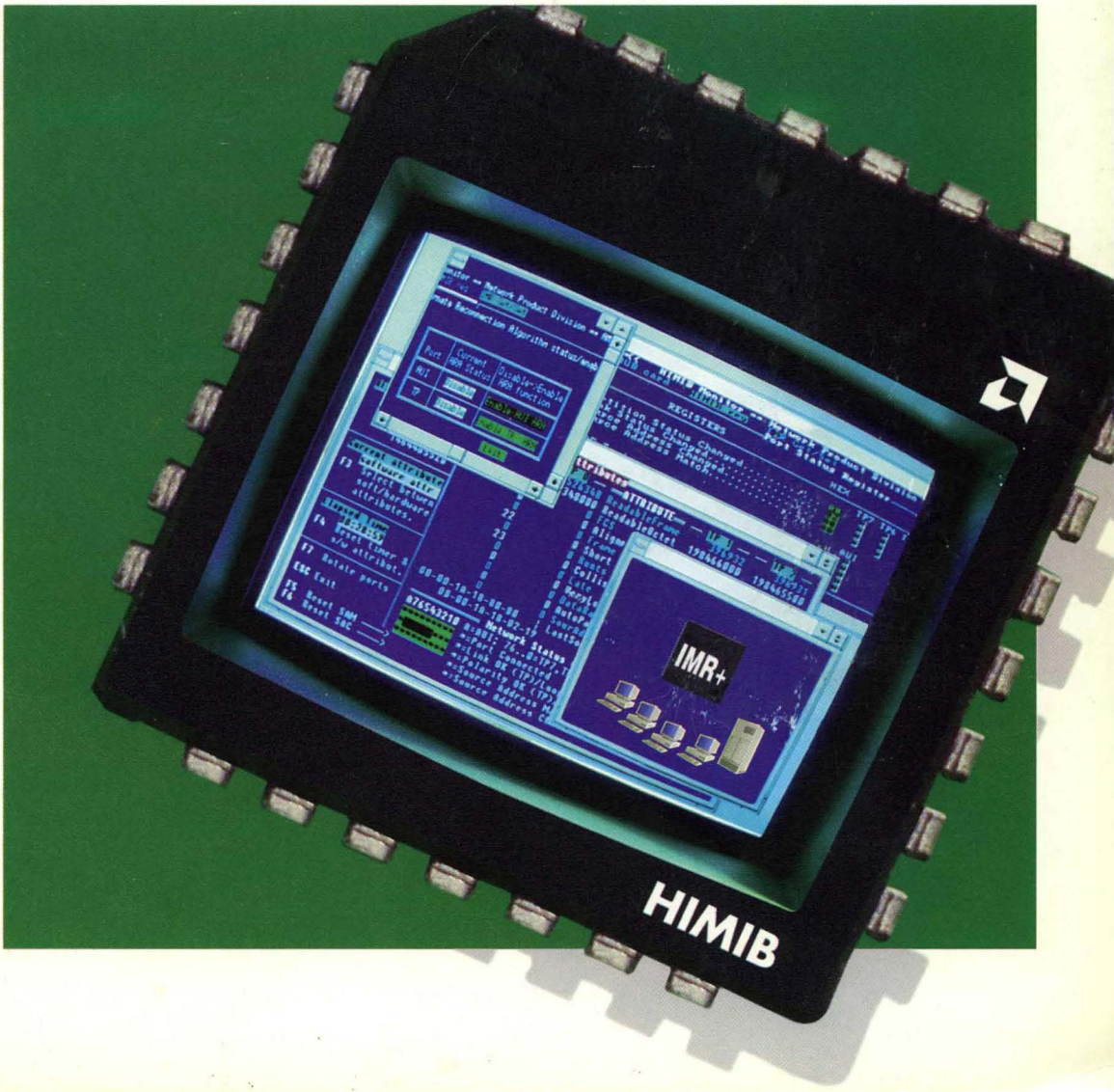
IEEE 802.3 Repeater

Technical Manual

Advanced
Micro
Devices



IEEE 802.3 Repeater

Technical Manual

ADVANCED MICRO DEVICES

TABLE OF CONTENTS

SECTION

1 TECHNOLOGY OVERVIEW

1.1 TERMINOLOGY/DEFINITION

Computer networks allow computer systems or stations to share information. The network consists of the communications medium (typically a cable plant), and network transmission devices to assist in the communication.

The physical cable plant is generally configured in either a star or a bus topology. In a bus topology all stations tap into a single cable that runs from one station to the next. In a star topology the stations are all connected to a central point, where a network hub or repeater connects the cable segments together.

The star topology offers two major advantages. The first advantage is that the computer network can now be merged with the existing building telephone network. The telephone is traditionally configured in a star topology, and a single integrated cable plant results in lower costs and easier cable management. The second advantage of star networks is that the repeater is a convenient point for monitoring and managing the network. Some station failures result in the stations transmitting endlessly on the network. In a bus topology this would disrupt the entire network. In a star topology, the hub or repeater unit can detect the fault and isolate that station from the network. The network remains operational and available for use by the other stations.

The access method is the set of rules that stations use to control when they can access the network. In Ethernet the access method is Carrier Sense, Multiple Access with Collision Detect, or CSMA/CD. This means that before a station transmits, it monitors the network for activity. It waits if it detects that another station is already transmitting. Once the network is available, the station is free to transmit. While transmitting, stations monitor the network for collisions with other stations, that is more than one station transmitting at the same time. When a station detects a collision it ceases to transmit, then after a variable amount of time it attempts to transmit once again.

To a large degree the access method determines the kind of hub needed in a star topology cableplant. CSMA/CD is based on the concept that station transmissions are broadcast on the complete network, and use that fact to determine when there is a collision. The simplest hub for Ethernet networks could have been a passive star, where each segment is passively coupled to the rest of the segments. However this would result in significant loss of signal strength, and high network error rates. For this reason 10BASE-T network hubs are active repeaters. These repeaters regenerate the received signal and broadcast it on the rest of the network. Additionally, an Ethernet repeater must perform collision detection for each of its attached links.

Repeaters perform these essential tasks:

- Repeaters regenerate the received signal to comply with IEEE 802.3 specifications prior to retransmitting it on all links.
- Repeaters detect and propagate collisions to all stations.
- And finally repeaters exclude (partition) stations from the network under certain fault conditions.

The repeater market can be broken down into four product categories. The following sections describe the different categories, along with their main functional characteristics.

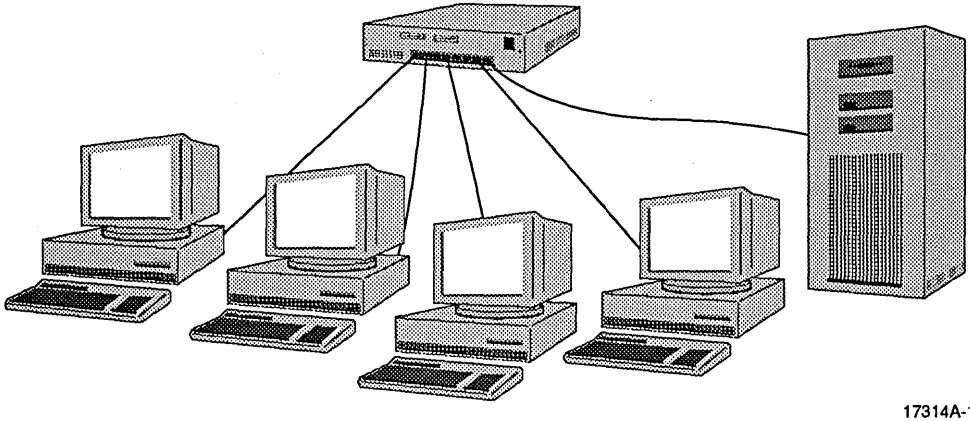
1.2 OVERVIEW OF APPLICATIONS

In the few years since the development of the IEEE 802.3 10BASE-T standard, 10BASE-T repeaters have evolved into diverse classes of products. These products are characterized by different numbers of ports, modularity, reliability, manageability, and different costs. This section describes four categories of IEEE 802.3 Repeater products, and their essential characteristics.

1.2.1 Fixed Configuration Unmanaged Repeaters

Unmanaged repeaters perform only the basic IEEE 802.3 functions. These devices vary in configuration from 8 to 32 ports. They adhere to the strategy of monitor by LED, manage by disconnect, in that they provide port status and error information through visual indicators only. They provide no facility for remote monitoring or control, although they may provide a simple local connection (RS232) to connect a computer terminal for basic maintenance.

Figure 1-1 Fixed Configuration Unmanaged Repeater



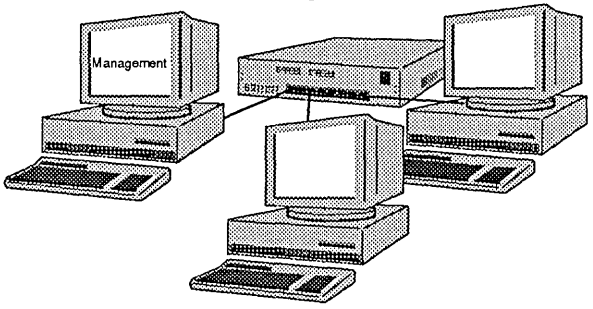
17314A-1

The advantage of these devices has been their low cost. They achieved this low cost by eliminating two essential capabilities, manageability and modularity. Manageability has historically been costly to implement, resulting in significantly higher product cost. Modularity allows repeaters to support variable port configurations. However modular repeaters require a higher initial cost for the system chassis, which is then amortized across many more ports.

1.2.2 Fixed Configuration Managed Repeaters

Figure 1-2 shows a network using a managed repeater. This repeater allows a remote management station to monitor and control its operation.

Figure 1-2 Fixed Configuration Managed Repeater



17314A-2

The term "management" means that the repeater supports access to information regarding its operation. Managed 10BASE-T repeaters differ in exactly what information is available, and how that information can be accessed. The recently completed IEEE 802.3 Repeater Management standard defines common sets of information required in managed IEEE 802.3 repeaters.

Most managed 10BASE-T repeaters support two modes of remote access. In-band access allows remote management stations to manage the repeater through the repeater network. Out-of-band access allows remote management through a separate interface, such as a serial port. Local management capabilities allow access to management information through a local interface, such as front panel switches, LED's, or an attached terminal.

Remote management uses a management protocol to transmit management information between the repeater and a remote management station. This can be as simple as a serial terminal interface for modem access, to a network management protocol such as the Simple Network Management Protocol (SNMP).

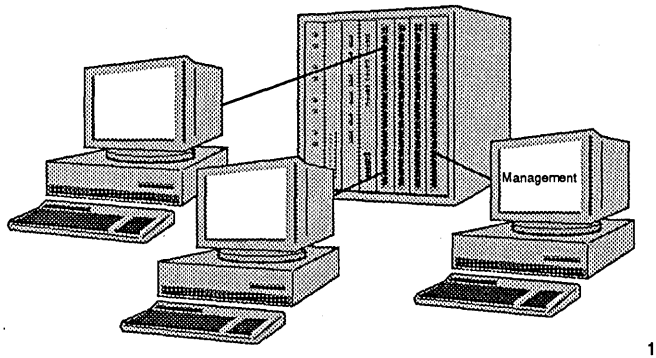
Fixed configuration managed repeaters have recently been extended to support system level modularity. Known as "rack and stack" repeaters, these devices combine the flexibility of the modular repeater with the low cost of a fixed configuration repeater. This product is essentially a fixed configuration repeater, with an additional external interface to connect it to other repeaters.

The external interface is used to allow these devices to act in concert as a single repeater. Some of these products allow a single "master" unit to manage other "slave" units through a management interface, resulting in additional cost savings.

1.2.3 Modular (Enterprise) Repeaters

Modular repeaters are used in environments where the potential need exists for a large number of ports. By combining the repeaters into a single system chassis, modular repeaters take advantage of shared costs such as the power supply and network management module. These systems have a high initial cost due to the increased size and power requirements, however they offer easy expansion by installing additional port modules.

Figure 1-3 Modular Repeaters



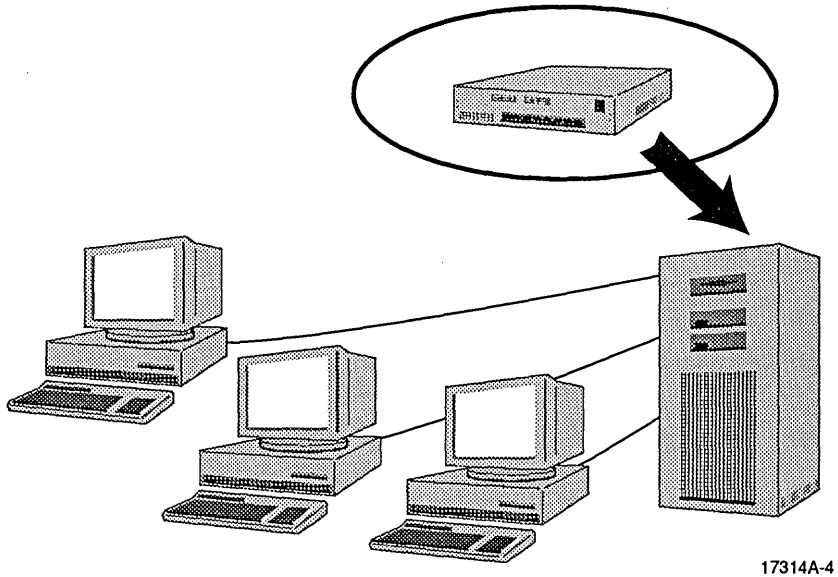
These systems also support enhanced capabilities, such as redundant power supplies and support for multiple LAN segments. Multiple LAN segment support allows card by card segmentation of the network in busy networks. These modular hubs often provide bridging and routing between the segments, as well as support for different media access protocols (Token Ring, FDDI etc.), and different media types (STP, UTP, Coax, Fiber Optics).

This class of system is generally deployed in large building and campus networks. In these environments network reliability is a major concern. For that reason this class of repeater almost always includes network management capabilities. That capability is reflected in the higher initial cost for these systems.

1.2.4 Server Based Repeaters

Server based repeaters are a new class of 10BASE-T product. These repeaters consist of one or more hub cards designed to be installed in the file server itself. In many cases, file servers are located near 10BASE-T repeaters. File servers already have a chassis, power supply, and processor capable of managing the repeater. In these environments, where the file server has slots available for use by one or more repeater cards, this provides a low cost option.

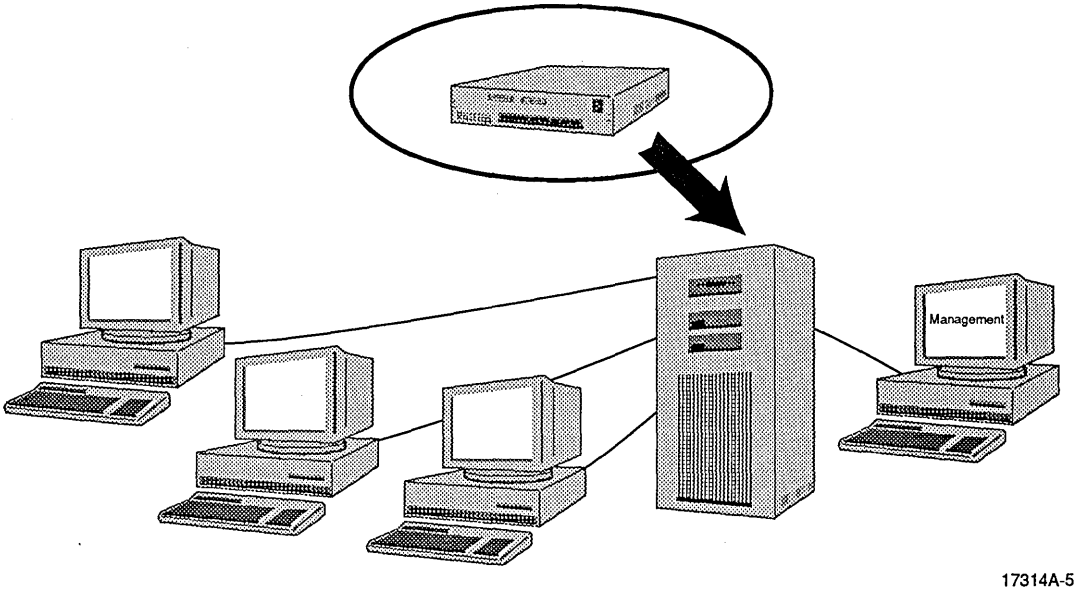
Figure 1-4 Unmanaged Server-Based Repeaters



17314A-4

Initially server based repeaters provided only unmanaged capabilities. These products are the equivalent of a fixed configuration dumb repeater. Newer server based repeaters use the capabilities of the server processor to add management support for these products. Novell, Inc. recently introduced the Hub Management Interface (HMI) specification. This specification provides a hardware independent interface between the Novell Netware server management software and the repeater (hub) card.

Figure 1-5 Managed Server-Based Repeaters



17314A-5

The AMD HIMIB and IMR+ repeater chipset directly supports the additional management requirements for server based repeaters. AMD offers a design kit called the ISA-HUB™-KT for managed server applications. This design kit provides a complete hardware and software solution for server based managed repeater applications. A complete user manual for the ISA-HUB is available from AMD (PID #17642A).

1.3 OVERVIEW OF STANDARDS

Computer networking standards are developed by several different standards bodies with overlapping jurisdiction. Ethernet repeaters are defined in the standards of three different organizations. The three organizations are the Institute of Electrical and Electronic Engineers (IEEE), the International Standards Organization (ISO), and the Internet Engineering Task Force (IETF).

The IEEE 802.3 committee defined the core Ethernet standard. This work involved the original coax and repeater specifications, the newer 10BASE-T specification, and various IEEE 802.3 management specifications. The IEEE standards are ultimately submitted to the International Standards Organization (ISO) for final approval as an international standard.

This was a straight forward process for the basic physical layer and media access specifications. However the network management specification is split between ISO and IEEE. ISO has developed standards for the specification of management information. The IEEE 802.3 committee used this as the basis for their work in the specification of repeater management. Additionally the IEEE 802.1 committee specified management information required for all repeaters in the IEEE 802 protocol suite.

These standards work together to specify the requirements for IEEE 802.3 Repeater Management. What is not specified in these standards is the protocol to transport network management information over a network. The most widely supported network management protocol is the Simple Network Management Protocol (SNMP). The Internet Engineering Task Force (IETF), an organization responsible for the development of guidelines for the administration of the U.S. based Internet, specified the SNMP.

The IETF as a organization ties into the world standards organization through the US Department of Defense, as an entity separate from the IEEE or ANSI. What is important is that the IETF specifies how management information will be represented when transmitted over the network. Specifically the IETF has defined an IEEE 802.3 Repeater Management Information Base (MIB) that is used to transmit repeater information through the network. This MIB was based on the work of the IEEE 802.3 Repeater Management task group.

A final specification of interest is the Hub Management Interface specification developed by Novell, Inc. This is an interface specification that describes what management information must be supported by IEEE 802.3 hub cards in Novell file servers, and how that information is passed to the Novell management protocol stack. This specification is based on an early draft version of the IEEE 802.3 Repeater Management standard, and has additional requirements beyond the published IEEE 802.3 specification.

1.3.1 IEEE 802.3 Repeater Standard

The original IEEE 802.3 standards defined the use of coax cable for the transmission of network information between many stations on a single network segment. The distance of the network and the maximum number of stations attached to the network were limited by signal loss on the coax cable.

IEEE 802.3 then developed Section 9 of the standard, detailing the use of a repeater to extend the distance and number of stations that could be attached to a single IEEE 802.3 network. The repeater regenerated the electrical signal and retimed the transmitted bits, connecting one coax segment to another. The standard supports an extended network topology where stations can be separated by as many as 4 repeaters.

All of the segments attached by repeaters are part of the same collision domain, meaning that the CSMA/CD access method extends through the repeaters over the entire network. The round trip delay time must be limited to the maximum collision window defined by the core IEEE 802.3 standard. The total size of this network is limited by the need to sense collisions between stations on opposite ends of the network.

Where the IEEE 802.3 coax repeater standard allows connection of multiple stations through a coax segment, the IEEE 802.3 10BASE-T standard for transmission of Ethernet over Unshielded Twisted Pair (UTP) cable requires that a repeater port be attached to each individual 10BASE-T station. An effective way of implementing this is with a multiport 10BASE-T repeater, which provides multiple 10BASE-T station ports. Each 10BASE-T link can be up to 100 meters long, and consists of two twisted pairs (4 conductors total) of UTP cable. One pair is used for transmit, and one for receive. The use of 10BASE-T repeaters in an 802.3 network is still constrained by the IEEE 802.3 limitation of at most 4 repeaters between any two end stations.

Where coax based IEEE 802.3 stations are physically attached to the same cable, 10BASE-T stations are physically attached to individual cables called "link segments". However the 10BASE-T repeater makes them appear to be connected to a single cable. It does this by propagating the incoming frames from any station to all other stations in the network, as well as detecting and signaling when a collision condition is detected.

The repeater receives frames from any attached station, and propagates it to the rest of the attached stations. During the propagation of the frame, the repeater regenerates the signal, and retimes the bits. Each repeater has a clock recovery circuit that enables it to receive and decode (separate clock from data) the incoming bit stream. When the repeater detects an incoming frame on any port, it connects that port to the clock recovery circuit where the frame is then decoded. The decoded bits are passed into a small FIFO, from which they are retransmitted on all ports using the repeater's locally generated clock.

This process removes the clock jitter and restores the signal level of the incoming frame. A few of the initial bits in the frame can be lost in this process because the clock recovery circuit requires a few bit times to synchronize with the incoming data. Subsequent frames can come from different stations, each with a different clock source. This means that the repeater clock recovery circuit must be able to synchronize to a different clock source on each frame.

The repeater also plays a central role in collision detection and propagation. Collision is detected by monitoring activity on all ports. When two or more incoming ports go active simultaneously, or when the repeater detects a receive mode collision on any port, it signals a collision condition by transmitting a Jam signal to all ports. This is logically identical to the collision behavior of a coax based IEEE 802.3 network, with the exception that collision detection by the repeater is quicker and easier than the DC collision detection method used by stations on coax IEEE 802.3 networks.

1.3.2 IEEE 802.3 Repeater Management

The IEEE 802.3 committee is working to define the management requirements for IEEE 802.3 networks. This work initially resulted in the Section 5 "Layer Management" standard, and more recently the Section 19 "Layer Management for 10Mb/s Baseband Repeaters" standard. The IEEE 802.3 Repeater Management specification defines what is managed inside a repeater, not how a repeater is managed by a network manager. The individual entities inside a repeater that can be managed are referred to as the management objects. These objects fall into three basic classes.

