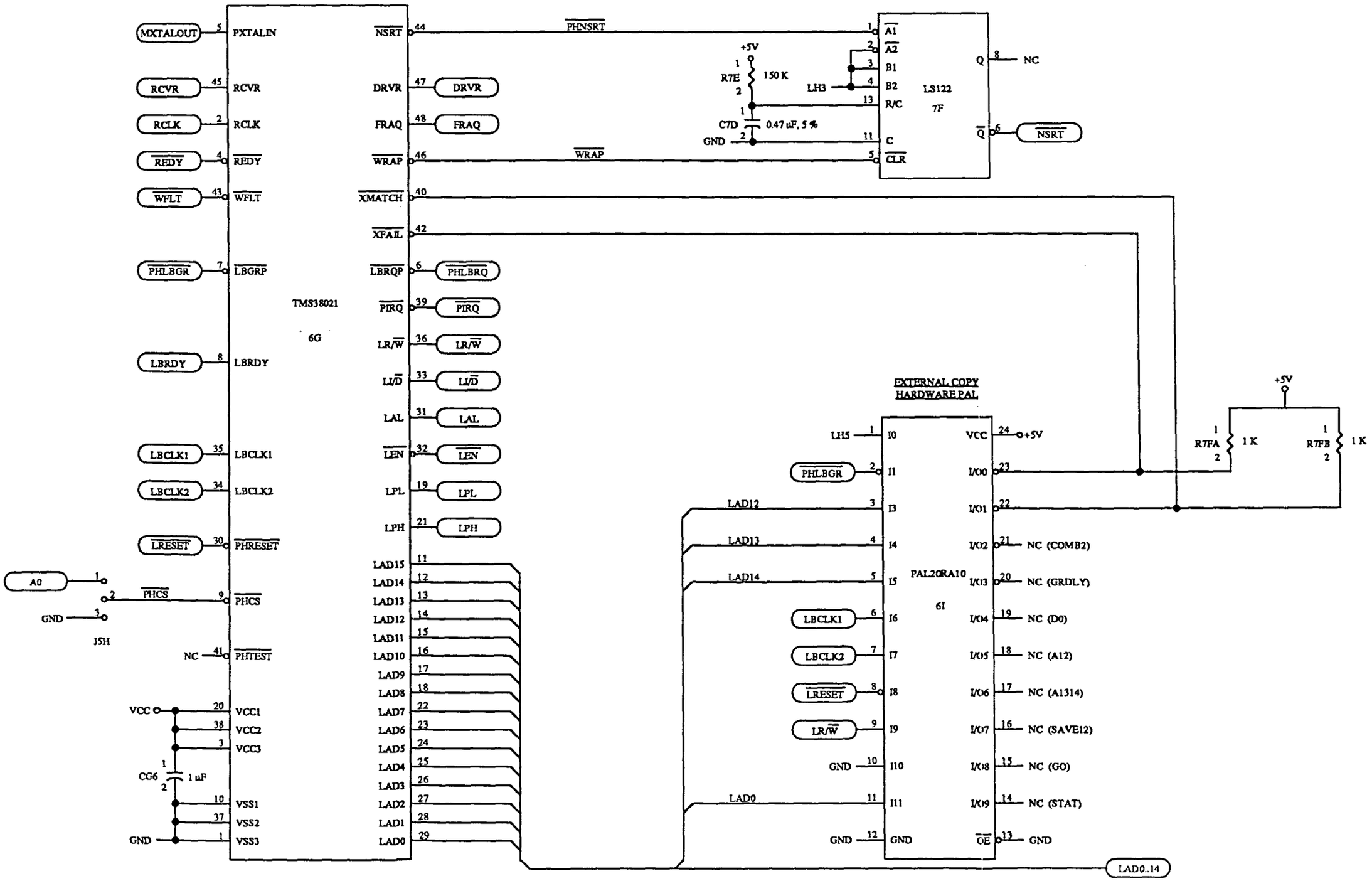


Figure 5.2 (Continued)