Attributes

Attributes are the accessible pieces of information in the repeater. They provide status information about the operational state of the repeater and the network, such as the

health of the repeater, the number of collisions, the total number of received bytes, etc. Attributes can be readable, writable, or both. In the case of the IEEE 802.3 Repeater Management standard, all attributes are read only.

Actions

Actions are commands that the network manager can direct the repeater to execute. They allow the manager to alter the state or value of a management object, such as resetting the repeater, enabling or disabling ports, etc.

Notifications

Notifications are the messages the repeater is required to send to a network manager when a significant event occurs. Unlike the attributes and actions that originate at the network manager, notifications originate at the repeater itself. Because of their potential impact on the available network bandwidth, these messages are only used to report significant events.

IEEE 802.3 management objects relate to the repeater in a hierarchical manner. Some of the objects support management of the overall repeater. Others are used to manage the individual ports in the repeater. An 8 port repeater would then require one of these objects for each port. Individual groups of ports are also assigned several objects to support management of a cluster of ports as a single entity. A single repeater could then have 4 groups, each consisting of 24 ports for a total of 96 ports.

The IEEE 802.3 management objects are grouped into three sets, or packages. Packages can each consist of any number of attributes, actions and notifications. The three packages defined for IEEE 802.3 Repeater Management are Basic Control, Performance Monitoring, and Address Tracking.

Basic Control is the required minimum set of objects that must be supported by managed IEEE repeaters. Performance Monitoring is the optional set of management objects necessary to monitor the performance of IEEE 802.3 repeaters. These objects consist of many real-time port and repeater statistics. Support for these objects is provided by the AMD HIMIB chip. Address tracking provides port level detection of attached station addresses. Like the Performance Monitoring package, the Address Tracking package requires statistics supported by the HIMIB device.

1.3.3 IEEE 802.3 MAU Management

The most recent element in the IEEE 802.3 architecture to undergo management standardization is the Media Attachment Unit (MAU). The IEEE 802.3 MAU is the actual physical line interface. Using MAU management, the network manager can determine status information and initiate control actions on the MAU.

Each port in a managed repeater is subject to support of the IEEE 802.3 MAU Management requirements. Like the IEEE 802.3 Repeater Management suite, the IEEE 802.3 MAU Management standard is organized in several packages. Basic Control provides the minimum mandantory requirements for managed MAUs. MAU Control enables the network manager to control as well as monitor the MAU operation. Media Loss is a conditional attribute that provides optional support for AUI attachments. Broadband DTE MAU is the final conditional package, providing optional support for broadband network management.

1.3.4 Simple Network Management Protocol

The IEEE 802.3 suite of management standards specifies the management objects required to support standard interoperable management of IEEE 802.3 systems. It does not address the issue of how this information is communicated to a remote manager. The Simple Network Management Protocol (SNMP) is one of many network manage-

ment protocols. Because of its origins in the TCP/IP networking community the SNMP is widely used in complex heterogeneous corporate networks.

The basic function of the SNMP is to provide a protocol to "get" and "set" management information base (MIB) attributes in remote stations, and for the remote station to signal an "alarm" condition to the network manager. Using this capability network managers can monitor (get) and control (set) remote station operation. The IEEE 802.3 notification is supported by the SNMP alarm message.

In an IEEE 802.3 Repeater, SNMP allows network managers to enable and disable ports, as well as to acquire frame error and performance statistics. The SNMP protocol is specified in Request for Comments (RFC) 1157.

1.3.5 MIB-I, MIB-II, Repeater MIB and RMON MIB

Along with a network management protocol (SNMP), network management requires a standard way of addressing specific MIB attributes. This structure of network objects is referred to as the containment tree, and consists of a hierarchy of management object classes. The IEEE 802.3 repeater ports can be addressed individually or as part of a group. The groups are combined into a repeater, which in turn is part of a station. The station could in turn contain additional repeaters and other manageable objects.

The IETF defined MIB-I (RFC 1156) as the original specification of management objects for stations in an SNMP network. These objects dealt primarily with network interfaces and elements of the protocol stack. MIB-II (RFC 1213) modified these object definitions, extending them to cover other related objects. Neither MIB-I nor MIB-II defined support for repeater management objects.

The IETF Repeater MIB (RFC 1368) was developed based on the IEEE 802.3 Repeater Management standard. It describes the SNMP based management attributes required in repeater management objects.

These MIBs are generally oriented toward management by network management stations within the local campus network. The amount of traffic to actively monitor station operation is not excessive by LAN standards, however because of bandwidth limitations remote monitoring is more difficult. Because of this a specification for remote management over a wide area network was defined. This allows a remote network monitoring station to report general network information back to a central management station. Only when requested by the central management station would detailed traffic information be transmitted between the remote monitor and the central manager.

The specification for a remote network monitoring management information base is referred to as the RMON MIB. This MIB specifies a broad selection of optional management and monitoring capabilities, allowing vendors to craft RMON products which span a broad range of applications and cost.

While not directly related to IEEE 802.3 Repeater Management, many of the RMON MIB (RFC 1271) attributes mimic those defined in the IEEE 802.3 management suite. 10BASE-T repeaters are often appropriate devices to host the remote management engine, and could have RMON MIB capabilities installed as an option.

SECTION

2 REPEATER MANAGEMENT STANDARDS

2.1 IEEE 802.3 REPEATER MANAGEMENT

This section describes the IEEE 802.3 Repeater Management Information Base (MIB). The IEEE 802.3 repeater MIB consists of several different object types. From an implementation point of view the important ones are the attributes, actions, and notifications. Attributes are used to monitor and control device operation. Actions force the repeater to execute a management command, while notifications are used by the repeater to signal the network manager that a significant event has occurred. The following sections list the three management packages, and their management objects.

The IEEE 802.3 MIB objects are grouped into three packages, known as the Basic Control package, the Performance Monitoring package, and the Address Tracking package. Repeater management objects fall into one of three main classes. The "repeater" object class contains those objects necessary for overall repeater management. The "group" object class contains objects necessary to manage collections of ports. Ports can be clustered into groups for easier manageability, and the "group" MIB attributes provide observability into those groups. A group is frequently used to denote the characteristics of a physical implementation, such as a 12 or 16 port card which fits into a modular chassis. The "port" object class contains objects that are used to manage the operation of individual ports in the repeater group.

2.1.1 Basic Control Package (Mandatory)

Table 2-1 lists the attribute, action, and notification management objects in the IEEE 802.3 Repeater Management Basic Control package. These management objects are primarily concerned with repeater configuration information and repeater status (health). Following the table is a brief description of the management object. Because the Basic Control package objects do not require high speed event tracking, they are typically implemented in software. For the exact specification of these and other objects refer to the actual standard documents.

Table 2-1 IEEE 802.3 Repeater Basic Capabilities Package

Repeater Class	Object Name	Object Type	Ref. Para
Repeater	repeaterID	Attribute Get	19.2.3.2
Repeater	repeaterGroupCapacity	Attribute Get	19.2.3.2
Repeater	groupMap	Attribute Get	19.2.3.2
Repeater	repeaterHealthState	Attribute Get	19.2.3.2
Repeater	repeaterHealthText	Attribute Get	19.2.3.2
Repeater	repeaterHealthData	Attribute Get	19.2.3.2
Repeater	resetRepeater	Action	19.2.3.3
Repeater	executeNonDisruptiveSelfTest	Action	19.2.3.3
Repeater	repeaterHealth	Notification	19.2.3.4
Repeater	repeaterReset	Notification	19.2.3.4
Repeater	groupMapChange	Notification	19.2.3.4
ResourceTypeID	resourceTypeIDName	Attribute Get	N/A
ResourceTypeID	resourceInfo	Attribute Get	N/A
Group	groupID	Attribute Get	19.2.5.2
Group	groupPortCapacity	Attribute Get	19.2.5.2
Group	portMap	Attribute Get	19.2.5.2
Group	portMapChange	Notification	19.2.5.2
Port	portID	Attribute Get	19.2.6.2
Port	portAdminState	Attribute Get	19.2.6.2
Port	autoPartitionState	Attribute Get	19.2.6.2
Port	portAdminControl	Action	19.2.6.3

repeaterID
"repeaterID" is a read-only attribute used to identify the specific instance of the repeater in the system. It consists of an integer value between 1 and 1024.

repeaterGroupCapacity
"repeaterGroupCapacity" is a read-only attribute used to identify the maximum number of groups supported by this repeater. It consists of an integer value between 1 and 1024. "repeaterGroupCapacity" can be larger than the actual number of installed groups.

groupMap
"groupMap" is a read-only attribute used to identify the actual installed groups in this repeater. It consists of a-bit string "repeaterGroupCapacity" bits in length. Each-bit signals the absence ("0") or presence ("1") of the group, numbered from lowest to highest.

repeaterHealthState
"repeaterHealthState" is a read-only attribute used to indicate the general health of the repeater. It consists of a numeric value. The possible values are:

1	other	undefined or unknown
2	OK	no known failures
3	repeaterFailure	known to have a repeater-related failure
4	groupFailure	known to have a group-related failure
5	portFailure	known to have a port-related failure
6	generalFailure	has an unspecified failure type

In the event that multiple failures are present, the highest priority failure (lowest numbered failure) should be reported.

repeaterHealthText
"repeaterHealthText" is a read-only attribute, consisting of a text string that provides relevant information about the operational state of the repeater to a network manager. The contents of this attribute are vendor specific, and can be used to provide detailed failure information or problem resolution instructions. The string must be printable text no longer than 255 characters in length.

repeaterHealthData
"repeaterHealthData" is a read-only attribute consisting of a block of information related to the operational state of the repeater. The encoding of the information is vendor specific, and must not exceed 255 bytes in length.

resetRepeater
"resetRepeater" is an action. When received, the repeater state is reset to "Start" state as defined in the IEEE 802.3 Section 9 standard. This forces the repeater to execute a disruptive self-test. The exact requirements of the self test are unspecified. The repeater maintains management information throughout the execution of this action, and transmits a "repeaterReset" notification in response to this action request. During this action the repeater must not inject any packets onto any segment. Received packets may or may not be transferred at the implementors discretion.

executeNonDisruptiveSelfTest
"executeNonDisruptiveSelfTest" is an action. When received, the repeater executes a vendor specific test. During the test the state of the repeater is unchanged, and the management information remains intact. This test must not inject any packets onto any segment attached to the repeater. This test does not interfere with the transfer of any packets through the repeater. Completion of this test results in a "repeaterHealth" notification.

repeaterHealth
"repeaterHealth" is a notification. It is sent when the health state of a repeater changes, not including initial powering up of the repeater. At a minimum the repeaterHealth notification consists of the "repeaterHealthState" attribute. It optionally includes the "repeaterHealthText" and "repeaterHealthData" attributes.

repeaterReset
"repeaterReset" is a notification. It is sent when the repeater is reset at power-up, or upon completion of the resetRepeater action. repeaterReset notification contains repeaterHealthState, and optionally repeaterHealthText and repeaterHealthData attributes.

groupMapChange
"groupMapChange" is a notification that a group has been logically inserted or removed from the repeater. This notification is not sent during repeater power up. This notification consists of the new groupMap attribute.

resourceTypeIDName
"resourceTypeIDName" is a read-only attribute. It is used to contain the name of the resourceTypeID managed object, a fixed value of "RTID". See 802.1F Common Definitions and Procedures for IEEE 802 Management Information (Draft) for additional details.

resourceInfo
"resourceInfo" is a read-only attribute provided by repeater manufacturer. It is used to describe the resource. The attribute contains the ManufacturerOUI (Organizational Unit Identifier), ManufacturerName, ManufacturerProductName, and ManufacturerProduct-Version. See 802.1F Common Definitions and Procedures for IEEE 802 Management Information (Draft) for additional details.

groupID
"groupID" is a read-only attribute used to identify a specific instance of a group class in the repeater. It is an integer value in the range of 1 to 1024. This value cannot exceed "repeaterGroupCapacity".

groupPortCapacity
"groupPortCapacity" is a read-only attribute used to identify the number of potential ports in this instance of a group. It is an integer value in the range of 1 to 1024. The number of actual ports present may be less than or equal to the groupPortCapacity.

portMap
"portMap" is a read-only attribute used to identify the actual installed ports in this group. It consists of a-bit string "groupPortCapacity" bits in length. Each-bit signals the absence ("0") or presence ("1") of the port, numbered from lowest to highest.

portMapChange
"portMapChange" is a notification that a port has been logically inserted or removed from the group. This notification is not sent during repeater power up. This notification consists of the new portMap attribute.

portID
"portID" is a read-only attribute used to identify the specific instance of the port in the group. It consists of an integer value between 1 and 1024.

portAdminState
"portAdminState" is a read-only attribute used to indicate whether the port is currently enabled or disabled. A disabled port does not transmit or receive. This attribute is preserved across repeater reset and loss of power. This attribute is not writable, however it is controlled through the portAdminControl action. This attribute takes precedence over the auto-partition mechanism. A value of '1' means the port is disabled, a value of '2' means it is enabled.

autoPartitionState
"autoPartitionState" is a read-only attribute indicating the state of the port autoPartition state machine. A value of a 1 means that the port is currently auto partitioned, a value of 2 means that the port is currently not auto partitioned.

portAdminControl
"portAdminControl" is an action. When received it modifies the "portAdminState" attribute. A '1' is used to enable the port, while a '0' is used to disable the port. A disabled port does not transmit or receive any information. A transition from disabled to enabled in the "portAdminState" causes the auto partition state machine to return to the 'BEGIN' state, as defined in Section 9 of the IEEE 802.3 standard.

2.1.2 Performance Monitoring Package (Optional)

The performance monitoring package provides additional statistics beyond those provided by the basic control capabilities package. These statistics consist of multiple 32-bit counters on each port, all of which are fully supported by the AMD HIMIB. Prior to the introduction of the AMD HIMIB these management objects required extensive hardware and software support.

Table 2-2 IEEE 802.3 Performance Monitoring Package

Repeater Class	Object Name	Object Type	Ref. Para.
Repeater	transmitCollisions	Attribute Get	19.2.3.2
Port	readableFrames	Attribute Get	19.2.6.2
Port	readableOctets	Attribute Get	19.2.6.2
Port	frameCheckSequenceErrors	Attribute Get	19.2.6.2
Port	alignmentErrors	Attribute Get	19.2.6.2
Port	framesTooLong	Attribute Get	19.2.6.2
Port	shortEvents	Attribute Get	19.2.6.2
Port	runts	Attribute Get	19.2.6.2
Port	collisions	Attribute Get	19.2.6.2
Port	lateEvents	Attribute Get	19.2.6.2
Port	veryLongEvents	Attribute Get	19.2.6.2
Port	dataRateMismatches	Attribute Get	19.2.6.2
Port	autoPartitions	Attribute Get	19.2.6.2

transmitCollisions
"transmitCollisions" is a read-only attribute that counts the number of transmit collisions this repeater has detected. The value of the transmitCollisions attribute is a 32-bit counter with a minimum rollover time of 16 hours.

readableFrames
"readableFrames" is a read-only attribute that counts the number of valid frames detected by the port. Valid frames are from 64 bytes to 1518 bytes in length, have a valid frame CRC and are received without a collision. This attribute is a 32-bit counter with a minimum rollover time of 80 hours.

readableOctets
"readableOctets" is a read-only attribute that counts the number of total octets received on each port. This number is determined by adding the frame length to this register at the completion of every valid frame. This attribute is a 32-bit counter with a minimum rollover time of 58 minutes.

frameCheckSequenceErrors
"frameCheckSequenceErrors" is a read-only attribute that counts the number of frames detected on each port with an invalid frame check sequence. This counter is incremented on each frame of valid length (64 bytes to 1518 bytes) that does not suffer a collision during the frame. This counter is incremented on each invalid frame, however it is not incremented for frames with both framing errors and frame check sequence errors. This attribute is a 32-bit counter with a minimum rollover time of 80 hours.

alignmentErrors
"alignmentErrors" is a read-only attribute that counts the number of frames detected on each port with an FCS error and a framing error. Frames that have both framing errors and FCS errors are counted by this attribute, but not by the frameCheckSequenceErrors attribute. This attribute is a 32-bit counter with a minimum rollover time of 80 hours.

framesTooLong
"framesTooLong" is a read-only attribute that counts the number of frames that exceed the 1518 byte maximum frame size. This attribute is a 32-bit counter with a minimum rollover time of 61 days.

shortEvents
"shortEvents" is a read-only attribute that counts the number of instances where activity is detected with a duration less than the "ShortEventMaxTime" (74–82-bit times). This attribute is a 32-bit counter with a minimum rollover time of 16 hours.

runts
"runts" is a read-only attribute that counts the number of instances where activity is detected with a duration greater than the "ShortEventMaxTime" (74–82-bit times), but less than the minimum valid frame time (512-bit times, 64 bytes). This attribute is a 32-bit counter with a minimum rollover time of 16 hours.

collisions
"collisions" is a read-only attribute that counts the number of instances where a carrier is detected on the port, and a collision is detected. This attribute is a 32-bit counter with a minimum rollover time of 16 hours.

lateEvents
"lateEvents" is a read-only attribute that counts the number of instances where a collision is detected after the LateEventThreshold (480–565-bit times) in the frame. This event will be counted both by the "lateEvents" attribute, as well as the "collisions" attribute. This attribute is a 32-bit counter with a minimum rollover time of 81 hours.

veryLongEvents
"veryLongEvents" is a read-only attribute that counts the number of times the transmitter is active for greater than the MAU Jabber Lockup Protection Timer allows (4 ms–7.5 ms). This attribute is a 32-bit counter with a minimum rollover time of 198 days.

dataRateMismatches
"dataRateMismatches" is a read-only attribute that counts the number of occurrences where the frequency, or data rate of the incoming signal is detectably different from the local transmit frequency. This attribute is a 32-bit counter that is incremented on each such event.

autoPartitions
"autoPartitions" is a read-only attribute that counts the number of instances where the repeater has partitioned this port from the network. This attribute is a 32-bit counter that is incremented on each such event.

2.1.3 Address Tracking Package (Optional)

The address tracking package provides optional information related to the MAC address of frames received on each port. This information can be used to monitor attached station addresses on a port by port basis. All objects in the Address Tracking package are directly supported by the AMD HIMIB.

Table 2-3 IEEE 802.3 Address Tracking Package

Repeater Class	Object Name	Object Type	Ref. Para.
Port	lastSourceAddress	Attribute Get	19.2.6.2
Port	sourceAddressChanges	Attribute Get	19.2.6.2

lastSourceAddress
"lastSourceAddress" is a read-only attribute that saves the value of the Source Address field in the last frame it received. This attribute is a 6-byte field.

sourceAddressChanges

"sourceAddressChanges" is a read-only attribute that counts the number of times the source address field of received frames changes. This attribute is a 32-bit counter with a minimum rollover of 81 hours.

2.2 IEEE 802.3 MAU MANAGEMENT

Section 20 of the IEEE 802.3 standard defines the management requirements of a Media Attachment Unit (MAU). Table 2-4 defines the management objects specified in IEEE 802.3 Section 20. This standard was in draft form at the time this document was written, so the objects may change. The objects had not yet been assigned numbers at the time this was written. Refer to the draft document for detailed descriptions of the MAU object definitions.

Like the IEEE 802.3 Repeater Management Standard, the IEEE 802.3 MAU Management Draft consists of several packages. The Basic package is the minimum subset required for a managed MAU. The MAU Control package is an optional package that provides management control of MAU operation. The Media Loss Tracking package (actually one attribute) is a mandatory package for AUI's that keeps statistics on the media availability. The Media Loss Tracking package is optional for other MAU's. Finally, the Broadband DTE MAU package is a required set of management objects in broadband MAU's.

Table 2-4 IEEE 802.3 MAU Management

Package	Object Name	Object Type	Reference
Basic	mauID	attribute	20.2.3.2
Basic	mauType	attribute	20.2.3.2
Basic	mauMediaAvailable	attribute	20.2.3.2
Basic	jabber	attribute	20.2.3.2
Basic	mauAdminState	attribute	20.2.3.2
Basic	jabber	notification	20.2.3.4
MAU Control	resetMAU	action	20.2.3.3
MAU Control	mauAdminCtrl	action	20.2.3.3
Media Loss Tracking	mauLoseMediaCounter	attribute	20.2.3.2
Broadband DTE MAU	bBandSplitType	attribute	20.2.3.2
Broadband DTE MAU	bBandFrequencies	attribute	20.2.3.2

mauID

"mauID" is a read-only attribute used to identify the specific instance of the MAU on the port. It consists of an integer value.

mauType

"mauType" is a read-only attribute used to identify the IEEE 802.3 MAU type. This consists of an integer value, that is the section number where the specific type of MAU is specified.

mauMediaAvailable

"mauMediaAvailable" is a read-only attribute used to signal the status of the media link. For link based MAU's such as 10BASE-T this is the link test fail state. For other MAU's such as the basic AUI interface this is the loopback detection status.

mauLoseMediaCounter

"mauLoseMediaCounter" is a read-only attribute used to count the number of times the mauMediaAvailable attribute has gone from'available' to any other state. This counter must not be allowed to increment at a rate greater than 10 times per second.

jabber
"jabber" is a two part attribute consisting of a jabberFlag, and a jabberCounter. Both parts of the attribute are read-only.

The jabber Counter attribute counts the number of times the jabberFlag enters the 'fault' state. This counter must not be allowed to increment at a rate greater than 40 times per second.

mauAdminState
"mauAdminState" is a read-only attribute used to indicate the current control status of the MAU, that is what state the manager previously commanded the MAU to assume. Refer to mauAdminCtrl for the state definitions.

bBandSplitType
"bBandSplitType" is a read-only attribute used to monitor the configuration of a broadband IEEE 802.3 MAU. This attribute indicates whether the broadband cabling system is a single cable system or a dual cable system.

bBandFrequencies
"bBandFrequencies" is a two part integer used for a broadband IEEE 802.3 MAU consisting of the Transmitter Carrier Frequency and the Translation Offset Frequency. The Transmitter Carrier Frequency part of the attribute is the actual transmitter carrier frequency divided by 250 KHz. Likewise the Translation Offset Frequency part of the attribute is the translation offset frequency divided by 250 KHz.

resetMAU
"resetMAU" is an action that results in the MAU performing a reset. This reset must be at least 1/2 second in length, during which the AUI DI, DO, and CI should be idle. This reset should operate the same as a power on reset.

mauAdminCtrl
"mauAdminCtrl" is an action that controls the operational state of the MAU.

jabber
"jabber" is a notification that the MAU has detected a jabber fault. This is the same condition that determines the state of the jabber attribute (jabberFlag).

2.3 NOVELL'S HUB MANAGEMENT INTERFACE (HMI)

Novell Inc. has defined the Hub Management Interface (HMI) specification. Supporting IEEE 802.3 or hub cards in Novell file servers, this specification defines the hardware independent driver requirements for managed third party hub card vendors. This document specifically focuses on support for 10BASE-T repeaters, rather than supporting IEEE 802.3 repeaters in general.

The interface consists of two components. The first is a definition of objects accessible through "get" and "set" commands from the hub management protocol layers. The second is a specification of the memory structure used to communicate between the hub driver and the hub protocol stack. Two different memory structures are defined, the Hub Interface Table (HIT), and the Group Interface Table (GIT).

The IEEE 802.3 managed objects are split into two groups. Part of the objects are managed using the Get and Set capabilities of HMI. Others are accessed through variables in the HIT and GIT. The HMI specification was developed during the draft stages of the IEEE 802.3 Repeater Management specification, and differs in the number of managed objects.

Note that the following IEEE 802.3 Repeater Management objects have no HMI equivalence: repeaterID, resourceInfo, groupID, portMap, portMapChange, and portID. Additionally HMI offers only very limited support for the newly developed MAU Management specification.

2.3.1 Novell HMI/IEEE 802.3 Repeater MIB Mapping

Table 2-5 provides a summary of the HMI management objects, and their IEEE 802.3 equivalents. The text following the table provides an overview of the objects not previously described in the 802.3 Repeater Management section. For more detailed information refer to the Novell HMI specification.

The Novell management attributes are divided into four classes. The first three mimic the IEEE 802.3 Repeater Management definitions, with Basic Control, Performance Monitoring, and Address Tracking packages. The fourth is called Novell Extensions, and includes additional management attributes.

Table 2-5 HMI to IEEE 802.3 Management Object Equivalencies

Object Package	HMI Object Name	HMI #	IEEE Object Name
Basic	InfoTablePointer	0	
Basic	ResetHubAction	1	resetRepeater
Basic	ExecutesSelfTest1Action	2	executeNonDisruptive SelfTestAction
Basic	ExecutesSelfTest2Action	3	
Basic	PortAdminState	4	portAdminState
Basic	AutoPartitionState	5	autoPartitionState
Performance	TransmitCollisions	6	transmitCollisions
Performance	RepeaterVeryLongEvents	7	
Performance	PortVeryLongEvents	8	veryLongEvents
Performance	ReadableFrames	9	readableFrames
Performance	ReadableOctets	10	readableOctets
Performance	FrameCheckSequenceErrors	11	frameCheckSequenceErrors
Performance	AlignmentErrors	12	alignmentErrors
Performance	FramesTooLong	13	framesTooLong
Performance	ShortEvents	14	shortEvents
Performance	Runts	15	runts
Performance	Collisions	16	collisions
Performance	LateEvents	17	lateEvents
Performance	DataRateMismatches	18	dataRateMismatches
Performance	AutoPartitions	19	autoPartitions
Address	LastSourceAddress	20	lastSourceAddress
Address	SourceAddressChanges	21	sourceAddressChanges
Novell	PortLinkState	22	aMediaAvailable (MAU Mgmt)
Novell	RepeaterTotalOctets	23	
Novell	PortType	24	

InfoTablePointer

"InfoTablePointer" is a pointer to the Hub Information Table (HIT). This pointer is used to access other Hub management information.

ExecuteSelfTest1Action

"ExecuteSelfTest1Action" is equivalent to the 802.3 Repeater Management "executeNonDisruptiveSelfTest" action.

ExecuteSelfTest2Action
"ExecuteSelfTest2Action" is an action that causes the repeater to perform a disruptive self test. This test may interfere with the transfer of packets through the repeater.

PortAdminState
This HMI management object consists of two IEEE 802.3 objects. Both Set and Get operations are available for this object. The Set operation corresponds to the IEEE 802.3 portAdminControl action, while the get operation corresponds to the IEEE 802.3 portAdminState management attribute.

RepeaterVeryLongEvents
This management attribute tracks the number of VeryLongEvents the repeater has seen on all ports. This attribute is the sum of the IEEE 802.3 veryLongEvents attribute across all ports in the repeater.

PortLinkState
This management attribute is a Novell vendor specific extension to the basic IEEE 802.3 management attributes. It's purpose is to track on a port by port basis whether the port is receiving link status pulses. This attribute consists of an integer value from 1 to 3, with the following meaning:

1	Link Up
2	Link Down
3	Not Applicable – The port is not a 10BASE-T port.

RepeaterTotalOctets
This management attribute is a Novell vendor specific extension to the basic IEEE management attributes. This attribute counts the total number of bytes repeated by the hub, whether the frame had a valid FCS or not. This counter adds 8 bytes per frame for the preamble.

PortType
This management attribute is a Novell vendor specific extension to the basic IEEE management attributes. This attribute is a number from 1 to 4, with the following meanings:

1	Other port type
2	Normal port type
3	Local host port
4	AUI port

2.3.2 Hub Information Table (HIT)

Table 2-6 describes the structure of the Hub Information Table (HIT). Several of the variables in the table represent IEEE 802.3 repeater management attributes, and are passed through the table instead of through managed object requests.

Table 2-6 HMI Hub Information Table

HMI Object Name	HIT Table Offset (hex bytes)	IEEE Object Name
Reserved	00	
HubType	10	
Reserved1	11	
MajorVersion	12	
MinorVersion	13	
ManufacturerID	14	resourceTypeIDName
Reserved2	17	
HubDescriptionPointer	18	
HubVersionPointer	1C	
HealthState	20	repeaterHealthState
HealthTextPointer	24	repeaterHealthText
HealthDataPointer	28	repeaterHealthData
HealthDataLength	2C	
GroupsSupported	2E	repeaterGroupCapacity
GroupInfoTablePointer	30	
CapabilitiesBitMap	34	

Most of the entries in this table do not have IEEE 802.3 equivalents. The following is a brief description of these parameters. Refer to the HMI specification for more details.

Reserved
This is a 16-byte reserved field.

HubType
This 1-byte field is set to a '1' to signify that this Hub is a 10BASE-T hub.

Reserved1
This is a 1-byte reserved field.

MajorVersion
This field is a HIT major version number. The version number of this table is '1'.

MinorVersion
This field is a HIT minor version number. The current number of this table is '0'.

ManufacturerID
This is a 3-byte manufacturer's ID field, as specified in the IEEE 802.1F Recommended Practice draft document. The IEEE 802.1F equivalent attribute is the resourceInfo and is typically the three leading bytes of the manufacturer's MAC address.

Reserved2
This is a 1-byte reserved field.

HubDescriptionPointer
This field points to a text string describing this hub.

HubVersionPointer
This field points to a text string that describes the hub's version number.

HealthDataLength
This is the length in bytes of the preceding HealthDataPointer. Because the HealthData string contains binary information, it cannot be terminated by a '0'.

GroupInfoTablePointer
This is a 4-byte pointer to the Group Information Table (GIT).

CapabilitiesBitMap
This field is a 1-byte value indicating the level of management capabilities present in this HMI managed HUB. These numbers can be in the range from 0 to 3, with the following meanings:

0	Basic Control
1	Performance Monitoring
2	Address Tracking
3	Novell Extensions

2.3.3 Group Information Table (GIT)

Table 2-7 is the group information table. It is used to provide information related to specific groups of ports in the Repeater. Several of the IEEE 802.3 MIB attributes are passed through the Group Information Table instead of being transferred as managed objects.

Table 2-7 HMI Group Information Table

GIT #	GIT Object Name	IEEE Object Name
00	Installed	groupMap
01	Slot	
02	Ports	groupPortCapacity
04	Description	
08	ObjectIDPointer	
0C	ObjectIDLength	
0E	InstalledTime	
12	DriverWorkspace	

Slot
This is an optional 1-byte field to indicate the slot number of the selected Hub Group.

Description
This is a 4-byte pointer to a text string indicating information about this group.

ObjectIDPointer
This is a 4-byte pointer to the array of 16-bit words, that provides the manufacturer's group identification information. The format for this information is documented in the "Structure and Identification of Management Information for TCP/IP based Internets" enterprise subtree. This provides an exact specification of the class of group being managed.

ObjectIDLength
This 2-byte value is the length of the Object ID array in bytes.

InstalledTime
This 4-byte field is the system time when the group was last installed in the hub.

DriverWorkspace
This 10-byte field is available for the driver to use as temporary storage during it's execution.

2.3.4 HMI Notifications

In addition to HMI managed objects, and attributes passed through the HIT and GIT, the Novell HMI document specifies how various events in the Repeater can cause a notification message to be passed to the Hub manager. Table 2-8 lists the notification messages required by the HMI specification. These notifications correspond to their IEEE 802.3 equivalents.

Table 2-8 HMI Notifications

HMI Object Name	HMI #	IEEE 802.3 Object Name
HealthChange	1	repeaterHealth
GroupChange	2	groupMapChange
HubReset	3	repeaterReset

2.4 IETF MANAGED OBJECTS FOR IEEE 802.3 REPEATERS

As the IEEE 802.3 committee defined the repeater management objects, the IETF extended the definition to specify a protocol and encoding to use in the remote management of these objects. The IETF also defined additional objects for use in managing repeaters. The actual definition and syntax of these objects is defined in RFC 1368, "Definitions of Managed Objects for IEEE 802.3 Repeater Devices".

Table 2-9 lists the IETF 802.3 Repeater MIB 'get' and 'set' attributes, and cross references it to the IEEE 802.3 Repeater Management attributes and actions. Refer to RFC 1368 for a detailed description of each management object. Many of the SNMP managed objects have no corresponding analog in the IEEE 802.3 repeater MIB.

Table 2-9 SNMP to IEEE 802.3 Attribute Cross-reference

IETF Identification Code	IETF 802.3 Repeater MIB Reference	IEEE 802.3 Repeater MIB Reference
snmpDot3RptrMgt.1.1.1	rptrGroupCapacity	repeaterGroupCapacity
snmpDot3RptrMgt.1.1.2	rptrOperStatus	repeaterHealthState
snmpDot3RptrMgt.1.1.3	rptrHealthText	repeaterHealthText
snmpDot3RptrMgt.1.1.4	rptrReset	resetRepeater
snmpDot3RptrMgt.1.1.5	rptrNonDisruptTest	executeNonDisruptiveSelf TestAction
snmpDot3RptrMgt.1.1.6	rptrTotalPartitionedPorts	
snmpDot3RptrMgt.1.2.1.1.1	rptrGroupIndex	groupID
snmpDot3RptrMgt.1.2.1.1.2	rptrGroupDescr	
snmpDot3RptrMgt.1.2.1.1.3	rptrGroupEntry	
snmpDot3RptrMgt.1.2.1.1.4	rptrGroupOperStatus	
snmpDot3RptrMgt.1.2.1.1.5	rptrGroupLastOperStatusChange	
snmpDot3RptrMgt.1.2.1.1.6	rptrGroupPortCapacity	groupPortCapacity
snmpDot3RptrMgt.1.3.1.1.1	rptrPortGroupIndex	
snmpDot3RptrMgt.1.3.1.1.2	rptrPortIndex	portID
snmpDot3RptrMgt.1.3.1.1.3	rptrPortAdminStatus	portAdminState portAdminControl
snmpDot3RptrMgt.1.3.1.1.4	rptrPortAutoPartitionState	autoPartitionState
snmpDot3RptrMgt.1.3.1.1.5	rptrPortOperStatus	

Table 2-9 SNMP to IEEE 802.3 Attribute Cross-reference (continued)

IETF Identification Code	IETF 802.3 Repeater MIB Reference	IEEE 802.3 Repeater MIB Reference
snmpDot3RptrMgt.2.1.1	rptrMonitorTransmitCollisions	transmitCollisions
snmpDot3RptrMgt.2.2.1.1.1	rptrMonitorGroupIndex	
snmpDot3RptrMgt.2.2.1.1.2	rptrMonitorGroupTotalFrames	
snmpDot3RptrMgt.2.2.1.1.3	rptrMonitorGroupTotalOctets	
snmpDot3RptrMgt.2.2.1.1.4	rptrMonitorGroupTotalErrors	
snmpDot3RptrMgt.2.3.1.1.1	rptrMonitorPortGroupIndex	
snmpDot3RptrMgt.2.3.1.1.2	rptrMonitorPortIndex	portID
snmpDot3RptrMgt.2.3.1.1.3	rptrMonitorPortReadableFrames	readableFrames
snmpDot3RptrMgt.2.3.1.1.4	rptrMonitorPortReadableOctets	readableOctets
snmpDot3RptrMgt.2.3.1.1.5	rptrMonitorPortFCSErrors	frameCheckSequenceErrors
snmpDot3RptrMgt.2.3.1.1.6	rptrMonitorPortAlignmentErrors	alignmentErrors
snmpDot3RptrMgt.2.3.1.1.7	rptrMonitorPortFrameTooLongs	framesTooLong
snmpDot3RptrMgt.2.3.1.1.8	rptrMonitorPortShortEvents	shortEvents
snmpDot3RptrMgt.2.3.1.1.9	rptrMonitorPortRunts	runts
snmpDot3RptrMgt.2.3.1.1.10	rptrMonitorPortCollisions	collisions
snmpDot3RptrMgt.2.3.1.1.11	rptrMonitorPortLateEvents	lateEvents
snmpDot3RptrMgt.2.3.1.1.12	rptrMonitorPortVeryLongEvents	veryLongEvents
snmpDot3RptrMgt.2.3.1.1.13	rptrMonitorPortDataRate Mismatches	dataRateMismatches
snmpDot3RptrMgt.2.3.1.1.14	rptrMonitorPortAutoPartitions	autoPartitions
snmpDot3RptrMgt.2.3.1.1.15	rptrMonitorPortTotalErrors	
snmpDot3RptrMgt.3.1.1.1.1	rptrAddrTrackGroupIndex	
snmpDot3RptrMgt.3.1.1.1.2	rptrAddrTrackPortIndex	portID
snmpDot3RptrMgt.3.1.1.1.3	rptrAddrTrackLastSourceAddress	lastSourceAddress
snmpDot3RptrMgt.3.1.1.1.4	rptrAddrTrackSourceAddrChanges	sourceAddressChanges
snmpDot3RptrMgt.3.2	rptrAddrTrackGroupInfo	
snmpDot3RptrMgt.3.3	rptrAddrTrackPortInfo	

Table 2-10 lists the IETF traps, which correspond to the IEEE management notifications objects.

Table 2-10 SNMP Trap to IEEE 802.3 Notification Cross-Reference

IETF Identification Code	IETF 802.3 Repeater MIB Reference	IEEE 802.3 Repeater MIB Reference
snmpDot3RptrMgt.1	rptrHealth	repeaterHealth
snmpDot3RptrMgt.2	rptrGroupChange	groupMapChange
snmpDot3RptrMgt.3	rptrResetEvent	repeaterReset

SECTION

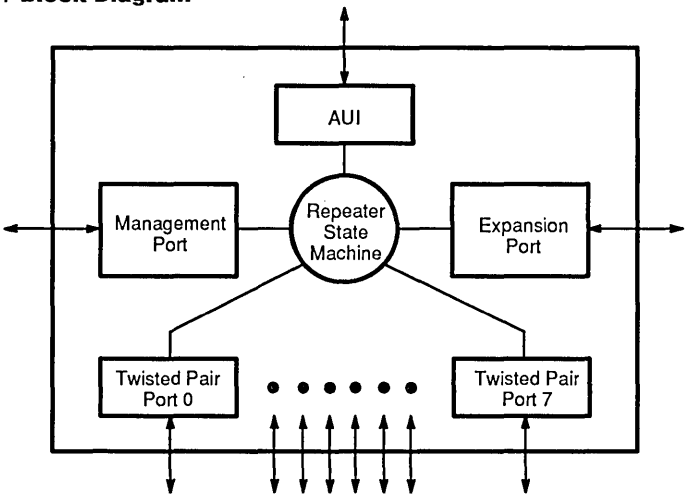
3 IMR/IMR+ OVERVIEW

3.1 FUNCTIONAL DESCRIPTION

The Am79C981 Integrated Multiport Repeater Plus device is a single chip implementation of an IEEE 802.3/Ethernet repeater (or hub). In addition to the eight integral 10BASE-T ports plus one AUI port comprising the basic repeater, the IMR+ chip also provides the hooks necessary for complex network management and diagnostics. The IMR+ device is also expandable, enabling the implementation of high port count repeaters based on several IMR+ devices.

The IMR+ device interfaces directly with AMD's Am79C987 Hardware Implemented Management Information Base (HIMIB) device to allow a fully managed multiport repeater to be implemented as specified by the IEEE 802.3 Layer Management for 10 Mb/s Baseband Repeaters Standard. When the IMR+ and HIMIB devices are used as a chip set, the HIMIB device maintains complete repeater and per port statistics which can be accessed on demand by a microprocessor through a simple 8-bit parallel port.

Figure 3-1 IMR+ Block Diagram



The IMR+ device differs from the original IMR chip in only a few minor internal details. The IMR+ device is pin, software and timing compatible with the IMR device, and may be used as a direct replacement in an existing IMR-based design. Most of the enhancements in the IMR+ device relate to the provision of additional internal status information, which is primarily included for use by the companion HIMIB device. The HIMIB device requires this enhanced status information in order to allow implementation of a fully managed repeater, compliant to the IEEE 802.3 Layer Management for 10 Mb/s Baseband Repeaters Standard. The implementation of managed repeaters is covered in detail in Section 5 of this manual, and the complete definition of all changes between the

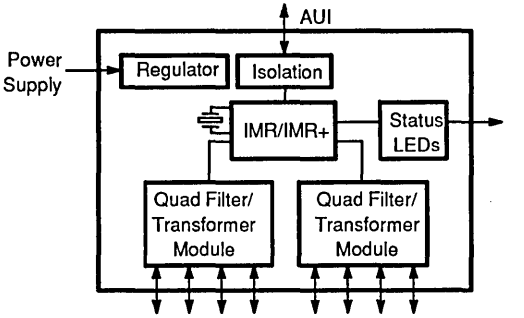
IMR and IMR+ devices is contained in this section under "Differences Between the IMR and IMR+ Devices".

From the perspective of a simple low cost, unmanaged repeater application, the primary additional feature offered by the IMR+ device over and above that of the original IMR device, is the provision of a "Minimum Mode", designed to minimize the external support logic to provide simple diagnostic and status related indicators (LEDs). The Minimum Mode is explained in more detail in this section under "Minimum Mode Operation".

3.2 IMR/IMR+ BASED "VELCRO™ HUB" DESIGN

For low end systems, the IMR combined with a power supply, crystal and EMI/RFI filter/transformer modules, effectively produces a fully operational 10BASE-T repeater, with an AUI port to allow connection to an existing 10BASE2/5 coax backbone.

Figure 3-2 Simple "Velcro™ Hub" Example

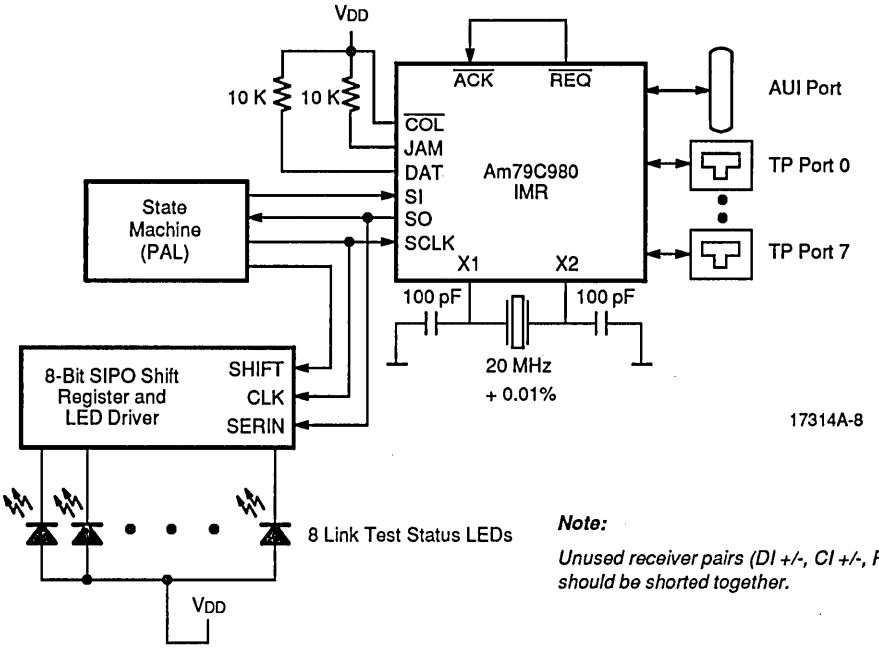


3.2.1 IMR-Based "Velcro™ Hub" Design

A "Velcro Hub" application example is shown in Figure 3-3. The external PAL connects to the Management Port of the IMR device, and implements a simple state machine. This is used to configure the programmable options of the IMR device at power up (if the default options need to be changed), and then continuously clocks in Management Port "Get" requests on the SI pin. It also clocks out the results on the SO pin to obtain internal status information (such as the Link Status, Partitioning or Polarity). The result of the "Get" request, output on the SO pin by the IMR device, is shifted into the external serial-to-parallel converter, and used to drive LED status indicators, using a simple pulse stretch and driver circuit.

The state machine can be enhanced to allow external selection (through a front panel mounted switch or other selection mechanism) of different Management Port "Get" requests, such that the same LEDs can be used to display various status information. The "IMR Velcro™ Hub Board" is available from AMD as an example design kit, which details these concepts.

Figure 3-3 "Velcro Hub" Design Using the IMR Device



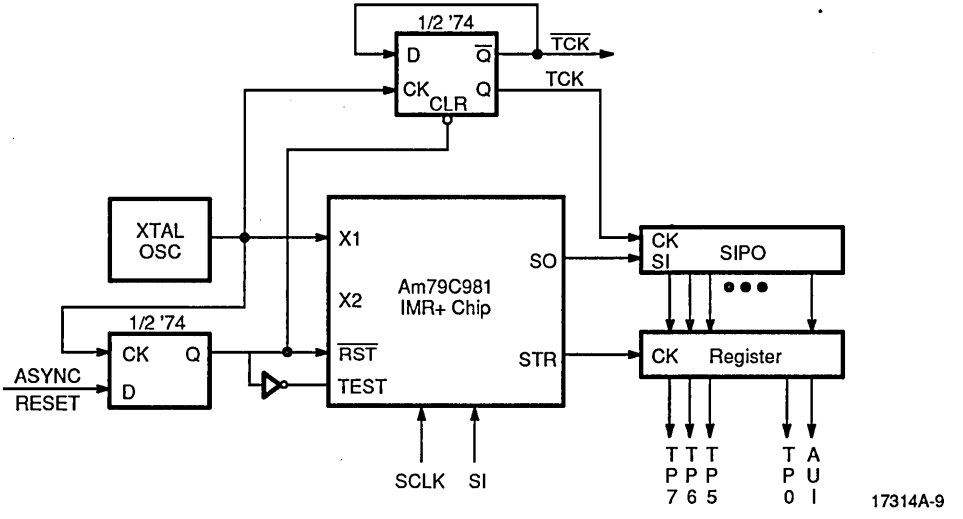
Note:

Unused receiver pairs (DI +/-, CI +/-, RXD_n +/-) should be shorted together.