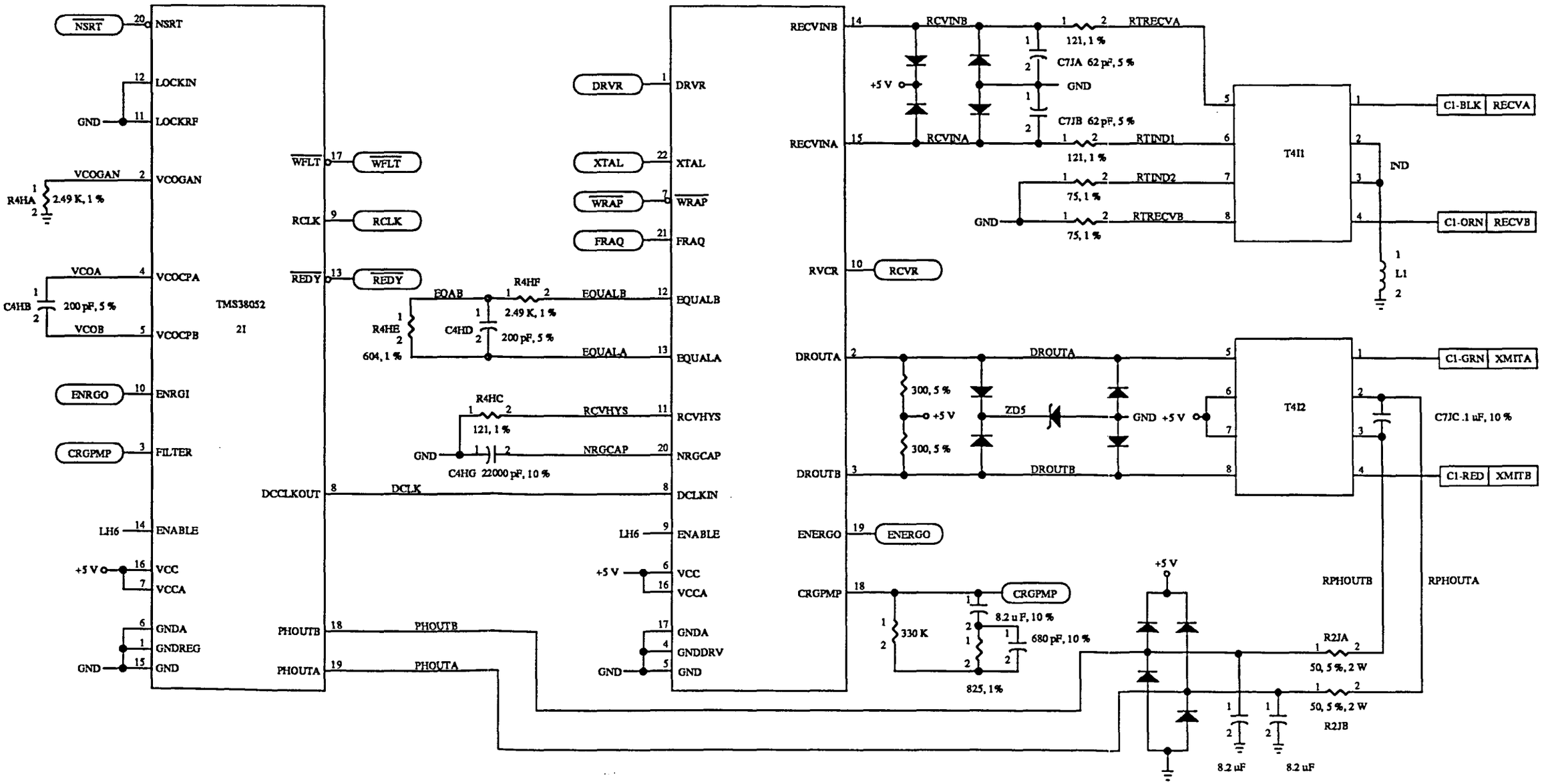


Figure 5.2 (Continued)

provide most of the functions necessary to design an intelligent IEEE 802.5 network interface board. A high level block diagram of the network interface designed for this thesis is shown on the first page of the schematic diagrams in Figure 5.2. This block diagram illustrates the relationship among the different chips in the chip set. The functions of each chip in the TMS380 family are described in the following subsections.

5.1.1 TMS 38010 Communications Processor

The TMS 38010 is a 16-bit microprocessor especially designed for data communications applications. The 38010 can address 32K 16-bit words of memory using a 16-bit address bus. 1408 words of RAM are included internal to the chip. This on-chip RAM buffers the frames being transmitted and received. An additional 8K words of external RAM have been provided for the network interface boards constructed for this bridge to increase the frame buffer space. The on-chip RAM and external RAM are both parity protected. The parity generation and checking logic is included in the 38010 microprocessor.