3.2.2 IMR+ Based "Velcro Hub" Design

The design of a simple repeater using the IMR+ device is simplified by use of the "Minimum Mode", which allows status information to be continuously output from the Management Port of the IMR+ device, without the need for an external state machine. Figure 3-4 shows an application example employing the IMR+ device in Minimum Mode.

Figure 3-4 "Velcro" Hub Design Using the IMR+ Device



In this configuration, the IMR+ device will continuously output one of four status conditions on the SO pin, dependent on the programming of the TEST and SI pins. The programming of SI and TEST allows any one of the following status indications to be reported on SO:

AUI Port Loop Back Status (1-bit) and Twisted Pair Port Link Status (8-bits)
AUI/Twisted Pair Port Partitioning Status (9-bits)
AUI Port SQE Test Error Status (1-bit) and Twisted Pair Polarity Status (8-bits)
AUI/Twisted Pair Port Bit Rate Error Status (9-bits)

The state of the SI and TEST pins can be changed at any time to select alternative status of the SO pin.

The data on the SO pin is output as a 10-bit serial stream (AUI status first, followed by TP0 status, TP1 status, etc.). A blank period of 100 ns will occur after the 9 status bits have been output, during which time the STR pin will be active, delineating the start/end of each "status window". The SO output stream can be directly input to a serial-to-parallel convertor, and used to drive LED status indicators using appropriate latches and drivers. The details of the programming and timing of Minimum Mode are covered in the following section.

Since the data presented on the SO pin may be valid for only one "status window" (so an individual bit, such as the AUI Loop Back Error, may be active for only 100 ns), external pulse stretch circuitry should be included to ensure that the LED indicators are visible.

Note that the normal "Get" and "Set" capabilities of the IMR+ Management Port are unavailable when Minimum Mode is selected.

3.2.3 Minimum Mode Operation

The IMR+ Minimum Mode supports designs of low end, unmanaged repeaters. This mode uses minimal additional support logic to display the following status indicators:

- AUI Port Loop Back Status and Twisted Pair Port Link Status
- AUI/Twisted Pair Port Partitioning Status
- AUI Port SQE Test Error Status and Twisted Pair Port Receiver Polarity Status
- AUI/Twisted Pair Port Bit Rate Error Status

Additionally the Minimum Mode of the IMR+ chip supports automatic receive polarity detection/correction without the addition of external logic.

The IMR+ device determines that Minimum Mode is selected by monitoring the state of the TEST pin while $\overline{RST}$ is asserted. If TEST is HIGH (asserted), while reset is active ($\overline{RST}$ LOW) it enters Minimum Mode. Additionally the state of the SI pin during reset determines if the IMR+ device is to be programmed for Automatic Polarity Detection/ Correction.

The TEST input must be deasserted on the rising edge of reset. A maximum delay of 100 ns is allowed to account for propagation delay.

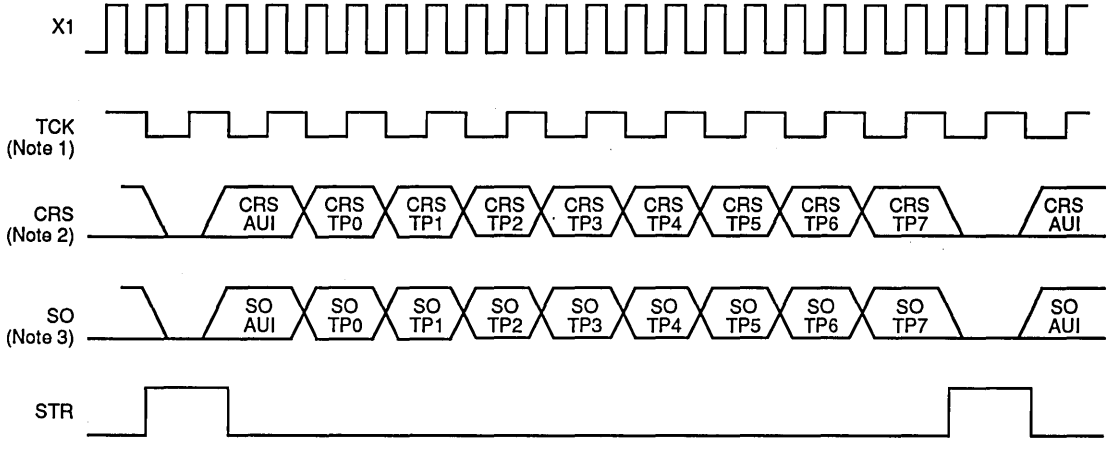
TEST	SI	Function
0	0	Normal Management Mode
0	1	Normal Management Mode
1	0	Minimum Mode, Receive Polarity Detection/Correction disabled.
1	1	Minimum Mode, Receive Polarity Detection/Correction enabled.

In Minimum Mode, the SO pin is used to serially output the IMR+ chip's status information based on the state of the SI and SCLK pins:

SCLK	SI	SO Output
0	0	AUI Port SQE Test Error Status + TP Port Receive Polarity Status
0	1	Bit Rate Error Status (All ports)
1	0	AUI Port Loop Back Status + TP Ports Link Status
1	1	Partitioning Status (All ports)

The output format and timing is identical to that of the CRS and STR when the IMR+ device is used in the stand alone mode (see Figure 3-5).

Figure 3-5 Management Port Minimum Mode and Port Activity Monitor Signal Relationship



17314A-10

Notes:

1. *Externally generated signal illustrates internal IMR+ chip clock phase relationship.*
2. *CRS timing with the C-bit cleared (IMR+ Chip Programmable Options)*
3. *For Minimum Mode*

When SI = 0, SO outputs the related AUI status bits (Loop Back Error or SQE Test Error, depending on the value of SCLK), followed by the 8 TP status bits (receive polarity or Link Status, depending on the value of SCLK). The TP status bits are output in order from port 0 to port 7.

When SI = 1, the Port Partitioning Status or Port Bit Rate Error Status indicators are output (depending on the value of SCLK). The AUI is output first, followed by the TP port indicators starting with port 0.

The timing of the SO output matches that of the Port Activity Monitor (PAM). The state of the SI and SCLK pins is checked at the end of every cycle after all port status indicators have been output. The rising edge of the X1 clock, occurring before falling edge of STR, is used to strobe in the state of the SI and SCLK pins.

The IMR+ device may be programmed into Minimum Mode only at reset, and cannot be modified until a subsequent reset.

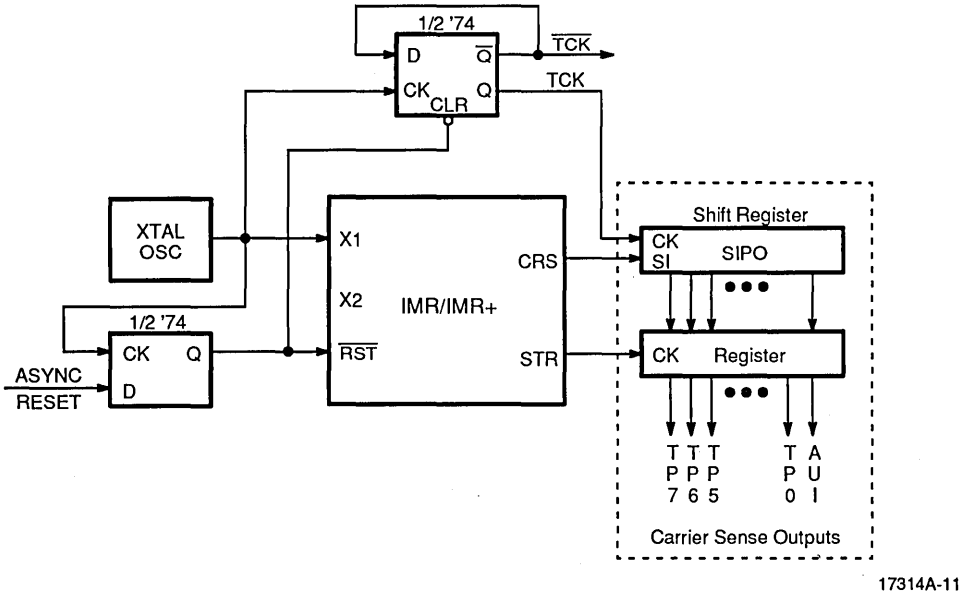
3.3 PORT ACTIVITY MONITOR (PAM) OPERATION

Two IMR+ device pins, CRS and STR, are used to serially output the state of the internal Carrier Sense signals from the AUI and the eight twisted pair ports. This function together with external hardware or software can be used to monitor repeater receive and collision activity. The accuracy of the CRS signals is 10 Bit Times (BT) (1 µs). A transition to active state by any of the internal carrier sense bits that lasts for less than 10BT is latched internally and is used to set the appropriate bit during the next sample period. See Figure 3-5 for an illustration of the timing of the port activity monitor.

3.3.1 Stand-Alone IMR/IMR+ Device (With STR Output)

Figure 3-6 shows how external hardware can be employed to convert the serial CRS bit stream into a parallel format suitable for a "receive activity" display. Note that since the data presented on the CRS pin may be valid for only a single 100 ns period, and the fact that a maximum Ethernet packet length is only ~1.2 ms in duration, external pulse stretch circuitry should be included to ensure that the LED indicators are visible.

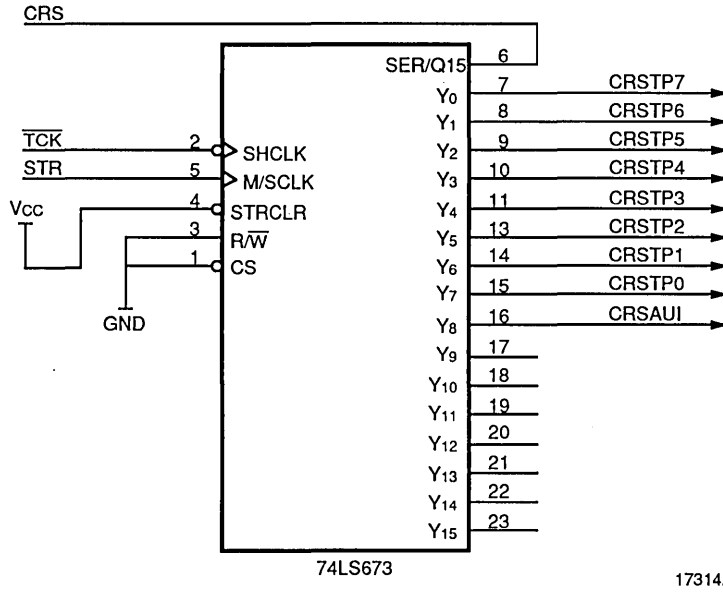
Figure 3-6 Port Activity Monitor



17314A-11

Figure 3-7 shows one possible hardware implementation of the interface between the IMR+ device and some LED displays. It should be stressed that this is only an example, and is not intended to represent the most cost-effective method of implementing this function. This implementation may also be used to display the data output on the SO pin in Minimum Mode.

Figure 3-7 Example Implementation of Port Activity Monitor Registers



3.3.2 IMR+ With HIMIB Device (No STR Output)

When the IMR+ chip is used with the HIMIB device, the STR pin of the IMR+ chip becomes an input. See Section 5 under "Managed Repeater Status Indicators" for details on how to implement the port activity monitor in this case.

3.4 LINK TEST STATE MACHINE DESCRIPTION

The link test function in the IMR and IMR+ devices is implemented as specified by the 10BASE-T Standard. During periods of transmit pair inactivity, "Link Test" pulses will be sent over the twisted pair medium every 8 ms–17 ms, to indicate medium integrity.

When the link test function is enabled, the absence of Link Test pulses and receive data on the RXD± pair of an IMR+ chip's 10BASE-T port will cause the port to go into a Link Fail state. In the Link Fail state, data transmission, data reception and the collision detection functions are disabled, and remain disabled until valid data or 4 consecutive Link Test pulses appear on the RXD± pair. During Link Fail, issuing the command to get Link Test Status of TP Ports using the Management Port will return the state for the port as zero (cleared). When the link is identified as functional, the corresponding status will be reported as one. Note that in the case of the IMR device, if a 10BASE-T port is disabled, its corresponding status will always be returned as "Link Pass". In the case of the IMR+ device, the status of the link will be reported accurately even if the port is disabled.

In order to inter-operate with systems which do not implement Link Test pulses, this function can be disabled by issuing the Disable Link Test Function command using the

IMR+ Management Port (with the appropriate individual port identified within the command). With link test disabled, the data driver, receiver and loopback functions as well as collision detection remain enabled irrespective of the presence or absence of data or Link Test pulses on the RXD± pair. Note that the IMR/IMR+ device will continue to transmit Link Test pulses, regardless of the state of the Disable Link Test function, or whether the port is disabled or auto-partitioned. This ensures that the MAU at the opposite end of the link segment will interoperate, regardless of whether it requires Link Test pulses or not.

3.5 RECEIVE POLARITY DETECTION/CORRECTION ALGORITHM

The IMR/IMR+ device receive function includes the ability to invert the polarity of the signals appearing at the RXD± pair if the polarity of the received signal is reversed (such as in the case of a wiring error). This feature allows data packets received from a reverse wired RXD± input pair to be corrected in the IMR/IMR+ device prior to re-transmission via the twisted pair, AUI or Expansion Port interfaces. Following reset, the polarity detection/correction function must be explicitly enabled for the 10BASE-T ports. This is accomplished by issuing the "Set" command Enable Automatic Receiver Polarity Reversal using the IMR/IMR+ Management Port, or by programming the IMR+ device into the Minimum Mode. Any port with polarity detection/correction enabled which is in the Link Fail state, will detect the receive polarity based on the polarity of subsequent packets with a valid End of Transmit Delimiter (ETD).

When in the Link Fail state, the IMR+ device will recognize Link Test pulses of positive polarity. Positive Link Test pulses are defined as received signal with a positive amplitude greater than 520 mV with a pulse width of 60 ns–200 ns. This positive excursion may be followed by a negative excursion. This definition is consistent with the expected received signal at a correctly wired receiver when a Link Test pulse which fits the template of Figure 14-12 in the 10BASE-T Standard is generated at a transmitter and passed through 100 m of twisted pair cable. Negative Link Test pulses are ignored.

Exit from the Link Fail state is made due to the reception of four consecutive positive Link Test pulses, which are spaced not closer than 2.04 ms–4.1 ms (nominal) and not greater than 65.5 ms–131.1 ms apart. When the IMR/IMR+ enters the Link Pass state, the polarity detection/correction algorithm will remain "armed" until two consecutive packets with valid ETD of identical polarity are detected. When "armed", the receiver is capable of changing the initial or previous polarity configuration based on the most recent ETD polarity.

On receipt of the first packet with valid ETD following Link Fail, the IMR+ device will utilize the inferred polarity information to configure its RXD± input, regardless of its previous state. On receipt of a second packet with a valid ETD with correct polarity, the detection/correction algorithm will "lock-in" the received polarity. If the second (or subsequent) packet is not detected as confirming the previous polarity decision, the most recently detected ETD polarity will be used as the default. Note that packets with invalid ETD have no effect on updating the previous polarity decision. Once two consecutive packets with valid ETD have been received, the IMR+ device will disable the detection/correction algorithm until either a Link Fail condition occurs or the port is disabled and re-enabled (when the port is forced into Link Fail) using the Management Port.

If normal (correct) polarity is detected for a port, issuing the command to get Receive Polarity Status of TP Ports using the Management Port will return the state for the port as zero (cleared). If the reversed polarity has been detected/corrected, the corresponding status will be reported as one.

3.6 ALTERNATE RECONNECTION ALGORITHM

The AUI port and each of the TP ports can be partitioned when experiencing excessive frequency or duration of collisions. Excessive frequency of collisions is defined as more than 31 consecutive collisions on a single port. Excessive duration of collision is defined as the SQE signal active for more than 1024-bit times.

Reconnection is the process whereby the port is returned to operational status after partitioning. The IMR+ device supports two reconfiguration algorithms. The standard IEEE 802.3 algorithm is the default algorithm, and is activated when the IMR+ chip is reset. This algorithm allows a partitioned port to be reconnected if it is able to transmit or receive data for a period of 512 bit times without experiencing a collision.

The Alternate Reconnection Algorithm will reconnect the partitioned port only if the port is able to transmit data for a period of 512 bit times without a collision. This mode is programmed through the management port. Once the Alternate Reconnection Algorithm is programmed, the IMR+ chip must be hardware reset to return it to the default (standard 802.3) algorithm.

3.7 INTERACTION BETWEEN PORT DISABLE AND PORT AUTOPARTITION

Disabling a port forces the Autopartition State Machine of that port into the Idle State (Port Reconnected). Therefore, a partitioned port may be reconnected by first disabling and then re-enabling the port. This is in accordance with the 802.3 Repeater Management Standard requirements.

3.8 IMR/IMR+ MANAGEMENT PORT

Table 3-1 lists the Management Port command op-codes for both the IMR and IMR+ devices. Additionally, the table lists the command formats, and the response codes for the management 'Get' commands. This section provides a detailed description of the IMR+ device command op-codes.

Table 3-1 IMR/IMR+ Management Commands

Command Type	Command Description	IMR/IMR+	Command Code	Response (Get Only)
Set	IMR+ Chip Programmable Options	IMR+ Only	0000 1CSA	
Set	Alternate AUI Port Partitioning Algorithm	IMR/IMR+	0001 1111	
Set	Alternate TP Port Partitioning Algorithm	IMR/IMR+	0001 0000	
Set	AUI Port Disable	IMR/IMR+	0010 1111	
Set	AUI Port Enable	IMR/IMR+	0011 1111	
Set	TP Port Disable	IMR/IMR+	0010 ####	
Set	TP Port Enable	IMR/IMR+	0011 ####	
Set	Disable Link Test Function (per TP port)	IMR/IMR+	0100 0###	
Set	Enable Link Test Function (per TP port)	IMR/IMR+	0101 0###	
Set	Disable Auto Polarity Correction (per TP port)	IMR/IMR+	0110 0###	
Set	Enable Auto Polarity Correction (per TP port)	IMR/IMR+	0111 0###	
Get	AUI Port Status (BSL Cleared)	IMR+ Only	1000 1111	PBSL 0000
Get	AUI Port Status (SL Cleared)	IMR+ Only	1000 1011	PBSL 0000
Get	AUI Port Status (B Cleared)	IMR+ Only	1000 1101	PBSL 0000
Get	AUI Port Status (None Cleared)	IMR+ Only	1000 1001	PBSL 0000
Get	TP Port Partitioning Status	IMR/IMR+	1000 0000	C7 ... C0
Get	Bit Rate Error Status of TP Ports	IMR+ Only	1010 0000	E7 ... E0
Get	Link Test Status of TP Ports	IMR/IMR+	1101 0000	L7 ... L0
Get	Receive Polarity Status of TP Ports	IMR/IMR+	1110 0000	P7 ... P0
Get	MJLP Status	IMR+ Only	1111 0000	M000 0000
Get	Version	IMR IMR+	1111 1111 1111 1111	XXXX 0000 XXXX 0001
Get	AUI Port Status	IMR Only	1000 1111	P000 0000

3.8.1 Management Port

The IMR+ device management functions are enabled when the TEST pin is tied LOW. The management commands are byte oriented data and are input serially on the SI pin. Any responses generated during execution of a management command are output serially in a byte-oriented format by the IMR+ device on the SO pin. Both the input and output data streams are clocked with the rising edge of the SCLK pin. The serial command data stream and any associated results data stream are structured in a manner similar to the RS232 serial data format, i.e., one Start Bit followed by eight Data Bits.

The externally generated clock at the SCLK pin can be either a free running clock synchronized to the input bit patterns or a series of individual transitions meeting the

setup and hold times with respect to the input bit pattern. If the latter method is used, it is to be noted that 20 SCLK clock transitions are required for proper execution of management commands that produce SO data, and that 14 SCLK clock transitions are needed to execute management commands that do not produce SO data.

3.8.2 Management Commands

The following section details the operation of each management command available in the IMR+ chip. In all cases, the individual bits in each command byte are shown with the MSB on the left and the LSB on the right. Data bytes are received and transmitted LSB first and MSB last. See Table 3-1 for a summary of the management commands.

SET (Write) Opcodes

IMR+ Chip Programmable Options

SI data: 0000 1CSA
SO data: None

IMR+ Chip Programmable Options can be enabled (disabled) by setting (resetting) the appropriate bit in the command string. The three programmable bits are: **C**—CI Reporting; **S**—AUI SQE Test Mask, and **A**—Alternative Port Activity Monitor (PAM) Function. These options can be enabled (disabled) by setting (resetting) the appropriate bit in the command string.

When writing to this register through the Am79C987 HIMIB device, the A and C bits should not be changed (A=0, C=1).

C—CI Reporting

Setting this bit alters the function of the STR pin. In this mode, the STR pin becomes an input in response to the AMD's Am79C987 HIMIB device. Upon deassertion of $\overline{RST}$, the HIMIB automatically sets this bit following IMR+ device type detection.

When this mode is selected, the CRS output bit string format is modified to include CI carrier bit (in addition to AUI carrier). This bit occupies the bit position immediately preceding the AUI bit in the CRS bit string (10 bits) output. Note that the AUI carrier bit gets asserted if either the CI or DI signal pairs are active.

S—AUI SQE Test Mask

Setting this bit allows the IMR+ chip to ignore activity on the CI signal pair, during the SQE Test window following a transmission on the AUI port. This event occurs when the attached MAU has the SQE Test option enabled, therefore generating a burst of CI activity following every transmission. This is interpreted by the IMR+ device as a collision, causing the IMR+ device to generate a full Jam pattern. Although the MAU attached to a repeater is required not to have its SQE Test function active, this is a common installation error, causing difficulty in diagnosing network throughput problems.

The SQE Test Window, as defined by the IEEE 802.3 (Section 7.2.2.2.4), is from 6-bit times to 34-bit times (0.6 μs to 3.4 μs). This includes delay introduced by a 50 m AUI. CI activity that occurs outside this window is not ignored and is treated as true collision.

Note that enabling this function does not prevent the reporting of this condition by the IMR+ device (S bit in GET AUI Port Status) since the two functions operate independently.