The LAN adapter bus is designed to allow DMA transfers between buffer RAM and the network and between buffer RAM and the system bus (IBM-PC bus). The TMS 38010 includes the arbitration logic for accesses to the LAN adapter bus for these DMA operations.

5.1.2 TMS 38021 Token Ring Protocol Handler

The TMS 38021 Protocol Handler performs hardware based protocol functions for the token ring network. The 38021 implements bit encoding and decoding for the token ring; frame address recognition; and the state machines to capture tokens, transmit and receive frames, manage the adapter buffer RAM and provide token priority controls.

In addition, the 38021 contains 16K of on-chip ROM that contains the adapter firmware executed by the communications processor. This ROM is preprogrammed by Texas

Instruments with the adapter firmware. This frees the designer from the necessity of writing the low-level (MAC layer) software to drive the 38010 and ensures MAC layer compatibility among all implementations based on the TMS380 chip set.

The 38021 is also responsible for transferring frames between the adapter memory and the TMS 38051/38052 ring interface chips. The 38021 contains two transmit and two receive DMA channels dedicated for these transfers. For frame reception, the 38021 also contains the logic to compare the destination address of passing frames against the local address. Frames whose destination address matches the local address are automatically captured if buffer space is available. In addition, input pins on the 38021 allow external logic to signal the 38021 to capture a passing frame whose destination address doesn't match the local address. This function is used on the bridge design to capture all passing frames which contain a routing information field as indicated by a one in the most significant bit of the source address.

5.1.3 TMS 38030 System Interface Chip

The TMS 38030 System Interface Chip provides the interface between the LAN adapter bus and the host system bus. The host system bus used in this design is the 8-bit IBM-PC bus. Generally, the TMS 38030 is capable of becoming a bus master on the host system bus in order to transfer command and status blocks and token ring frames between host memory and LAN adapter memory. However, the IBM-PC bus used in this design does not allow external bus masters. Therefore, the circuitry used in this design contains logic that creates the appearance of host bus DMA capability to the 38030. However, these "pseudo DMA" cycles are actually controlled by software that executes as an interrupt service routine on the IBM-PC whenever the 38030 requests access to the host system bus. The interrupt service routine performs a block memory move operation to transfer the data to or from the LAN adapter. Hardware handshaking is provided to both the 8088 microprocessor in the IBM-PC and the

38030 System Interface Chip to synchronize these transfers.

5.1.4 TMS 38051 and 38052 Ring Interface

The TMS 38051 Ring Interface Transceiver and the TMS 38052 Ring Interface controller collectively provide the interface between the LAN adapter and the 802.5 token ring network. These chips contain the logic to provide the ring clock when the LAN adapter is serving as the active monitor, the phase lock loop circuitry for clock recovery, data detection, and phase alignment, the drive current to energize the relay in the wiring concentrator, a loop-back path for diagnostic testing, and error detection of wire faults.

5.2 IBM-PC Interface Logic

The LAN adapter's interface logic to the IBM-PC bus is shown on pages two through four of the schematic diagrams in Figure 5.2. The PC accesses the network adapter through both an I/O mapped interface and a memory mapped interface.

5.2.1 I/O interface

The I/O interface is mapped into a contiguous block of 16 I/O addresses. The base address of this block is configured through the jumpers at location J2E. The F521 comparator at location 2F compares address bits four through nine on the PC bus against the address range selected by these jumpers, and generates the IOSEL signal if the addresses match. The 20L10 PAL at location 2G further decodes the lower address bits to generate one of eight chip select signals if the IOSEL signal is active.

The SIFCS chip select signal is generated by the 20L10 for accesses to the 38030 System Interface Chip. The 38030 contains four 16 bit registers used for transferring command and

status information to the 38010. These four registers are accessed by the PC as eight 8-bit registers which reside at the bottom eight I/O addresses in the LAN adapter's I/O address range.

The RDLOAD, RDMIDAD, and RDHIAD chip select signals are used to generate read strobes to a set of three registers that hold the address for DMA requests from the 38030 to the IBM-PC. These three registers are implemented by the LS373 transparent latches at locations 3D, 2D, and 1D. The registers hold the low, middle, and high bytes of the 24 bit DMA address, respectively. The register values are set by latching the address bus from the 38030 during DMA requests to the PC. The DMA request is mapped into an interrupt request to the PC, and the PC's interrupt service routine reads the starting DMA address from these registers before performing a block memory move to transfer the data.

The RDLOLEN and RDHILEN strobes are used to read the low byte and high byte of the DMA transfer length during the DMA interrupt service routine. These registers reside outside the 38030 in the LS374's at locations 3E and 3F, but identical registers are present in the 38030. When the 38010 writes the DMA transfer length to the 38030's internal registers, the external registers also record this length. The 16L8 PAL at location 5D generates the write signal to these external registers whenever the 38010 writes to the address of the register's internal to the 38030. When the 38030 generates a DMA request to the PC, the DMA interrupt service routine reads the transfer length from these registers before performing the block memory move.

The RDSTAT signal is used to provide a read strobe to a 3-bit status register. The bits in this status register are read by the IBM-PC during the interrupt service routine for interrupts generated by the LAN adapter. The bits encode the type of interrupt (DMA read, DMA write, or 38030 event interrupt).

Finally, the WRCTL signal is used to enable a addressable latch used as a control register

by the PC to control operation of the LAN adapter. The PC is able to reset the LAN adapter through this control register, as well as generate a HOLD ACKNOWLEDGE signal to the 38030 during DMA requests by the 38030.

5.2.2 Memory interface

The memory interface from the IBM-PC to the LAN adapter is mapped into an 8K block of contiguous memory addresses. The starting address of this block is set by the jumpers at location 1E. The F521 comparator at location 1F generates the MEMSEL signal whenever the address on bits 13 through 19 of the PC address bus matches the settings of the jumpers. The MEMSEL signal is not used to directly read memory on the LAN adapter, but rather is generated during the block reads or writes used by the PC to implement the pseudo DMA transfers. The MEMSEL signal is used by the 20RA10 PAL at location 7I to perform synchronization handshaking during these transfers.

5.3 LAN Adapter Local Bus Logic

The LAN adapter local bus logic is shown on pages five and six of the schematic diagrams in Figure 5.2. The 38010 Communications Processor, 38021 Protocol Handler, and 38030 System Interface can each serve as a bus master on the local bus. Normally, the 38010 operates as bus master to access instructions and data from external memory, but the 38010 releases the bus during during DMA local bus requests by the 38021 or 38030.

The 38010, 38021 and 38030 connect to the local bus through a common set of 16 multiplexed address/data lines and several control lines. The AS373 transparent latches at locations 5F and 6F latch the 16-bit address for the current bus cycle from the multiplexed bus to produce a non-multiplexed address that remains stable for the complete bus cycle. The

AS245 data transceivers at locations 4E, 5E, and 6E provide added current buffering for external memory accesses. The buffered data bus contains 16 data bits and 2 parity bits.

The EPROMs at locations 7A and 7B contain the bridge protocol code for the 38010. The 38021 contains code in ROM that allows the 38010 to serve as an 802.5 network interface node; however, for the LAN adapter to serve as an 802.5 bridge the external EPROMs are necessary. Parity checking by the 38010 is disabled for EPROM accesses, so a third EPROM is not necessary.

The 8K x 8 static RAM chips at locations 3A, 3B, and 3C provide the 38010 with extra frame buffering memory. Parity generation and checking is performed for accesses to this RAM, so three RAM chips are used.

5.4 Network Interface Logic

The network interface logic is shown on pages seven and eight of the schematics in Figure 5.2. The 38021 is programmed to capture all frames transmitted on the network; however if the XFAIL pin on the 38021 is driven low during a frame capture, the 38021 aborts the capture and frees the frame buffer. The 20RA10 PAL at location 6I monitors the header of the incoming frames as they are stored to local memory on the LAN adapter. If an incoming frame does not contain a one in the most significant bit of the source address, indicating the presence of a routing information field, the PAL generates the XFAIL signal to notify the 38021 that the capture should be aborted.

The 38051 and 38052 on page eight of the schematics provide the interface to the token ring cable. The discrete components on this page perform signal conditioning and line termination functions.