A—Alternative Port Activity Monitor (PAM) Function

Setting the Alternative Port Activity Monitor Function allows the PAM function to be altered such that the Carrier Sense data is presented unmodified. In default operation the PAM output (Carrier Sense bit in the CRS bit stream) is masked if the port is either disabled or partitioned. This does not allow the Repeater Management software to sense activity on all segments at all times. The ability to monitor partitioned or disabled ports allows fault tolerant features to be built into the Repeater Management software.

Alternate AUI Port Partitioning Algorithm

SI data: 00011111
SO data: None

The AUI port Partitioning/Reconnection scheme can be programmed for the alternate (transmit only) reconnection algorithm by invoking this command. To return the AUI back to the standard (transmit or receive) reconnection algorithm, it is necessary to reset the IMR+ device. The standard partitioning algorithm is selected upon reset.

Alternate TP Ports Partitioning Algorithm

SI data: 00010000
SO data: None

The TP ports Partitioning/Reconnection scheme can be programmed for the alternate (transmit only) reconnection algorithm by invoking this command. All TP ports are affected as a group by this command. To return the TP ports back to the standard (transmit or receive) reconnection algorithm, it is necessary to reset the IMR+ device. The standard partitioning algorithm is selected upon reset.

AUI Port Disable

SI data: 00101111
SO data: None

The AUI port will be disabled upon receiving this command. Subsequently, the IMR+ chip will ignore all inputs (Carrier Sense and SQE) appearing at the AUI port and will not transmit any data or Jam Sequence on the AUI port. Issuing this command will also cause the AUI port to have its internal partitioning state machine forced to its idle state. Therefore, a Partitioned Port may be reconnected by first disabling and then re-enabling the port.

AUI Port Enable

SI data: 00111111
SO data: None

This command enables a previously disabled AUI port. Note that a partitioned AUI port may be reconnected by first disabling (AUI Port Disable command) and then re-enabling the port with this command. All ports are enabled upon reset.

TP Port Disable

SI data: 00100###
SO data: None

(### is TP port number)

The TP port designated in the command byte will be disabled upon receiving this command. Subsequently, the IMR+ device will ignore all inputs appearing at the disabled port's receive pins and will not transmit any data or JAM Sequence on that port's transmit pins. Issuing this command will also cause a TP port to have its partitioning state machine returned to its Idle State (Port Reconnected). Therefore, a partitioned port may be reconnected by first disabling and then re-enabling the port. The disabled port will continue to report correct Link Test Status.

TP Port Enable

SI data: 00110###
SO data: None

(### is TP port number)

This command enables a previously disabled TP port. Re-enabling a disabled port causes the port to be placed into Link Test Fail state. This ensures that packet fragments received on the port are not repeated to the rest of the network. Note that to force a TP port into the Link Fail state and/or to reconnect a partitioned TP port, the port should first be disabled (TP Port Disable command) and then re-enabled with this command. All ports are enabled upon reset.

Disable Link Test Function of a TP Port

SI data: 01000###
SO data: None

(### is TP port number)

This command disables the Link Test Function at the TP port designated in the command byte, i.e., the TP port will no longer be disconnected due to Link Fail. A TP port which has its Link Test Function disabled will continue to transmit Link Test pulses. If a twisted pair port has Link Test disabled, then reading the Link Test Status indicates it being in Link Test Pass.

Enable Link Test Function of a TP Port

SI data: 01010###
SO data: None

(### is TP port number)

This command re-enables the Link Test Function in the TP port designated in the command byte. This command executes only if the designated TP port has had the Link Test Function disabled by the Disable Link Test Function command. Otherwise, the command is ignored. Link Test is enabled upon reset.

Disable Automatic Receiver Polarity Reversal

SI data: 01100###
SO data: None

(### is TP port number)

This command disables the Automatic Receiver Polarity Reversal Function for the TP port designated in the command byte. If this function is disabled on a TP port with reverse polarity (due to a wiring error), then the TP port will fail Link Test due to the reversed polarity of the Link Pulses. If the Link Test Function is also disabled on the TP port, then the received reversed polarity packets would be repeated to all other network ports in the IMR+ chip as inverted data. Automatic Polarity reversal is disabled upon reset.

Enable Automatic Receiver Polarity Reversal

SI data: 01110###
SO data: None

(### is TP port number)

This command enables the Automatic Receiver Polarity Reversal Function for the TP port designated in the command byte. If enabled in a TP port, the IMR+ chip will automatically invert the polarity of that TP port's receiver circuitry if the TP port is detected as having reversed polarity (due to a wiring error). After reversing the receiver polarity, the TP port could then receive subsequent (reverse polarity) packets correctly.

GET (Read) Opcodes

AUI Port Status

SI data: 10001111
SO data: PBSL0000

The combined AUI status allows a single instruction to be used for monitoring AUI port. The four status bits reported are:

P Partitioning Status. This bit is 0 if the AUI port is partitioned and 1 if connected.

B Bit Rate Error. This bit is set to 1 if there has been an instance of FIFO Overflow or Underflow, caused by data received at the AUI port. This bit is cleared when the status is read.

S SQE Test Status. This bit is set to 1 if SQE Test is detected by the IMR+ chip. This bit is cleared when the status is read. A MAU attached to a repeater must have SQE Test disabled. This bit is set even if the AUI port is disabled or partitioned.

L Loop Back Error. The MAU attached to the AUI is required to loopback data transmitted to DO onto the DI circuit. If loopback carrier is not detected by the IMR+ device, then this bit is set to 1 to report this condition. This bit is cleared when the status is read. For a repeater this is the only indication of a broken or missing MAU.

Alternate AUI Port Status

SI data: 10001111
SO data: PBSL0000

There are three further variations of the above command, allowing selective clearing of a combination of B, S, and L bits. They are primarily included for use by the HIMIB chip. These are:

Alternative 1.

SI data: 10001011
SO data: PBSL0000

B is not cleared. S and L are cleared.

Alternative 2.

SI data: 10001101
SO data: PBSL0000

S and L are not cleared. B is cleared.

Alternative 3.

SI data: 10001001
SO data: PBSL0000

None of S, B and L are cleared.

TP Port Partitioning Status

SI data: 10000000
SO data: P7.....P0

Pn = 0 TP port n partitioned

Pn = 1 TP port n connected

The partitioning Status of all eight TP ports are accessed by this command. If a port is disabled, reading it partitioning status will indicate that it is connected.

Bit Rate Error Status of TP Ports

SI data: 10100000
SO data: E7.....E0

This allows a single command to be used to report the Bit Rate Error Status condition (FIFO Overflow or Underflow) of all Twisted Pair ports. The 8 bits of the output pattern correspond to each of the 8 TP ports, with least significant bit corresponding to port 0.

The status bit for a port is set to 1 if there has been an instance when data received from that port has caused a FIFO error.

All status bits stay set until the status is read.

Link Test Status of TP Ports

SI data: 11010000
SO data: L7.....L0

Ln = 0 TP Port n in Link Test Fail
Ln = 1 TP Port n in Link Test Pass

The Link Test Status of all eight TP ports are accessed by this command. A disabled port continues to report correct Link Test Status. Re-enabling a disabled port causes the port to be placed into Link Test Fail state. This ensures that packet fragments received on the port are not repeated to the rest of the network.

Receive Polarity Status of TP Ports

SI data: 11100000
SO data: P7.....P0

Pn = 0 TP Port n Polarity Correct
Pn = 1 TP Port n Polarity Reversed

The status of all eight TP port polarities are accessed with this command. The IMR+ chip has the ability to detect and correct reversed polarity on the TP ports' RXD+/- pins. If the polarity is detected as reversed for a TP port, then the IMR+ chip will set the appropriate bit in this command's result byte only if the Polarity Reversal Function is enabled for that port.

MJLP Status

SI data: 11110000
SO data: M00000000

Each IMR+ chip contains an independent MAU Jabber Lock Up Protection Timer. The timer is designed to inhibit the IMR+ device transmit function, if it has been transmitting continuously for more than 65536 Bit Times. The MJLP Status bit (M) is set to 1 if this happens. This bit remains set and is only cleared when the MJLP status is read by using this command.

Version

SI data: 11111111
SO data: XXXX0001

This command (1111 1111) can be used to determine the device version.

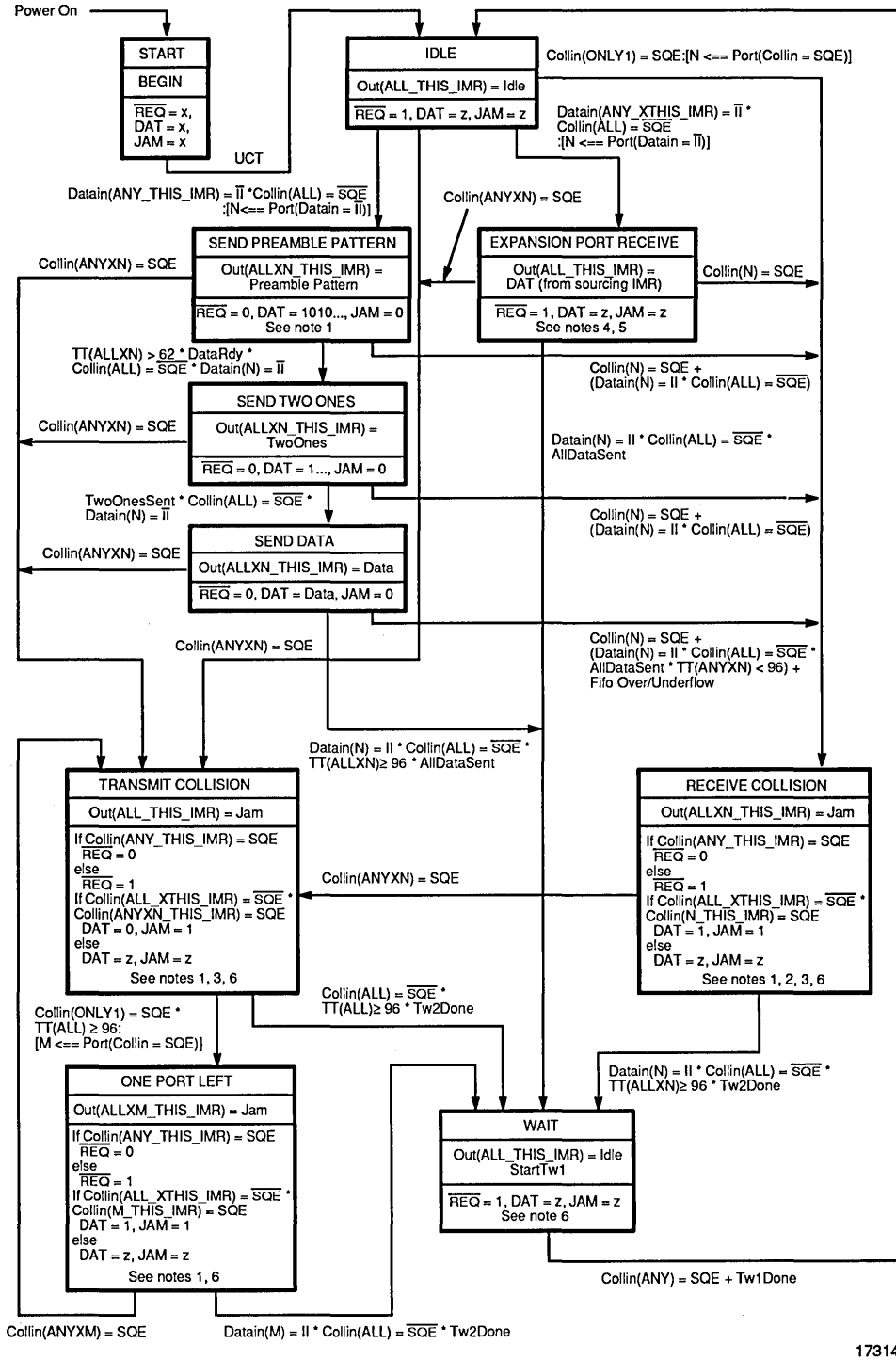
The IMR+ chip responds by the bit pattern: XXXX 0001

The IMR chip (Am79C980) responds by the bit pattern: XXXX 0000

3.9 IMR+ DEVICE REPEATER STATE MACHINE DESCRIPTION

The state diagram in Figure 3-8 describes the relationship between the IMR+ repeater state machine and the IMR+ Expansion Port. The diagram is similar to the IEEE 802.3 Repeater Unit State Diagram (see ISO/IEC 8802-3: 1990 (E) ANSI/IEEE Std 802.3-1990 Edition, Figure 9-2). Each IMR+ device contains an independent implementation of the state machine. When multiple IMR+ devices are interconnected to form a single, high port count repeater, it is the Expansion Port that keeps these state machines synchronized with each other.

Figure 3-8 IMR+ Device State Machine



17314A-13

Referring to Figure 3-8, the upper box in each block contains the name of the state. The center box refers to the output at the AUI and TP ports. The lower box describes the state of the $\overline{REQ}$, DAT, and JAM output pins for the IMR+ device with this state machine embedded.

There are several items to keep in mind when interpreting the state diagram:

When DAT = z, and JAM = z, DAT and JAM are either in high-impedance ($\overline{ACK}$ = 1) or input ($\overline{ACK}$ = 0).

The meaning of "ALL", "ANY", "ANYXN", "ANYXM", etc. refers to the repeater as a unit (a single repeater can be formed by connecting multiple IMR+ devices together). The meaning is modified if "_THIS_IMR" or "_XTHIS_IMR" are appended. "_THIS_IMR" refers to the IMR+ device serviced by the local state machine. i.e., "ANY_THIS_IMR" refers to any AUI or TP port on this particular IMR+ device. "_XTHIS_IMR" refers to any IMR+ device except the one serviced by the state machine. i.e., "ALL_XTHIS_IMR" refers to all AUI and TP ports except the ones on this particular IMR+ device.

$\overline{COL}$ = 0 implies Collin(ANYXN) = SQE or Collin(ANYXM) = SQE.
$\overline{ACK}$ = 0 && DAT = 0 && JAM = 1 implies Collin(ANYXN) = SQE or Collin(ANYXM) = SQE.
$\overline{ACK}$ = 0 && DAT = 1 && JAM = 1 implies Collin(N) = SQE or Collin(ONLY1) = SQE.
$\overline{ACK}$ = 0 && JAM = 0 implies Datain(ANY) = $\overline{II}$ or Datain(N) = $\overline{II}$.

The state diagram would not completely describe the Expansion Port without the following notes.

Notes:

1. *$\overline{REQ}$ is set to 0 one cycle prior to DAT, JAM being driven. In other words, if $\overline{REQ}$ transitions from 1 to 0, DAT and JAM will not be driven until one cycle after the transition. Similarly, if a transition from multiple $\overline{REQ}$ lines being driven ($\overline{COL}$ = 0, ACK = 1) to only one $\overline{REQ}$ line being driven ($\overline{COL}$ = 1, $\overline{ACK}$ = 0) DAT and JAM will not be driven until one cycle after the transition.*
2. *When entering the RECEIVE COLLISION state from any state other than the RECEIVE COLLISION or the EXPANSION PORT RECEIVE state, the sourcing IMR+ device will guarantee $\overline{REQ}$ = 0, DAT = 1, JAM = 1 for one cycle even if Collin(ALL) = $\overline{SQE}$ or Datain(N) = II.*
3. *If Collin(ALL) = $\overline{SQE}$ before TT( ) >= 96, then $\overline{REQ}$ = 1, DAT = z, JAM = z even though Out( ) = Jam. Note 2 supersedes this.*
4. *The condition to transition from EXPANSION PORT RECEIVE to RECEIVE COLLISION state is different than from SEND PREAMBLE PATTERN, SEND TWO ONES, and SEND DATA to RECEIVE COLLISION. Collin(N) = SQE is the only way from EXPANSION PORT RECEIVE to RECEIVE COLLISION state. This is done by sensing DAT = 1, JAM = 1, hence this is the reason the condition stated in Note 2 is necessary.*
5. *The condition to transition from EXPANSION PORT RECEIVE to WAIT state is different than from SEND DATA to WAIT. TT(ALLXN) >= 96 is not needed to transition from EXPANSION PORT RECEIVE to WAIT state.*
6. *The WAIT state, Tw1Done, and Tw2Done are implemented in the 10BASE-T transceivers and not in the repeater state machine.*

3.10 RESPONSE TO PREAMBLE ONLY

A 96-bit preamble only packet will be signaled as a receive collision on the expansion port at the tail end of IMR/IMR+ device transmission because the packet has no start of frame delimiter (SFD).

3.11 RESPONSE TO IPG SHRINKAGE

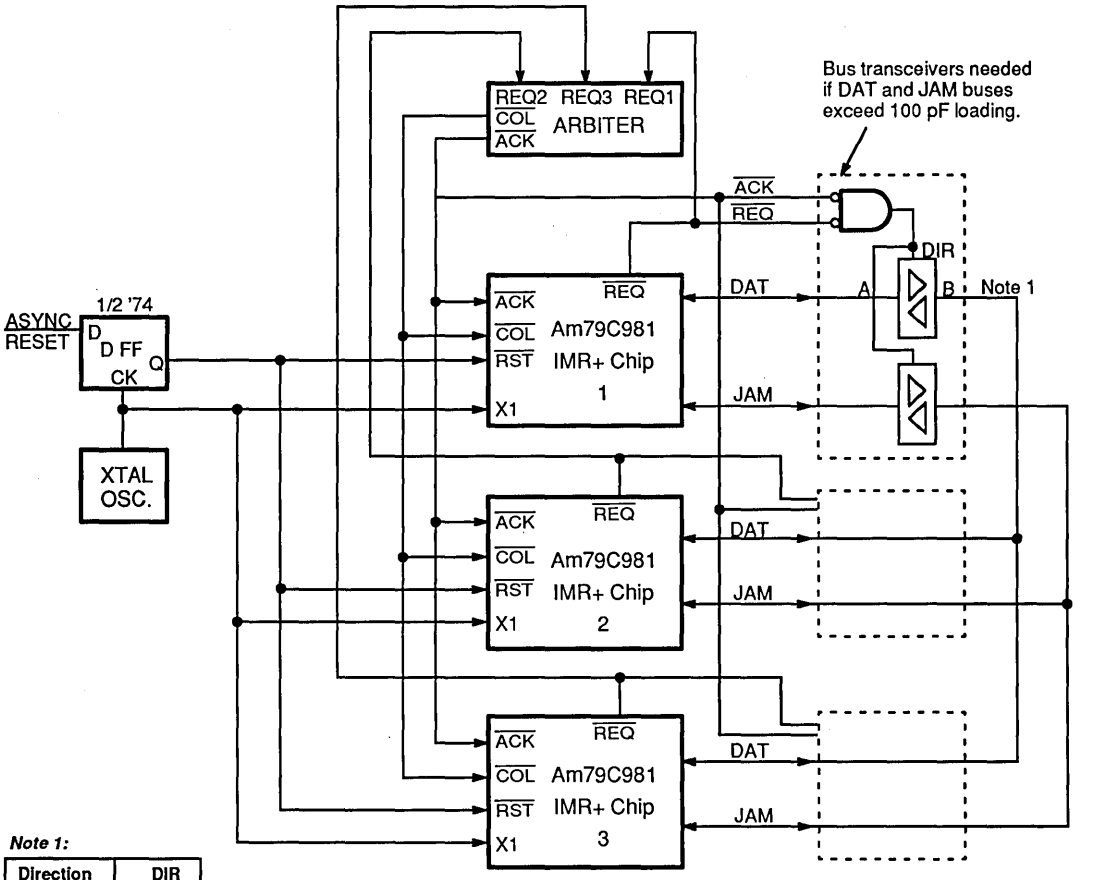
A packet reception which commences in less than the normal 9.6 µs inter-packet gap (IPG) period (due to IPG shrinkage, or the effect of a carrier drop out at an intermediate point in the transmission path), will still be treated as a new received packet by the IMR/IMR+ device. However, the PLL in the IMR/IMR+ chip requires 43-bit times of Idle

(maximum) after the end of reception to re-acquire the home frequency and to guarantee accurate decoding of received data on a subsequent packet. Thus the second packet in this sequence may be garbled by the IMR/IMR+ device if this IPG spacing is not maintained. Normal network components and operation will not cause IPG shrinkage to below 48-bits, and will therefore not be any problem for either the IMR or IMR+ devices.

3.12 DESIGNING REPEATERS USING MULTIPLE IMR+ DEVICES

Multiple IMR+ devices may be connected together to form large repeaters either with or without the HIMIB device. See Section 5 for a discussion of multiple IMR+ repeater design using the IMR+ and HIMIB devices. Figure 3-9 below shows a multiple IMR+ chip based design.

Figure 3-9 Multiple IMR+ Based Repeater



Note 1:

Direction	DIR
B → A	LOW
A → B	HIGH

17314A-14

3.13 EXPANSION PORT

The IMR+ chip Expansion Port is comprised of five pins; two are bi-directional signals (DAT and JAM), two are input signals ($\overline{ACK}$ and $\overline{COL}$), and one is an output signal ($\overline{REQ}$). These signals are used when a multiple-IMR+ device repeater application is employed. In this configuration, all IMR+ chips must be clocked synchronously with a common clock connected to the X1 inputs of all IMR+ devices. Reset needs to be synchronized to the X1 clock.

The IMR+ device expansion scheme allows the use of multiple IMR+ chips in a single board repeater or a modular multiport repeater with a backplane architecture. The DAT pin is a bidirectional I/O pin which can be used to transfer data between the IMR+ devices in a multiple-IMR+ chip design. The data sent over the DAT line is in NRZ format and is synchronized to the common clock. The JAM pin is another bidirectional I/O pin that is used by the active IMR+ chip to communicate its internal status to the remaining (inactive) IMR+ devices. When JAM is asserted HIGH, it indicates that the active IMR+ device has detected a collision condition and is generating Jam Sequence. During this time when JAM is asserted HIGH, the DAT line is used to indicate whether the active IMR+ chip is detecting collision on one port only or on more than one port. When DAT is driven HIGH by the IMR+ chip (while JAM is asserted by the IMR+ chip), then the active IMR+ device is detecting a collision condition on one port only. This 'one-port-left' signaling is necessary for a multiple-IMR+ device repeater to function correctly as a single multiport repeater unit. The IMR+ chip also signals the 'one port left' collision condition in the event of a runt packet or collision fragment; this signal will continue for one expansion port bus cycle (100 ns) before deasserting $\overline{REQ}$.

The arbitration for access to the bussed bi-directional signals (DAT and JAM) is provided by one output ($\overline{REQ}$) and two inputs ($\overline{ACK}$ and $\overline{COL}$). The IMR+ chip asserts the $\overline{REQ}$ pin to indicate that it is active and wishes to drive the DAT and JAM pins. An external arbiter senses the $\overline{REQ}$ lines from all the IMR+ devices and asserts the $\overline{ACK}$ line when one and only one IMR+ chip is asserting its $\overline{REQ}$ line. If more than one IMR+ chip is asserting its $\overline{REQ}$ line, the arbiter must assert the $\overline{COL}$ signal, indicating that more than one IMR+ device is active. More than one active IMR+ device at a time constitutes a collision condition, and all IMR+ devices are notified of this occurence via the $\overline{COL}$ line of the Expansion Port.

Note that a transition from multiple IMR+ devices arbitrating for the DAT and JAM pins (with $\overline{COL}$ asserted, $\overline{ACK}$ deasserted) to a condition when only one IMR+ chip is arbitrating for the DAT and JAM pins (with $\overline{ACK}$ asserted, $\overline{COL}$ deasserted) involves one expansion port bus cycle (100 ns). During this transitional bus cycle, $\overline{COL}$ is deasserted, $\overline{ACK}$ is asserted, and the DAT and JAM pins are not driven. However, each IMR+ device will remain in the collision state (transmitting jam sequence) during this transitional bus cycle. In subsequent expansion port bus cycles ($\overline{REQ}$ and $\overline{ACK}$ still asserted), the IMR+ devices will return to the 'master and slaves' condition where only one IMR+ device is active (with collision) and is driving the DAT and JAM pins. An understanding of this sequence is crucial if non-IMR+ devices (such as an Ethernet controller) are connected to the expansion bus. Specifically, the last device to back off of the Expansion Port after a multi-IMR+ chip collision must assert the JAM line until it too drops its request for the Expansion Port.

3.14 EXTERNAL ARBITER

A simple arbitration scheme is required when multiple IMR+ devices are connected together to increase the total number of repeater ports. The arbiter should have one input ($\overline{REQ1}$...$\overline{REQn}$) for each of the n IMR+ devices to be used, and two global outputs ($\overline{COL}$ and $\overline{ACK}$). This function is easily implemented in a PAL® device, with the following logic equations:

$$\begin{aligned}
ACK &= \overline{REQ1} \,\&\, \overline{REQ2} \,\&\, \overline{REQ3} \,\&\, ...\overline{REQn} \\
&+ \overline{REQ1} \,\&\, REQ2 \,\&\, \overline{REQ3} \,\&\, ...\overline{REQn} \\
&\quad \bullet \\
&\quad \bullet \\
&\quad \bullet \\
&+ \overline{REQ1} \,\&\, \overline{REQ2} \,\&\, \overline{REQ3} \,\&\, ... \, REQn \\
COL &= \overline{ACK} \,\&\, (REQ1 + REQ2 + REQ3 + ... \, REQn)
\end{aligned}$$

Above equations are in positive logic, i.e., a variable is true when asserted.

A single PALCE16V8 will perform the arbitration function for a repeater based on several IMR+ devices.

3.15 RESET CIRCUITRY

If the IMR+ device is used without the HIMIB device, is not connected to any other devices via its expansion bus, and the data presented on the SO and CRS pins is not used, then the $\overline{RST}$ signal can be asynchronous to the 20 MHz clock.

In many cases however the $\overline{RST}$ signal must be synchronized to this clock. Figure 3.4 shows one example where synchronization is necessary, and one possible circuit for accomplishing this function. The asynch reset signal is active low, and may be supplied by a pushbutton, software, or a conventional resistor-capacitor-diode power up reset circuit.

The $\overline{RST}$ signal generated by this circuit should be applied to every IMR+ device and associated logic in the system.

There is an alternative scheme for synchronizing $\overline{RST}$ with TCK which is useful in systems where different IMR+ devices may be powered up or otherwise reset at different times (see Figure 3-10). This scheme is used in the design of the "IMR Velcro Hub Board".

The reset curcuitry consists of a push button switch, an RC network, a Schmitt-trigger inverter, and a D flip-flop (Figure 3-10). The large RC time constant value used [t = (10 KΩ) (47 μF) = 470 ms) ensures that the reset pulse with is greater than the required minimum value of t_{RST} = 150 ms. However, this also means that the rise and fall times of the signal will be longer than desired. For this reason, the Schmitt-trigger inverter is used to buffer, and "square up" the asynchronous reset signal from the RC network to the input of the synchronizing D flip-flop. This synchronizing D flip-flop is clocked by RESET_CLK, and the Q output is used so that $\overline{RST}$ is a synchronous, active low, signal.

$\overline{RST}$ is a synchronous signal for two reasons. The first is to prevent any possible problems that might arise from an asnchronous reset signal (i.e., IMR/IMR+ device and peripheral chips recognizing reset at different times). The second, and more important reason, is that a reset signal synchronized to TCK (RESET_CLK) forces all the IMR/IMR+ devices connected to the expansion bus to be synchronized to each other after a reset cycle.

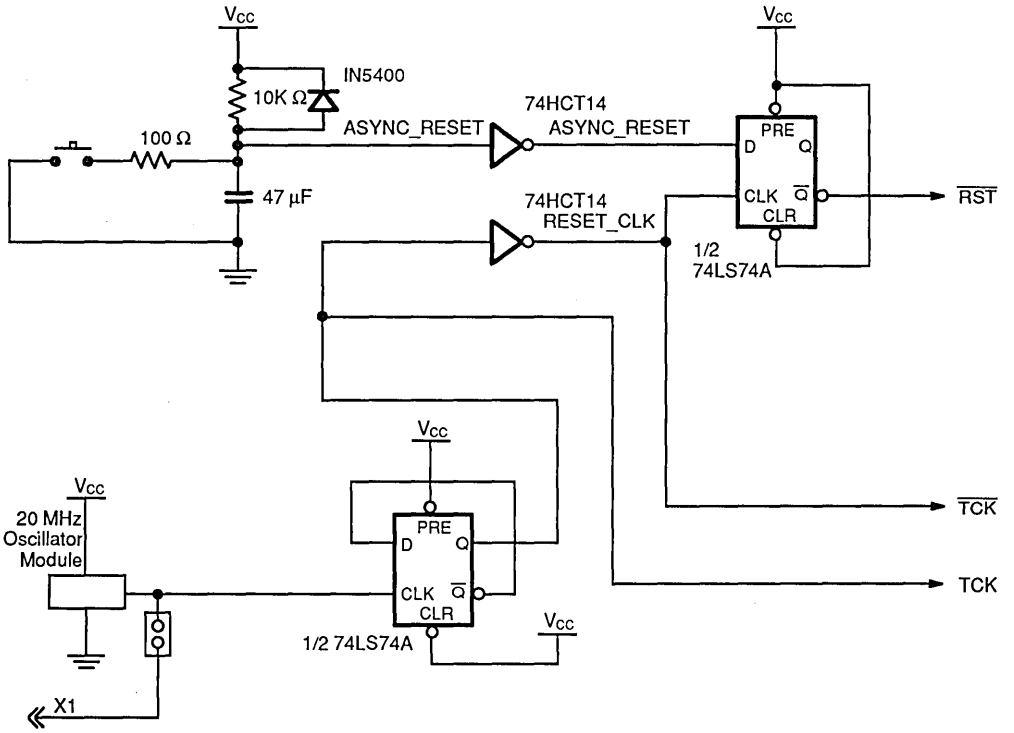
The implementation of Figure 3-4 recommends that, for generating a TCK signal of the same phase as the IMR device's internal 10 MHz TCK clock signal, the frequency divider D flip-flop's CLR input should be connected to a reset signal which is synchronized to the X1 clock. This design uses, the TCK signal for synchronization.

The reason for this is that, in Figure 3-4 the intent is to generate a TCK signal which, after a synchronized reset cycle, matches the IMR device's internal 10 MHz signal. In contrast, in Figure 3-10, the designs TCK signal is not affected by reset and is free running. Synchronizing RESET to TCK forces the IMR chip's internal 10 MHz signal to match the board's TCK signal signal after a reset cycle. Because of the propagation delay of the D flip-flop (t_{PD}), the end of the reset cycle is not recognized by the IMR+ chip until one X1 clock cycle after the rising edge of TCK (RESET_CLK). The recognition of the end of reset corresponds with the rising edge of TCK, and reesults in the same phase relationship between the IMR+ device's internal TCK and the board's TCK, as is the case when the circuit of Figure 3-4 is used.

The large RC time constant (see above) of the RC network used for the asynchronous reset signal ensures that the IMR+ chip will be reset upon powert up. Therefore, the IMR+ device will be synchronized to the board's TCK signal at all times.

Note that the $\overline{RST}$ signal is a local signal and is not included on the expansion bus. When two or three units of the "IMR Velcro Hub Board" are connected together over the expansion bus, each board has it's own reset signal synchronized by the inverse of the bus' 10 MHz TCK clock signal. In this way, the expansion bus scheme of the "IMR Velcro Hub Board" demonstrates one way the "hot swapping" of line cards in an expandable repeater can be accomplished.

Figure 3-10 Reset Circuitry



3.16 DIFFERENCES BETWEEN THE IMR AND IMR+ DEVICES

The following sections list the functions that have been added to the IMR+ device or have been modified from the original IMR device. These enhancements address a number of issues, including interfacing to the HIMIB chip, improved timing specifications, and enhanced standards support.

For a description of the Management Port commands available on the IMR and IMR+ devices, see in this section under "IMR/IMR+ Management Port".

3.16.1 IMR+ Chip Programmable Options

This command is not available in the IMR device. Three additional programming bits are provided in the IMR+ device, as follows:

S AUI Port SQE Test Mask. Mask SQE activity on the AUI port from being repeated to the 10BASE-T ports and the Expansion Port.

A Alternative Port Activity Monitor (PAM) Function. Provides raw carrier sense activity on the PAM, regardless of the enable/disable state or the auto-partition state of the AUI and/or 10BASE-T ports.

C CI reporting. Provides reporting of CI circuit activity within the PAM bit stream. See the IMR+ device Data Sheet (PID# 17306) for additional details.

See in this section the "Management Commands, SET (Write) Opcodes" under "IMR/IMR+ Management Port" for additional details.

3.16.2 Get AUI Port Status

The original Get AUI Port Status command available on the IMR device, is enhanced in the IMR+ chip to provide three additional status bits. Hence a single request can be used to monitor all key performance aspects of the AUI port. This command uses the same op-code (10001111) as the original Get AUI Port Status command of the IMR device. The following status is returned from the IMR+ device (see Table 3-1 for more detail):

P Partitioning Status. Reports the auto-partition condition of the AUI port. Identical in both the IMR and IMR+ devices.

B Bit Rate Error. Reports a FIFO underrun/overrun due to a frame received on the AUI port. This bit is not available in the IMR device, and is added to the IMR+ device.

S SQE Test Status. Reports that the transceiver connected to the AUI port has SQE Test enabled. This bit is not available in the IMR device, and is added to the IMR+ device.

L Loop Back Error. Reports that the AUI port is not exhibiting a DO circuit to DI circuit loopback path. This bit is not available in the IMR device, and is added to the IMR+ device.

See the IMR+ device Data Sheet (PID# 17306) for additional details.

In addition, three further versions (four versions in total) of this command are provided, which allow various combinations of bits to be cleared in the process of reading the status.

3.16.3 Get Bit Rate Error (TP Ports)

This command is not available in the IMR device, and is added to the IMR+ device. Reports a FIFO underrun/overrun due to a frame received on any of the 10BASE-T ports.

3.16.4 Get MJLP Status

This command is not available in the IMR device, and is added to the IMR+ device. Reports that the MAU Jabber Lockup Protection (MJLP) timer has activated.

3.16.5 Get Version

The Get Version command is used to determine the device version number.

The IMR device response to this command is:	XXXX 0000
The IMR+ device response to this command is:	XXXX 0001

3.16.6 Minimum Mode

This function is not available in the IMR device, and is added to the IMR+ device. See the "IMR+ Based Velcro™ Hub Design" and "Minimum Mode" sections for additional details.

3.16.7 FIFO Underflow/Overflow

In the IMR device, a FIFO over-run or under-run condition forces the device to enter the Transmit Collision State, causing transmission of the Jam pattern to all ports, including the receiving port.

In the case of the IMR+ device, the FIFO underflow/overflow error is handled using the Receive Collision State. In this case, the responsible port will not observe a Jam pattern returned from the IMR+ based repeater (a Bit Rate Error will be reported for the port). Since this error is caused by a serious receive error condition (i.e., the station is transmitting at a frequency substantially outside the allowable specification, or the transmission is excessively long), it is treated as a Receive Collision.

3.16.8 Twisted Pair Link Integrity Status

3.16.8.1 Port Disable and Link Test

On the IMR device, when a Twisted Pair port is disabled, its corresponding status is reported as "Link Pass", regardless of the condition of the Link Test state machine.

On the IMR+ device, a disabled port continues to report accurate Link Test Status.

Disabling an enabled Twisted Pair port on the IMR device, causes the port to be forced into the Link Fail state.

Re-enabling a disabled port on the IMR+ device, causes the port to be forced into the Link Fail state. This ensures that packet fragments received on the port are not repeated to the rest of the network.

3.16.8.2 Interaction Between Link Fail and Port Autopartition.

In the IMR device, a Twisted Pair port which is partitioned and subsequently enters the Link Fail state, will be reconnected.

In the IMR+ device, the Twisted Pair port Receive Link Test State machine is decoupled from the Autopartition State Machine. Entering the Link Fail state will not cause a partitioned port to be reconnected.

3.16.9 Preamble/Start of Frame Delimiter (SFD) Detection

Once receive preamble is detected, the Repeater State machine of the IMR device will commence the re-transmission of preamble on all permitted output ports, and begin searching for a "two ones" condition in the preamble. Once the "1, 1" Synch character has been detected, the packet data which follows is loaded into the internal FIFO, to be re-transmitted after the preamble has been completed at the output ports. If the IMR device does not detect an SFD, then it will remain in the preamble generation state until the receive carrier has dropped or a collision is detected.

In the case of the IMR+ device, after receive preamble is detected, the Repeater State machine will commence the re-transmission of preamble on all permitted output ports, but will ignore an SFD (Start of Frame Delimiter) which arrives with less than 15 bits of preamble. The IMR+ device will not monitor the input bit stream to search for the SFD until it has received 16 bits (1.6 µs) of preamble. The IMR+ device will decode the SFD

fully, and only respond to the valid SFD pattern (10101011). Once the valid SFD byte has been detected, the packet data which follows is loaded into the internal FIFO, to be re-transmitted after the preamble has been completed at the output ports. If the IMR+ device does not detect an SFD, then it will remain in the preamble generation state until the receive carrier has dropped or a collision is detected.

3.16.10 Hardware Reset

The IMR chip has a minimum reset pulse requirement of 150 µs. This requirement allows for the receive clock recovery circuitry to stabilize. The IMR+ device has the same requirement for a minimum reset pulse width at power on, however if the reset is applied while power stays active the pulse width requirement to reset the IMR+ chip decreases to 4 µs. Note that both the IMR and IMR+ devices remain in the reset state for 10 clock cycles (0.5 µs) following the end of the reset pulse.

During reset, receive activity on all Twisted Pair ports is ignored by the repeater if activity was started while the IMR/IMR+ device is in its internal RESET state. This ensures that packet fragments are not propagated onto the network due to a Management Software Reset (which will occur as a result of a Management Station action).

The reset circuitry in the IMR+ device ensures that the chip is internally reset only once, and for sufficient duration. This avoids multiple resets within the device, in the case where a slow rising edge reset signal, crosses the input buffer threshold several times due to ripple on the input waveform.

3.17 IMR/IMR+ PROPAGATION DELAYS

Since the IMR+ device is a general purpose repeater chip, it does not directly interface with a backplane bus from a specific vendor. Since there are numerous existing concentrator backplanes that are currently available in the market place, making a part which interfaces to a variety of these is a non trivial problem.

If the IMR+ device is to be utilized in a new design, where no previous backwards compatibility is required, then the dedicated Expansion Port can be used as the basis of the backplane, to interconnect individual IMR+ chip or groups of IMR+ devices, such that a modular concentrator architecture can be accomplished. This concept is discussed in the existing data sheet.

In order to provide the maximum flexibility to the installed base, where backwards compatibility with an existing architecture is required, it is necessary to describe the external behavior of the IMR+ device, in the form of throughput delays, so that designers can understand the issues and ensure that the critical repeater delays specified in Section 9 of 802.3 can be met.

Below are listed the IMR+ propagation delays for key parameters. The values are based on a combination of related empirical data, simulation, and bench/tester measurements. In general, maximum values have assumed theoretical worst case conditions. Propagation delays cited below, which do not directly match or correspond to data sheet parameters or IEEE 802.3 requirements, are not guaranteed by production testing.

In addition, data out delays pertaining to digital pins are considered to be zero in order to avoid double counting of such delays. For paths including a digital output buffer and an input buffer, the data out delays have been accounted for in the input buffer parameter in the form of a range of input setup times.

3.17.1 Start of Packet Propagation Delays

Note that for start of packet propagation delays, the input signal is assumed to be a valid Manchester encoded signal, with first bit of valid amplitude and pulse width. The input (stimulus) waveform will have a direct effect on the propagation delay. For instance, using a 5 MHz non-Manchester input into the TP ports, and measuring the propagation

delay to the TP output, may yield a delay value 50 ns larger than expected, due to the first bit not being valid Manchester.

Parameter	Min (ns)	Max (ns)
Data on AUI to $\overline{REQ}$ asserted	100	280
Data on AUI to data out on any TP port	190	370
Data on AUI to data out on DAT	200	380
Data on any TP port to $\overline{REQ}$ asserted	250	430
Data on any TP port to data out on AUI	340	520
Data on any TP port to data out on any TP port	340	520
Data on any TP port to data out on DAT	350	530
Data on DAT/$\overline{ACK}$ to data out on AUI port	90	190
Data on DAT/$\overline{ACK}$ to data out on any TP port	90	190
$\overline{ACK}$ asserted to data out on DAT	0	100

3.17.2 Start of Collision Propagation Delays

Parameter	Min (ns)	Max (ns)
AUI CI active to Jam sequence on AUI	290	470
AUI CI active to Jam sequence on any TP port	290	470
AUI CI active to Jam sequence on JAM	300	480
AUI CI active to $\overline{REQ}$ asserted	200	380
TP RX active to Jam sequence on AUI	340	520
TP RX active to Jam sequence on any TP port	340	520
TP RX active to Jam sequence on JAM	350	530
JAM/$\overline{ACK}$ asserted to Jam sequence on AUI	90	190
JAM/$\overline{ACK}$ asserted to Jam sequence on any TP port	90	190
$\overline{COL}$ asserted to Jam sequence on AUI	90	190
$\overline{COL}$ asserted to Jam sequence on any TP port	90	190

Note:

Propagation delays cited in these tables are not guaranteed by production testing.

3.17.3 End of Collision Propagation Delays

Parameter	Min (ns)	Max (ns)
AUI CI carrier sense deasserted to end of Jam sequence on AUI port	140	320
AUI CI/DI carrier sense deasserted to end of Jam sequence on any TP port	190	370
AUI CI/DI carrier sense deasserted to end of Jam sequence on JAM	200	380
AUI CI/DI carrier sense deasserted to $\overline{REQ}$ deasserted	200	380
TP carrier sense deasserted to end of Jam sequence on AUI port	140	320
TP carrier sense deasserted to end of Jam sequence on any other TP port	190	370
TP carrier sense deasserted to end of Jam sequence on JAM	200	380
TP carrier sense deasserted to $\overline{REQ}$ deasserted	200	380
JAM/$\overline{ACK}$ deasserted to end of Jam sequence on AUI port	40	140
JAM/$\overline{ACK}$ deasserted to end of Jam sequence on any TP port	90	190
$\overline{COL}$ deasserted to end of Jam sequence on AUI port	40	140
$\overline{COL}$ deasserted to end of Jam sequence on any TP port	90	190
Internal AUI/TP carrier sense deassertion	136	200

3.17.4 TP Start-up and Steady State Delays

TP start-up delay, from the first edge of a received waveform to the first edge of a transmitted waveform, varies from 400 ns to 550 ns. The path includes propagation delays, smart squelch, synchronization to the transmit clock (X1) and repeater state machine processing time with the variation introduced by propagation delays and synchronization time.

TP steady state delay is a function of internal propagation/timing delays as well as the extent to which data is buffered in the internal FIFO. The depth of FIFO usage required depends on the amount of preamble (including SFD) received as well as the magnitude of frequency mismatch between the extracted receive clock and the transmit clock. The FIFO will be nearly empty during transfers involving a received preamble of >64 bits coupled with a fast transmit clock (X1), whereas the FIFO will be nearly full during transfers involving a very short received preamble of <36 bits coupled with a slow transmit clock (X1). In total, the steady state delay can vary from 6 to 37 bit times.

3.17.5 TP Receive to DAT Active Delay

This parameter is identical to the TP start-up delay (see above).

3.17.6 Slow $\overline{ACK}$ or $\overline{COL}$ Signals

A late $\overline{ACK}$ assertion from an external expansion port bus arbiter does not prevent the expansion bus source IMR/IMR+ device from engaging in its preamble sequence. It will drive preamble on its TP and AUI ports while its expansion port remains in the high-impedance state. Expansion port destination IMR/IMR+ devices will be unable to repeat the preamble bits which normally would have been passed from the expansion bus source during the time in which $\overline{ACK}$ is tardy. For a $\overline{REQ}$ asserted to $\overline{ACK}$ asserted

delay which is tardy by up to 100 ns (fails the t_{CASET} parameter up to 100 ns), destination IMR/IMR+ devices will issue a preamble sequence which is shortened by one bit versus the source IMR/IMR+ device, and that preamble sequence will begin with a Manchester zero.

A late $\overline{ACK}$ deassertion from an external expansion port bus arbiter will induce destination IMR/IMR+ devices to repeat expansion port data even though the expansion port bus is not being driven with data by the source IMR/IMR+ device. Again an $\overline{ACK}$ destination deassertion delay of 100 ns will result in the transmission of one errant bit by all destination IMR/IMR+ devices, resulting in an undefined "dribbling bit" being added to the re-transmitted data. For an $\overline{ACK}$ deassertion which is more than 100 ns tardy (fails the $\overline{ACK}$ t_{CASET} by more than 100 ns), the original source IMR/IMR+ device will perceive itself as an expansion port destination device ($\overline{ACK}$ asserted, $\overline{REQ}$ deasserted) and will issue a minimum 96-bit jam sequence due to receiving a runt packet.

Supplying $\overline{ACK}$ immediately, followed by a tardy $\overline{COL}$ indication can result in expansion port bus contention if two IMR/IMR+ devices simultaneously request the bus.