6. BRIDGE SOFTWARE DESIGN

This Chapter describes the design of the bridge software that executes on the IBM-PC to control the two LAN adapter boards. Section 6.1 describes the software initialization process. After the system has been initialized, a small multitasking kernel is started to support multiple processes. The kernel design is described in Section 6.2. A timer module is also implemented to allow processes to delay for specified periods of time. This module is described in Section 6.3. Section 6.4 describes the interrupt service routines for LAN adapter interrupts, and Section 6.5 describes the processes used to control the bridge. All of the bridge software is written in the C programming language.

6.1 Initialization

The LAN adapter board firmware contains built-in self test programs that execute upon power-up. These programs test internal adapter RAM, perform a checksum on the adapter ROM, test the communication processor, and then perform a loop-back test on the ring. The status of these tests is made available the the host system. During the initialization procedure, the bridge software on the IBM-PC reads and reports the status of these tests on the PC's screen for both LAN adapter cards. If the tests complete successfully the PC initializes both LAN adapter cards and then initializes the multi-tasking kernel. Finally, five processes are started: a read and write process for each adapter and a management process that periodically prints bridge statistics to the local screen. At this point, initialization is complete, and the

bridge is able to forward frames or participate in the route identification process.

6.2 Multitasking Kernel Design

A simple kernel was implemented to allow a multitasking software design for the bridge. A multitasking approach is desirable due to the event driven nature of the bridge's operation. The multitasking kernel is a derived from a similar kernel described in Dlugosz (1988). The kernel executes as an MS-DOS program, but supports multiple processes in a non-preemptive fashion. Each process is allocated a separate stack. When a process wishes to relinquish the CPU, it performs a "yield" call to the kernel. The kernel save the state of the calling process on the process's stack and then selects another process from the ready list to be dispatched.

The kernel design supports several forms of interprocess communication. Since all processes share a common address space, shared memory is implied in the design. In addition, the kernel has provisions for semaphores and simple 1-word messages that may be exchanged between processes.

6.3 Timer Module

Certain operations in the bridge's software design require that a function be invoked after a specified delay. Such a feature can be used to provide software timeouts, or to periodically activate a background task. To provide this ability, a real-time clock was added to the bridge design, and a timer module was written to control this clock. The real-time clock was implemented on an expansion card containing an Intel 8254 clock/timer chip. This card can be programmed to generate an interrupt periodically at a software controlled rate. A rate of 100 ms was chosen for this design. The software to provide the time-out functionality consists of a

set of service calls that can be made by an executing process to request that a time-out call be added or removed from the time-out list, and an interrupt service routine to handle the interrupts from the real-time clock. Section 6.3.1 describes the timer module service calls, and Section 6.3.2 describes the timer interrupt service routine.

6.3.1 Timer module service requests

The timer module maintains a doubly linked list of nodes used to hold the current list of requests for time-out calls. Each node on the list contains the address of the function to be called, a parameter for the function, a number indicating the time in clock ticks until the function should be called, and the forward and backward pointers to the neighbor nodes on the list. The entries on the list are sorted by their "time to fire". Because of this timer ordering, the time field for each node is stored as the time to fire after the previous node fires. Thus, the total time until an entry is called is the sum of the time field for all entries up to and including that entry.

Two service requests are provided by the timer module: one to add a request to the timer list and one to delete an entry from the timer list. The delete service request allows a process to cancel a previous request for a delayed call. This is desirable if a process wishes to wait for an event for a maximum specified period of time. In this case a time-out request can be made to call a routine which will activate the process after the desired waiting period. If the event occurs before the end of the waiting period, the event handler can activate the process and the process can then cancel the time-out request.

The routine to add a request to the timer list accepts three parameters: the address of the function to be called, an integer parameter to be passed to the called function, and the desired timer delay. The routine walks the linked list to find the correct insertion point, adds a node for the new request to the linked list, and adjusts the time field of the entry immediately following

this new entry so that its value reflects the time to fire after the new entry rather than after the entry in front of the new entry. The add routine returns a pointer to the new entry. The calling process can use this pointer as a parameter for a subsequent call to cancel the delayed call request.

The routine to cancel an entry accepts only one parameter, the pointer that was returned from a previous call to add the entry. The cancel routine locates the desired entry in the linked list, removes it, and updates the time field of the following entry so that its value represents the time delay after the entry in front of the deleted entry.