SECTION

4 HIMIB OVERVIEW

The Hardware Implemented Management Information Base (HIMIB) device provides complete hardware support for the high performance network monitoring requirements of the IEEE 802.3 Repeater Management Standard. The HIMIB chip was designed to be used with the Integrated Multiport Repeater (IMR+), providing a flexible and complete solution for managed repeater design. It can also be interfaced with any Ethernet controller supporting a General Purpose Serial Interface (GPSI).

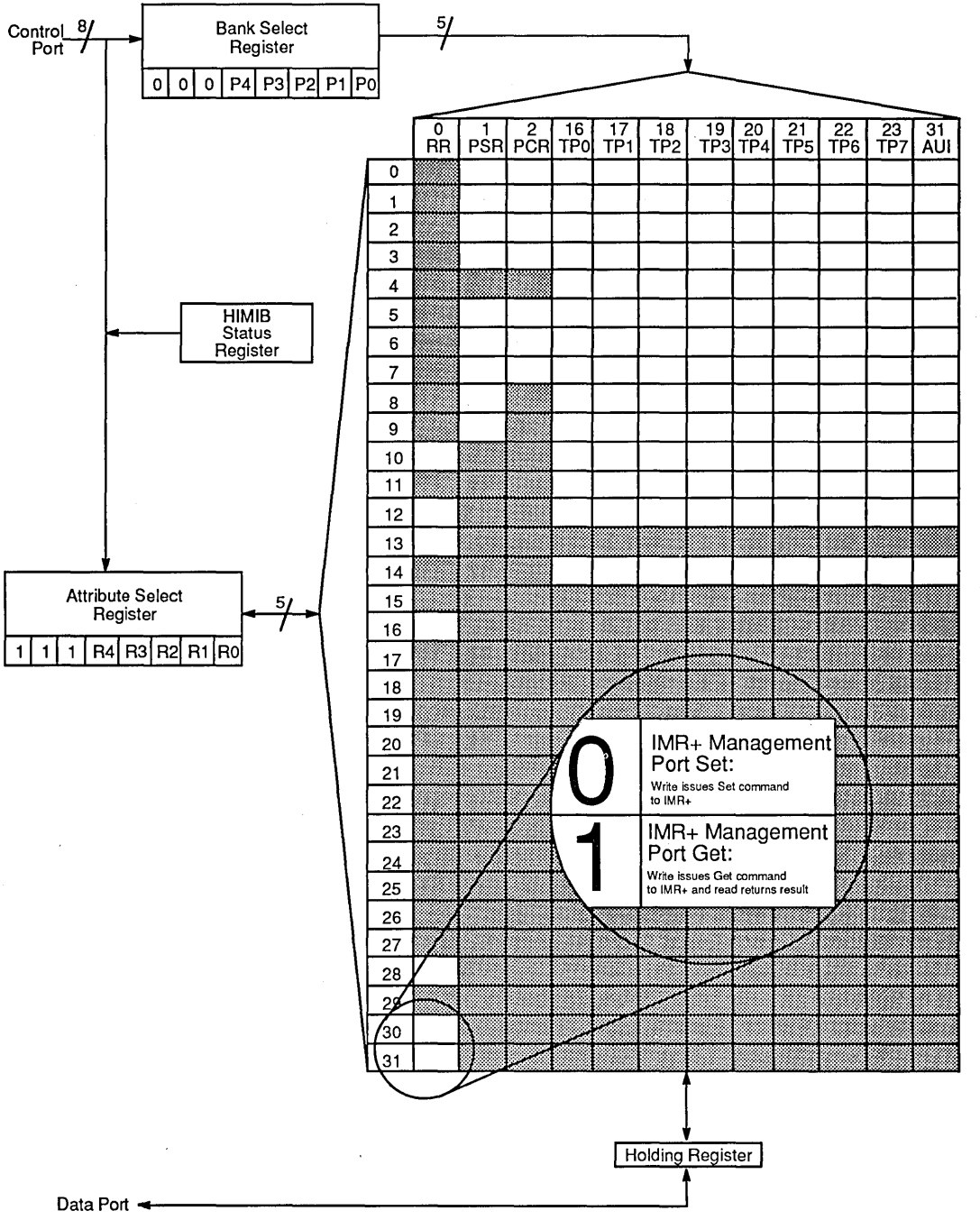
The IEEE 802.3 repeater management standard defines three management packages, for Basic Control, Performance Monitoring, and Address Tracking. Both Performance Monitoring and Address Tracking require real time statistics collection for each repeater port on a per-frame basis. The HIMIB chip, in conjunction with the IMR+ device, provides a complete solution for all required and optional statistics. All network activity related statistics will be updated autonomously by the HIMIB device with no host processor involvement. The HIMIB device connects directly to the IMR+ Expansion Bus, Management Port and Port Activity Monitor (PAM), enabling the HIMIB chip to directly track and count critical network events for local and remote management queries. Coupled with a processor and simple IEEE 802.3 MAC, the system can be expanded to support complete remote in-band network management capabilities. Additionally the HIMIB chip supports the Novell, Inc. Hub Management Interface (HMI) extensions to the IEEE 802.3 Repeater Management requirements.

The HIMIB chip provides a simple 8-bit asynchronous bus interface, allowing interfacing of a low cost 8-bit microprocessor, to provide an extremely low cost solution for managed repeater designs.

4.1 ARCHITECTURAL OVERVIEW

The HIMIB device autonomously carries out all statistics accumulation for managed repeater applications. These statistics are accumulated in an array of registers as shown in Figure 4-1. The register array is addressed as 32 banks with 32 registers in each bank. The HIMIB device currently only uses 12 banks, with a varying number of registers in each bank. Each register is addressed using a Port Number (Bank Number) and Register Number (Attribute Number) pointer combination. The lower 5 bits of both of these registers effectively address a single attribute register in the HIMIB chip's register array.

Figure 4-1 HIMIB Device Register Array



4.2 HIMIB/IMR+ CHIP-SET MANAGEMENT CAPABILITIES

Table 4-1A and 4-1B is a listing of the HIMIB device registers. In Table 4-1A, the columns are the register banks, while the rows are individual registers in the banks.

Table 4-1A HIMIB Registers

Reg #	Register Bank 0 Repeater Registers	Register Bank 1 Port Status Registers	Register Bank 2 Port Control Registers	Register Banks 16-23 10BASE-T Ports	Register Bank 31 AUI Port
0		TP Partition Status Change	TP Partition Change Interrupt Enable	Readable Frames	Readable Frames
1		AUI Partition Status Change	AUI Partition Change Interrupt Enable	Readable Octets	Readable Octets
2		TP Link Status Change	TP Link Status Change Interrupt Enable	Frame Check Sequence Errors	Frame Check Sequence Errors
3		AUI Loop Back Error	AUI Loop Back Error Interrupt Enable	Alignment Errors	Alignment Errors
4				Frames Too Long	Frames Too Long
5		AUI SQE Test Error	AUI SQE Test Error Interrupt Enable	Short Events	Short Events
6		TP Source Address Change	TP Source Address Change Interrupt Enable	Runts	Runts
7		AUI Source Address Change	AUI Source Address Change Interrupt Enable	Collisions	Collisions
8		TP Source Address Match Status		Late Events	Late Events
9		AUI Source Address Match Status		Very Long Events	Very Long Events
10	Source Address Match (6 Bytes)			Data Rate Mismatches	Data Rate Mismatches
11				Auto Partitions	Auto Partitions
12	Total Octets			Source Address Changes	Source Address Changes
13	Transmit Collisions				
14				Last Source Address (6 Bytes)	Last Source Address (6 Bytes)
16	Configuration Register				
28	Version/Device ID				
30	IMR+ Management Port Set Register				
31	IMR+ Management Port Get Register				

Table 4-1B HIMIB Registers

Register				Bytes	Access
Status Register		Note: Read the C Port for Status No Need to Specify the Port or Register Number		1	R
Port/Register Bank	**P[4:0]**	**Register**	**R[4:0]**	**Bytes**	**Access**
Repeater Registers	0	Source Address Match	10	6	R/W
		Total Octets	12	4	R
		Transmit Collisions	13	4	R
		Configuration Register	16	1	R/W
		Version/Device ID	28	1	R
		IMR+ Management Port Set Register	30	1	W
		IMR+ Management Port Get Register	31	1	R/W
Port Status Registers	1	TP Partition Status Change	0	1	R
		AUI Partition Status Change	1	1	R
		TP Link Status Change	2	1	R
		AUI Loop Back Error	3	1	R
		Reserved	4		
		AUI SQE Test Error	5	1	R
		TP Source Address Change	6	1	R
		AUI Source Address Change	7	1	R
		TP Source Address Match Status	8	1	R
		AUI Source Address Match Status	9	1	R
Port Control Registers	2	TP Partition Change Interrupt Enable	0	1	R/W
		AUI Partition Change Interrupt Enable	1	1	R/W
		TP Link Status Change Interrupt Enable	2	1	R/W
		AUI Loop Back Error Interrupt Enable	3	1	R/W
		Reserved	4		
		AUI SQE Test Error Interrupt Enable	5	1	R/W
		TP Source Address Change Interrupt Enable	6	1	R/W
		AUI Source Address Change Interrupt Enable	7	1	R/W
Attribute Registers	16–23, 31	Readable Frames	0	4	R
		Readable Octets	1	4	R
		Frame Check Sequence Errors	2	4	R
		Alignment Errors	3	4	R
		Frames Too Long	4	4	R
		Short Events	5	4	R
		Runts	6	4	R
		Collisions	7	4	R
		Late Events	8	4	R
		Very Long Events	9	4	R
		Data Rate Mismatches	10	4	R
		Auto Partitions	11	4	R
		Source Address Changes	12	4	R
		Reserved	13		
		Last Source Address	14	6	R/W

Note that all register locations listed as reserved and those which might be accessed by values or combinations of P and R which are not listed in the table above should not be accessed by the software. Read/write access to reserved registers may cause incorrect operation.

The following tables list the implementation requirements for management objects in a repeater based on the IMR+/HIMIB devices. They are organized into three tables. The first describes the management attributes that are supported in software. The second describes the management attributes supported directly by the HIMIB device. The third lists the management attributes supported by the IMR+ device.

Table 4-2 is a list of the management objects that would typically be implemented in a managed repeaters resident software. These objects are generally static, or change very infrequently, and hence present minimal overhead to the managed repeater software.

Table 4-2 Software Management Objects

Management Object Name	Management Class	Object Type
repeaterID	Repeater	Read Only Attribute
repeaterGroupCapacity	Repeater	Read Only Attribute
groupMap	Repeater	Read Only Attribute
repeaterHealthState	Repeater	Read Only Attribute
repeaterHealthText	Repeater	Read Only Attribute
repeaterHealthData	Repeater	Read Only Attribute
resetRepeater	Repeater	Action
executeNonDisruptiveSelfTest	Repeater	Action
repeaterHealth	Repeater	Notification
repeaterReset	Repeater	Notification
groupMapChange	Repeater	Notification
resourceTypeIDName	ResourceTypeID	Read Only Attribute
resourceInfo	ResourceTypeID	Read Only Attribute
groupID	Group	Read Only Attribute
groupPortCapacity	Group	Read Only Attribute
portMap	Group	Read Only Attribute
portMapChange	Group	Notification
portID	Port	Read Only Attribute
portAdminState	Port	Read Only Attribute
portAdminControl	Port	Action

Table 4-3 is a list of the management objects directly supported by the HIMIB, along with the bank and register addresses to access the HIMIB register contents. These objects are updated during any activity on any IMR+ device's network port. Hence the hardware counters provided by the HIMIB chip alleviates the managed repeater software from all time critical duties.

Table 4-3 HIMIB Management Objects

Management Object Name	HIMIB Register	Management Class	Bank Address	Register Address
transmitCollisions (note)	Transmit Collisions	Repeater	0	13
readableFrames	Readable Frames	Port	16-23, 31	0
readableOctets	Readable Octets	Port	16-23, 31	1
frameCheckSequenceErrors	FCS Errors	Port	16-23, 31	2
alignmentErrors	Alignment Errors	Port	16-23, 31	3
framesTooLong	Frames Too Long	Port	16-23, 31	4
shortEvents	Short Events	Port	16-23, 31	5
runts	Runts	Port	16-23, 31	6
collisions (note)	Collisions	Port	16-23, 31	7
lateEvents	Late Events	Port	16-23, 31	8
veryLongEvents	Very Long Events	Port	16-23, 31	9
dataRateMismatches (note)	Data Rate Mismatches	Port	16-23, 31	10
autoPartitions	Auto Partitions	Port	16-23, 31	11
lastSourceAddress	Last Source Address	Port	16-23, 31	14
sourceAddressChanges	Source Address Changes	Port	16-23, 31	12
RepeaterVeryLongEvents (HMI)	Frames Too Long (Sum all ports together)	Repeater	0	31
RepeaterTotalOctets (HMI)	Total Octets	Repeater	0	12

Note: *These attributes are only available to IEEE specified accuracy when used with the IMR+ device. They are not supported, or inaccurately reported, when the IMR (Am79C980) device is used.*

Note that using the Get Bit Rate Error Status of TP Ports command through the HIMIB chip's IMR+ Management Port Get Register will destroy the status of any set bits, and may cause the HIMIB to miss a DataRateMismatches attribute count. Although access to the raw bit rate status is provided, it is recommended that this is only be monitored through the HIMIB's DataRateMismatches attribute, and not by directly reading the IMR+ chip's internal status.

Table 4-4 lists the IEEE 802.3 management objects directly supported by the IMR+. These attributes are accessed by performing get operations through the IMR+ Management Port Get Register. Since these objects are accessed infrequently, they present minimal overhead to the managed repeater software.

Table 4-4 IMR+ Management Objects

Management Object Name	IMR+ Command	Management Class	Bank Address	Register Address	IMR+ Code
autoPartitionState	TP Port Partitioning Status	Port	0	31	1000 0000
autoPartitionState	AUI Port Status	Port	0	31	1000 1001 (note)
PortLinkState (HMI)	Link Test Status of TP Ports	Port	0	31	1101 0000

Note: *An access to Get AUI Port Status using any of the four options provided, will not be permitted to destroy the status of bits which the HIMIB chip requires during the update of port attributes. Hence, when any of the Get AUI Port Status commands is input through the HIMIB chip's IMR+ Management Port Get Register, the HIMIB chip will convert the command op-code prior to sending it serially over the IMR+ Management Port to the version which does not cause the clearing of any status bits (op-code 1000 1001).*

4.3 HIMIB DEVICE HARDWARE DESIGN CONSIDERATIONS

4.3.1 HIMIB Device Reset

The IMR+ and HIMIB devices are designed to be driven from a common reset signal. The reset signal has the following characteristics. The reset signal must be synchronized to the IMR+/HIMIB clock source. Figure 3-10 shows a typical reset synchronization circuit. The IMR+ chip has a power on reset requirement of 150 µs. Both the IMR+ and HIMIB devices have a soft (non power-on) reset requirement of 4 µs.

4.3.2 RDY Requirements

The HIMIB chip's processor interface uses asynchronous handshaking. The processor issues a read or write request, and then waits until the HIMIB chip signals it is done with the operation. The HIMIB device requires a variable amount of time to respond to processor read or write cycles. The longest read cycle is when the processor requests that the HIMIB device "Get" an on-chip IMR+ register value. This cycle requires that the HIMIB issues a Get request using the IMR+ Management Port, following which the IMR+ chip will return the desired register contents. The state of the HIMIB chip RDY pin indicates to the processor that the transfer is complete. It is driven low at the start of the read/write cycle, and is released when the HIMIB device is ready to complete the cycle. The RDY pin should be used to drive the processor ready or acknowledge input.

4.3.3 R/W Timing on Get and Set

Most interaction between a processor and the HIMIB device are simple read and write operations on the internal register array. However the HIMIB chip also acts as a processor interface to the IMR+ device internal control and status registers. The processor modifies the IMR+ internal control registers by writing to the IMR+ chip through the HIMIB chip's IMR+ Management Port Set Register (Bank 0, Register 30). After the processor writes to this register, the HIMIB chip will serialize the contents of the write data, and transmit the contents of this register over the IMR+ Management Port. The processor is masked from this parallel-to-serial conversion process and the access appears no different than writing to any other internal HIMIB device register, although there is an internal delay before the command is actually inside the IMR+ device.

However, reading status information from the IMR+ chip is a two step operation. First the processor must write an IMR+ device Get command to the HIMIB chip's IMR+ Management Port Get Register (Bank 0, Register 31). When this happens the HIMIB device transmits the contents of this register over the IMR+ Management Port. The IMR+ chip will interpret this command, and transmit the status information back to the HIMIB device. The HIMIB chip will place the status result returned from the IMR+ chip back in the IMR+ Management Port Get Register. The processor must issue a read of the HIMIB chip's IMR+ Management Port Get Register to retrieve the results of the Get operation. As far as the processor is concerned, this is a two cycle operation. First it writes the Get command to the HIMIB chip, then it reads the results back. The results are stored in the same register as the command was, so the processor does not need to modify the register array pointer registers to read the result. However, the processor does have to wait until the status information is in the HIMIB device register, or it will be put into a wait state while the read operation is performed.

An additional consideration is that the IMR+ Set and Get commands may be delayed. Normally the IMR+ Set and Get command are executed as fast as the IMR+/HIMIB Management Port interface allows. However, the IMR+ Set and Get commands may be queued behind other IMR+ Set or Get commands. These commands can come from the processor, or they can be Get commands generated internally by the HIMIB device itself. The HIMIB chip will issue Get requests at the end of a frame in order to update its statistics. The HIMIB device performs a total of four Get commands during the inter-packet gap, two at a time. Only after the first two are performed are the second two queued. If the processor has a Get/Set command queued when the first two commands have completed, it will be serviced. Therefore a maximum of two IMR+ Get commands can be inserted in the IMR+ command queue ahead of a processor command, resulting in delay servicing the processor command.

The worst case delay under these circumstances is 70 SCLK clock cycles. The processor will be forced to wait for this duration if it attempts to read the result of a Get command from the HIMIB device before it is complete. This delay period will only affect the processor's ability to access the IMR+ chip, and in no way delays the processor when accessing other HIMIB device registers.

As described, the HIMIB device takes longer to return Get command results after a frame. Get command results can also be delayed if the processor initiates a Set command quickly followed by a Get command. The processor will be made to wait until the Set command is transmitted over the IMR+ Management Port during which time the Get command will be queued up.

4.3.4 Read/Write Recovery Time

The HIMIB chip's processor interface is designed to support fully asynchronous operation. The HIMIB device has a "recovery" time requirement of 150 ns between successive accesses. This parameter is usually provided by normal processor read and write timing cycles. However an extremely high speed processor could violate this parameter, resulting in unpredictable operation. Under these circumstances the designer has two alternatives. Either delay the processor between cycles with an additional instruction, or add delay circuitry to the processor hardware interface.

4.4 HIMIB DEVICE SOFTWARE DESIGN CONSIDERATIONS

4.4.1 HIMIB Device Register Access

Programming of the HIMIB device is performed through two ports, a Command Port (C Port) and a Data Port (D Port). The Command Port is used to set up the internal register pointers (Port/Bank select and Register/Attribute select registers), and to read the HIMIB chip's Status Register. The Data Port is used to read and write the currently selected HIMIB device register.

The HIMIB chip's data registers vary in length from 8 bits to 48 bits. Because the HIMIB device registers are accessed one byte at a time, several accesses are required to perform a complete read or write. The read or write operation is actually performed on a separate holding register. The content of the actual register is transferred to the holding register on the first cycle of a read operation, or the last cycle of a write operation.

The registers are addressed by writing two bytes to the C Port. One byte is used to select the desired Port or Bank, and the other byte selects a specific Register or Attribute in the bank. The three most significant bits in the command byte select between the Bank Select register and the Attribute select register. A value of '000' in the upper three bits indicates the Bank select register, while a value of '111' indicates the Attribute select register. Valid Bank addresses are 0, 1, 2, 16 through 23, and 31. The Bank select register and the attribute select register can be written in any order. It is possible to keep the Bank select register constant while varying the Attribute select register, or vice versa. Using this capability the programmer can access different attributes for one port, or step across the same attribute for different ports.

4.4.3 HIMIB Device Interrupt Processing

The HIMIB chip is designed to operate in either polling or interrupt driven environments. This section examines the successful use of the HIMIB device in interrupt driven environments.

The simplest software possible is a single program running alone, without interrupts. This is often called a polling environment, because the program must continually poll, or interrogate any attached devices to determine what needs to be processed. In the case of a repeater, the processor could poll the HIMIB chip to determine if anything has happened that needs attention, such as partitioning of a port.

Interrupt driven environments complicate the matter because there are actually two programs operating in the same system. The main program, (often referred to as the background program because it runs when nothing else is going on) runs while there are no interrupts. If an interrupt occurs, the processor stops executing the main program, and begins to execute the interrupt service routine (ISR), which is in effect just another program. This does not cause any problems unless the ISR modifies some resource (memory or registers) that the main program was working on.

The HIMIB device has three resources that must be protected. They are the holding register, the pointer registers (Bank select and Attribute select), and the IMR+ Management Port Get Register. If the main program is in the process of using any of those registers, the interrupt service routine must not be allowed to use them.

As an example, assume that the main program is performing a read on the Last Source Address (LSA) register (Bank 16–23, 31, Attribute 14). The pointer registers will be pointing to the LSA register, and the contents of the LSA register are in the holding register. The processor may have even read one or more of the 6 bytes from the holding register before the interrupt occurs.

Now assume that the same HIMIB device generates an interrupt. The processor stops in the middle of reading the bytes from the HIMIB chip's holding register, and begins to execute the ISR. The ISR checks, and determines that the HIMIB device needs servicing. A poorly designed interrupt service routine would then begin to use the HIMIB chip's resources that the main program was in the process of using. It would reprogram the address pointer registers to another Bank and Attribute, and read the contents of that register into the holding register. Subsequently, the ISR would read the bytes out of the holding register, and then, after changing those registers that the main program was using (Bank select, Attribute select, and the holding register), return control to the main program.

Now the main program finishes reading the bytes from the holding register, not knowing that it has been changed by the ISR, and no longer holds the contents of the LSA register. Under these circumstances the main program will unknowingly use incorrect information.

There are several approaches to this problem. The simplest approach is for the main program to disable interrupts when manipulating a resource that might be modified by an ISR. This is appropriate if the system allows it (some systems such as UNIX make disabling interrupts difficult for ordinary programs), and interrupts are disabled for a short period of time relative to the needs of the interrupting device. In this case, the processor must allow interrupts often enough to adequately service the HIMIB devices interrupt events.