6.3.2 Timer module interrupt handler

The timer interrupt handler is called after every clock tick. This routine checks to see if there are any entries on the timer list, and if so it decrements the time field of the first entry. Since the time field of each entry contains the time delay following the previous entry, this effectively decrements the time field for all entries. If the time field of the first entry has reached zero, the entry is removed from the linked list and the function whose address is stored in the entry is called with the specified parameter. If there are one or more entries with a time field of zero following this first entry, they too are removed and called in a similar manner.

6.4 LAN Adapter Interrupt Handlers

Each LAN port controlled by the bridge connects to a token ring network through an 802.5 LAN adapter interface card. Each LAN adapter card contains an interrupt signal which is activated by the card to signal an interrupt to the microprocessor on the IBM-PC. This bridge design contains two LAN ports. Therefore, in addition to the timer interrupt handler, two LAN adapter interrupt handlers were written, one for each port. Since the operation of the

two ports is symmetric, the interrupt handler code is virtually the same for both ports. This section describes these interrupt handlers.

An interrupt can be generated by a LAN adapter card for two separate reasons. As explained in Section 5.2, a DMA request from the LAN adapter will generate an interrupt request to the PC. The TMS 38030 System Interface Chip can also generate an interrupt request to the PC to indicate the occurrence of one of several events of interest. Both types of interrupts are multiplexed through the same interrupt request signal to the PC.

The first operation of the interrupt handler is to determine which of the two reasons described above caused the interrupt. This determination is made by reading the status register on the LAN adapter card. If the interrupt is a DMA request from the 38030, the interrupt service routine determines the direction of data transfer, the starting address, and the DMA length by reading the appropriate registers on the LAN adapter and then performs a string move instruction to move the data. If the interrupt is an event notification from the 38030, the event type is determined by reading the interrupt register in the 38030, and the appropriate process is activated.

6.5 Bridge Processes

Five processes are used to control the bridge. A receive process exists for each bridge adapter. These processes are activated upon receipt of a frame from the appropriate network interface. The receive process checks the frame to determine if it should be forwarded. If it is a frame with a routing information field indicating a broadcast frame the receive process checks the route designator list to determine if the frame has already passed the bridge. If not the process adds a route designator to the routing information field and passes the frame to the transmit process of the other LAN adapter board. If the frame is not a broadcast frame, the

receive process checks the routing information field for the presence of its bridge number and the two LAN segments to which it is connected. If these identifiers are in the proper sequence in the routing information field, the frame is also passed to the transmit process of the other LAN adapter. The transmit process for each board accepts frames and programs its LAN adapter to transmit these frames. Finally, the management process periodically activates itself using the delayed time-out mechanism to update the screen with the current bridge statistics.

7. ACKNOWLEDGEMENTS

I would like to express my thanks to my Major Professor, Dr. Doug Jacobson, for his patience and understanding - especially during the times when my delay was high and my throughput was low. I would also like to thank Dr. Terry Smay and Dr. Johnny Wong for serving on my graduate committee, Gary Bridges for tracking down the parts I needed, and Pam Meyers for constantly making sure my paperwork was correctly filed.

8. BIBLIOGRAPHY

Backes, Floyd. 1988. Transparent Bridges for Interconnection of IEEE 802 LANs. IEEE Network 1 (January) : 5-9.

Currie, W. Scott. LANs Explained. New York: Ellis Horwood Limited, 1988.

Dixon, Roy C., and Daniel A. Pitt. 1988. Addressing, Bridging, and Source Routing. IEEE Network 1 (January) : 25-32.

Dlugosz, John M. 1988. A Multitasking Kernel for C Programmers. Computer Languages 10 (October): 49-61.

Hamner, M. Claire, and Gerald R. Samsen. 1988. Source Routing Bridge Implementation. IEEE Network 1 (January) : 33-36.

Institute of Electrical and Electronic Engineers. 1985. ANSI/IEEE Standard ISO Draft Proposal Token Ring Access Method IEEE, New York.

Perlman Radia. 1985. "An Algorithm for Distributed Computation of a Spanning Tree in an Extended LAN." In Proceedings of the 9th Data Communications Symposium Held in Montreal, 1984.

Soha, Michael, and Radia Perlman. 1988. Comparison of Two LAN Bridge Approaches. IEEE Network 1 (January) : 37-43.

Texas Instruments. 1986.TMS380 Adapter Chipset User's Guide Texas Instruments, Dallas, Texas.

Zhang, Lixia. 1988. Comparison of Two Bridge Routing Approaches. IEEE Network. 1. (January) : 44-48.