A second alternative is to write the ISR such that is does not modify the resource. In this case, the ISR would not be allowed to read any of the HIMIB chip's registers, or change the pointer registers. Rather, it would determine the source of the interrupt as best it could, then set a flag in memory telling the main program to deal with the situation. The main program then must check that flag periodically, to determine when servicing is required.

Another alternative is for the ISR to restore the state of the resources after it used them, which requires that the ISR know what state the resource needs to be returned to. However, the ISR cannot read any of the required HIMIB device registers directly (such as the Bank/Register select). One solution is for the processor to store the pointer values it is using in memory where the ISR can retrieve them, and use them to restore the resource. However, even assuming that the pointer register are restored to resume the reading of an attribute (for instance), the state of the holding register cannot be restored to the original value, if the holding resistor was used during the ISR, and the attribute changed. A read from an attribute should therefore be completed before the ISR is permitted to execute, or repeated in its entirety after the ISR has completed.

It is possible to implement a mixed solution, where the ISR can manipulate some of the resources, while others are protected by having the main program disable interrupts.

4.4.4 Counter Rollover

The HIMIB chip's attribute counter registers roll over after they reach the maximum available count. Depending on the exact attribute, this can occur at most every hour to every few days. The management processor must periodically read the contents of the registers to determine if a roll-over has occurred to keep accurate statistics. The program must increment a memory based counter to maintain statistics larger than 32 bits. The following equation calculates the current count based on the value in the HIMIB chip's attribute register, the initial value in the HIMIB chip's attribute register, and the rollover count for that attribute.

$$Count = HimibVal + (RolloverCount * 2^{32})$$

Note that the 32-bit counters in the HIMIB device, upon initialization, are set to a random value. Thus the counter may overflow from a maximum count of 0xFFFFFFFF to 0x0 at any time after power up. The HIMIB device does not provide any indication that this has occurred other than the change in the counter value.

Note that the fact a rollover occurs, may or may not be of importance to the managing agent process resident within the repeater. This will depend on the network management protocol (i.e., SNMP) and the management philosophy (i.e., frequency of polling of attributes by the management station).

SECTION

5 MANAGED REPEATER DESIGN

5.1 HIGH LEVEL DESIGN CONSIDERATIONS FOR MANAGED REPEATERS

This section discusses the design of managed repeaters using the Integrated Multiport Repeater Plus (IMR+) and Hardware Implemented Management Information Base (HIMIB) chip-set at a block diagram level. Subsequent sections discuss specific aspects of managed repeater design in more detail.

5.1.1 Introduction to Managed Repeater Design

To perform any kind of remote management on a network device, the Network Management System (usually in a centralized location, such as a Network Operating Center, where access can be controlled), needs to communicate with the remotely distributed components on the LAN. These remote devices, such as repeaters, as well as bridges, routers, network monitoring stations and other type of LAN interconnection devices, gather statistics that are particular to their functionality, location and configuration. The two primary methods to access the locally stored information in these devices is either "in-band" or "out-of-band" communications.

Typically, "in-band" management is used where the LAN itself provides the communications medium to allow the Network Management System (NMS) to request access to, and receive answers from, the remote device. In-band management makes use of the existing LAN infrastructure, so that additional communications paths are not required. The primary concerns in using in-band management are that if the LAN medium becomes disrupted (such as a cable break), or if the LAN becomes congested, it may become difficult to access management data in a timely fashion. Note that in the case of congestion, the actual gathering of network management information must be kept to a reasonable overhead, so that the management task itself does not become the cause of undue loading of the network.

Alternatively, "out-of-band" management may be used. In this case, any communications mechanism can be employed to access the management information stored in a remote device, such as a simple serial or parallel interface port, or a modem etc.. The principle issue for consideration in this case is that an additional communications channel must exist to the remote device. This may already be a requirement where the physical topology of the LAN does not permit direct inter-connection of the NMS and the managed device, such as in the case of a satellite office located many miles from the corporate headquarters. Typically, out of band management uses a lower speed link than the normal LAN connection.

In either case, the NMS must be able to communicate with the specific device, to request information from its local MIB table. This means that each device must be uniquely addressable. In the simplest example, a point-to-point link between the NMS and a single network device would be adequate. However, in a more typical network, the need is to monitor numerous remote devices from one central NMS, in order to provide management capabilities for preventative maintenance as well as fault diagnosis purposes.

For in-band management, the mechanism to address a device is to give it a unique address (either physical or logical). Typically this requires that each device can recog-

nize a packet addressed to it, a function performed by the MAC layer capabilities of a network device. In addition, in order to respond to a request for management data made by the NMS, the remote device must be capable of transmitting a network frame. This again requires the capabilities of a MAC. Specifically in the case of an 802.3 repeater, there is no concept of a MAC or any frame based messaging, since the repeater is strictly specified to operate as a "bit level device". For instance, neither the IMR or IMR+ devices inspect frames received (and subsequently repeated) for integrity at the packet level, such as valid FCS, number of bytes, source or destination address etc. Note however that in order to maintain the per port statistics required by the 802.3 repeater MIB, the HIMIB device does perform frame based error checking and monitoring. The HIMIB chip contains a "receive-only MAC" function for this purpose, although no transmit function is provided. This preserves simple modularity when multiple IMR+/HIMIB devices are interconnected to build a high port count repeater.

In order to construct a managed repeater based on the IMR+ and HIMIB devices, it is necessary to include a bi-directional communications port by which the management data is communicated. For in-band management, this requires the addition of an Ethernet controller device. For out-of-band management, an alternate communications port, such as serial interface, is required. The block diagram of Figure 5-1 shows an example of a typical managed repeater, providing in-band and (optional) out-of-band capabilities.

For in-band management services, the Ethernet controller shown in the example is the Media Access Controller for Ethernet (MACE, Am79C940) device. The modular approach of the IMR+/HIMIB chip-set solution allows the MACE device to be connected either to the IMR+ Expansion Port, or to an AUI port.

To connect to the IMR+ Expansion Port interface requires some additional logic, which can be incorporated within the expansion bus arbitration logic. The details of this scheme are covered later in this chapter. This approach is very flexible, since the NRZ data format used on the expansion bus allows the connection of a wide variety of Ethernet controllers. A management request, received on any one of the repeater ports (on one port of one IMR+ device), will be monitored by the companion HIMIB device. All frames (including frames requesting management data) will be repeated over the expansion bus to all other IMR+ devices in the repeater, and hence will be observed and recognized by the Ethernet controller. The agent process resident in the repeater (and executed by the CPU) will formulate the response to the request, and queue the reply for transmission by the Ethernet controller, which will send the reply using the normal transmit MAC function. The disadvantage of this approach is that the HIMIB device does not gather statistics for traffic received on the IMR+ Expansion Port, since it assume that data present on the expansion bus has already been monitored at the receive port of the IMR+ chip which is sourcing data to the bus. Hence frames generated by the Ethernet controller (replies or notifications from the repeater itself) will not be included within the attributes of the repeater.

In order to ensure the complete counting of all network traffic, including that generated by the repeater itself, it is recommended that the Ethernet controller be connected to one of the repeater ports itself. In a typical repeater consisting of multiple IMR+/HIMIB chip-sets, it is unlikely that all of the AUI ports are required. By connecting a suitable transceiver (such as the TPEX or TPEX+ chip) to the AUI port of any IMR+ device, an additional 10BASE-T port can be configured. Since most high integration Ethernet controllers incorporate a 10BASE-T transceiver function (such as the MACE or PCnet®-ISA devices), the two 10BASE-T devices can be directly connected, effectively forming a "zero length" 10BASE-T link internal to the repeater itself. An example of this interface is detailed later in this section.

5.1.2 Overview of Design Example

The function of a managed repeater will be described using the block diagram example in Figure 5-1.

The design goal for this example was to exhibit the lowest cost 24-port stand alone hub with in-band management and a comprehensive led status display.

At the core of the design are the three blocks containing the IMR+/HIMIB chip-set. Each IMR+ device is a complete "repeater on a chip" including eight 10BASE-T ports and a single AUI port The three devices are connected together via their Expansion Ports, to form a common expansion bus and thereby act as a single logical 24-port repeater. This core design would work as a non-managed repeater without the need for a microprocessor, but since this design is focused at a managed repeater application, a HIMIB device is partnered with each IMR+ chip, to autonomously monitor all nine of the IMR+ chip network ports.

The HIMIB device gathers data from the Management Port, Expansion Port and Port Activity Monitor (PAM) Port of the IMR+ chip, and automatically performs all of the network statistics gathering functions required by the IEEE 802.3k MIB.

The design specifies in-band management, which requires an Ethernet Media Access Controller (MAC) with a system interface. This design uses the Media Access Controller for Ethernet (MACE) device because this offers an 8-bit slave interface, allowing the whole CPU and memory system to be 8 bits wide. Because the HIMIB devices maintain the MIB automatically, an 8-bit CPU such as the 80C188 has more than adequate bandwidth for this application. Limiting the design to an 8-bit bus, minimizes component cost and complexity for the CPU, memory etc., as well as the associated manufacturing issues.

The MACE device may be connected either to the common expansion bus or to a spare AUI port of an IMR+ device. Interfacing to the expansion bus is the lowest cost approach (see Section 5.5). This has the drawback that it is not connected to a managed port of any of the IMR+ devices, so data generated by the MACE device will not be automatically included in the MIB statistics. This can be overcome to some extent by monitoring the MACE device's performance in software.

The other approach is to make the design fully managed by adding a TPEX or TPEX+ device to an unused AUI port on one of the IMR+ chips, and connecting the twisted pair interface of the MACE device directly to the TPEX chip. This would make the MACE appear to the hub as a MAC connected to the AUI port and thus provide full management at the expense of adding a TPEX device (see Section 5.4).

Porting an SNMP agent to the 80C188 processor, with drivers for the HIMIB and MACE devices, would complete the design for a fully in-band managed repeater.

A network manager at another station would then be able to send an SNMP frame to the repeater by setting the source address in the frame directly to the physical address of the MACE device. The MACE device would detect its unique address, and request attention from the 80188 CPU. The CPU would copy the frame to buffer memory and then process the SNMP instruction, which might for example be a request for one of the MIB attributes. The CPU would read this attribute from the appropriate HIMIB device(s), then generate an SNMP frame encapsulating the response, and send this as a reply using the transmit function of the MACE device.

Out-of-band management could be added very easily by adding the optional UART and serial drivers to the design. This could then interface to a modem allowing management to be performed remotely over telephone lines.

Because the CPU has considerable unused bandwidth, the LED display array can be driven using a very simple multiplexing technique capable of driving up to 128 LEDs,

allowing multiple LEDs per port, as well as general repeater status indicators. The CPU simply writes a value to an 8-bit latch where one bit drives one column, and the write to the latch increments a counter with 16 output pins, one of which is driven active at any one time to drive a row of the matrix.

5.1.3 Detailed Design Notes

Note that Figure 5-1 shows a much simplified block diagram. All of the necessary components are shown, but many connections are omitted for clarity.

Expansion Bus

The common expansion bus is formed by the arbitration logic between the Expansion Port on each IMR+ device. See the discussion of expansion bus design in Sections 5.2, 5.3 and 5.5.

Modules

These are the transformer/filter modules needed for the 10BASE-T interface (total 24 needed) and the transformer module needed for the AUI interface.

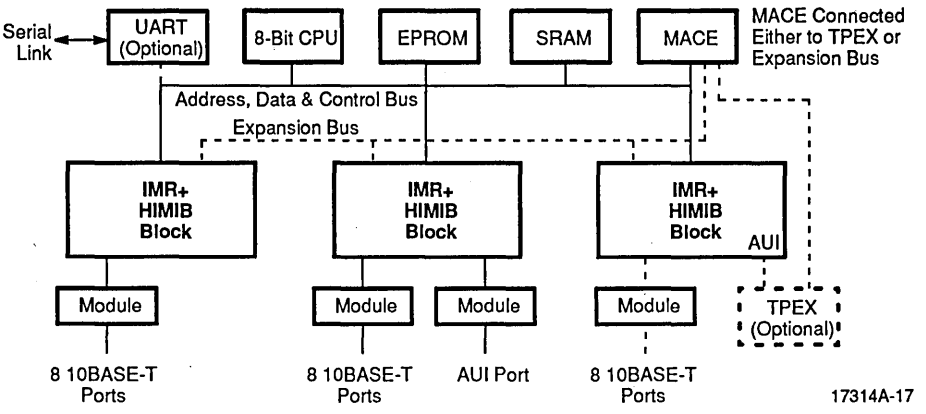
80C188

The AMD 80C188 is an excellent choice for this design because of its low cost, 8-bit bus and high integration. It has sufficient integrated chip select signals to control all of the devices on its bus. It handles all transfers between the MACE device and buffer RAM using either string move instructions or DMA transfers, with the MACE mapped as a memory slave device. It can also utilize the same 20 MHz clock as the Ethernet devices.

MACE Device

The MACE device has on-chip FIFOs allowing it to connect directly to the relatively slow 8-bit CPU bus of this example repeater. It does not promiscuously receive all of the frames repeated onto the expansion bus, hence the total bus bandwidth requirements are modest, and dependent only on the volume of management request/response packet activity.

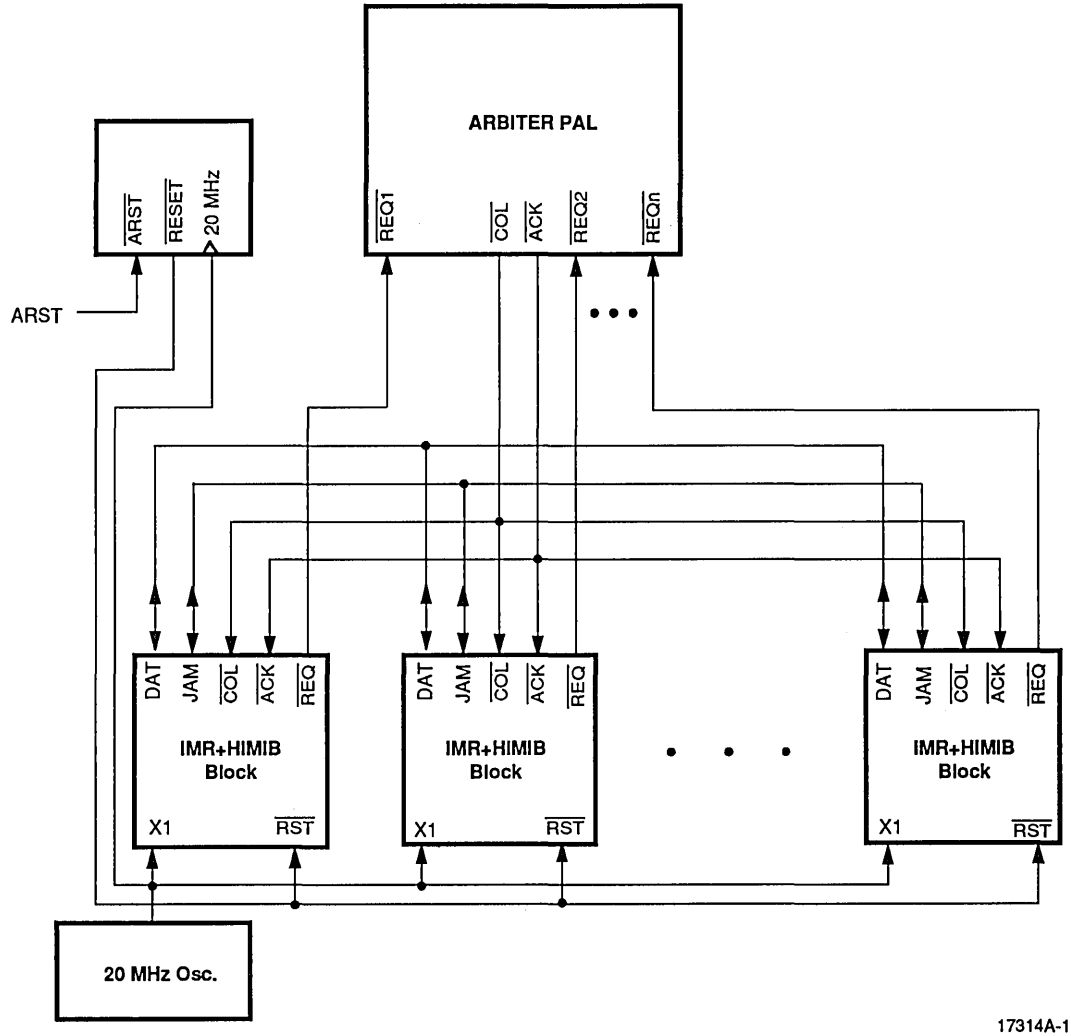
Figure 5-1 Simplified Block Diagram of 24-Port Repeater with In-Band Management



5.2 FIXED PORT REPEATER DESIGN

This section discusses the design of repeaters to support a fixed maximum number of ports using centralized arbitration.

Figure 5-2 Repeater Design Using Multiple IMR+/HIMIB Chip-Sets



17314A-18

Figure 5-2 shows a block diagram for a repeater based on multiple IMR+/HIMIB chip-sets using centralized arbitration. This is the same design as that described in Section 3.9 except that the HIMIB device has been added and that the use of buffers on the DAT and JAM lines is suggested. See the "IMR+ Device Repeater State Machine Description" in Section 3 for a detailed description of the Expansion Port operation. See the "IMR+/HIMIB Interface" section for a description of the IMR+/HIMIB chip-set block.

Note that the IMR+/HIMIB chip-set block includes buffering in Figure 5-3. In the case of this specific design (Figure 5-2) the buffers may be omitted for a solution based on four

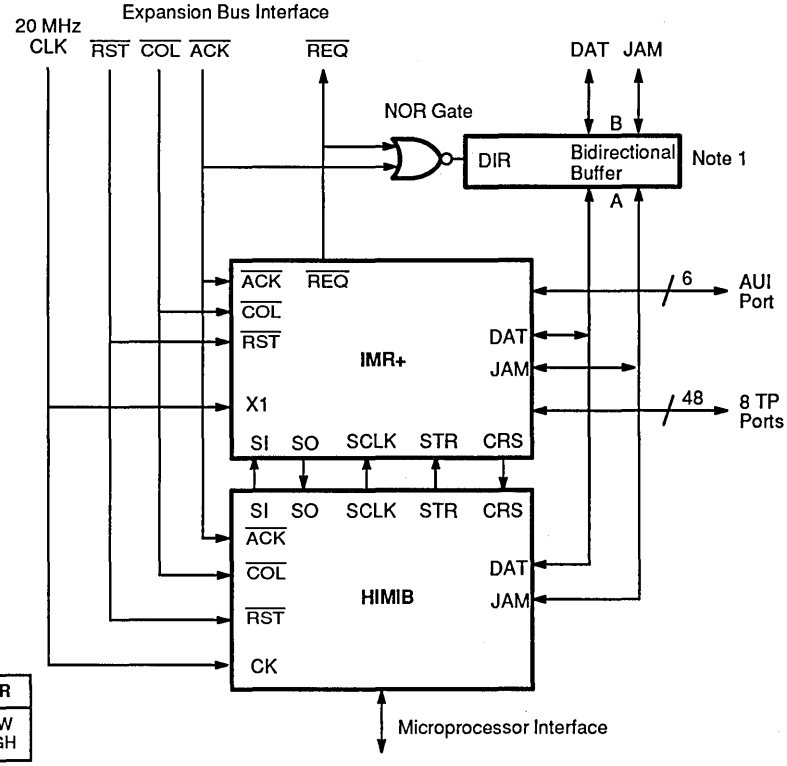
IMR+/HIMIB chip-sets provided the capacitance of the PCB traces for the DAT and JAM signals does not exceed 28 pF.

5.3 IMR+/HIMIB INTERFACE

Figure 5-3 below shows the interconnections between the IMR+ and HIMIB devices. It also shows a bi-directional buffer for the DAT and JAM signals and the direction control logic for this buffer (the buffer is always enabled). The buffer is only necessary if the load to be driven by the IMR+ device DAT and JAM output buffers exceeds 100 pF. If it can be guaranteed that the load will not exceed this figure the DAT and JAM pins of the IMR+ chip can drive the expansion bus directly.

The sense of the direction control must be chosen such that the buffers drive the expansion bus when both $\overline{REQ}$ and $\overline{ACK}$ are low (asserted).

Figure 5-3 Detail of IMR+ HIMIB Block



Note 1:

Direction	DIR
B → A	LOW
A → B	HIGH

5.4 AUI/MAC INTERFACE

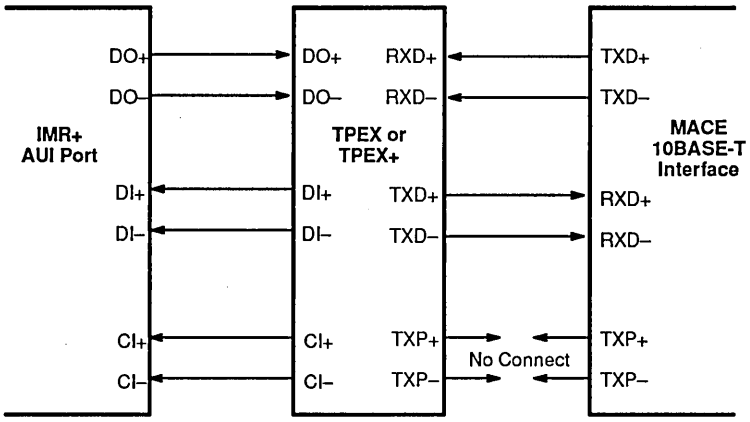
A MAC device may be connected to an IMR+ chip as a fully managed port by adding a TPEX device to an AUI port of one of the IMR+ chips, and connecting the twisted pair interface of the MACE device directly to the TPEX chip. This would make the MAC appear to the hub as a MAC connected to the AUI port and thus provide full management at the expense of adding a TPEX device.

The IMR+ device cannot be connected to an Ethernet MAC by simply connecting the two AUI ports together, since both devices effectively possess a DTE implementation of the AUI (versus a MAU implementation). This is to say that both the IMR+ and MACE

devices expect the Collision In (CI±) pair of the AUI to be a differential input, and collisions to be signaled using a 10 MHz pulse train (as supplied by a MAU). Hence the simplest way to connect the two devices is to connect a 10BASE-T MAU, such as the TPEX device, to the IMR+ chip's AUI port, and effectively construct a "zero length" 10BASE-T connection between the TPEX and the integrated 10BASE-T port of the MACE device (see Figure 5-4), using only PCB traces.

The RXD and TXD pins of the TPEX and MACE device are simply connected together, and the TXP outputs are left open.

Figure 5-4 IMR+/TPEX+ and MACE Devices Connected for In-Band Management



17314A-20

5.5 EXPANSION BUS/MAC INTERFACE

5.5.1 IMR+ Expansion Port Basics

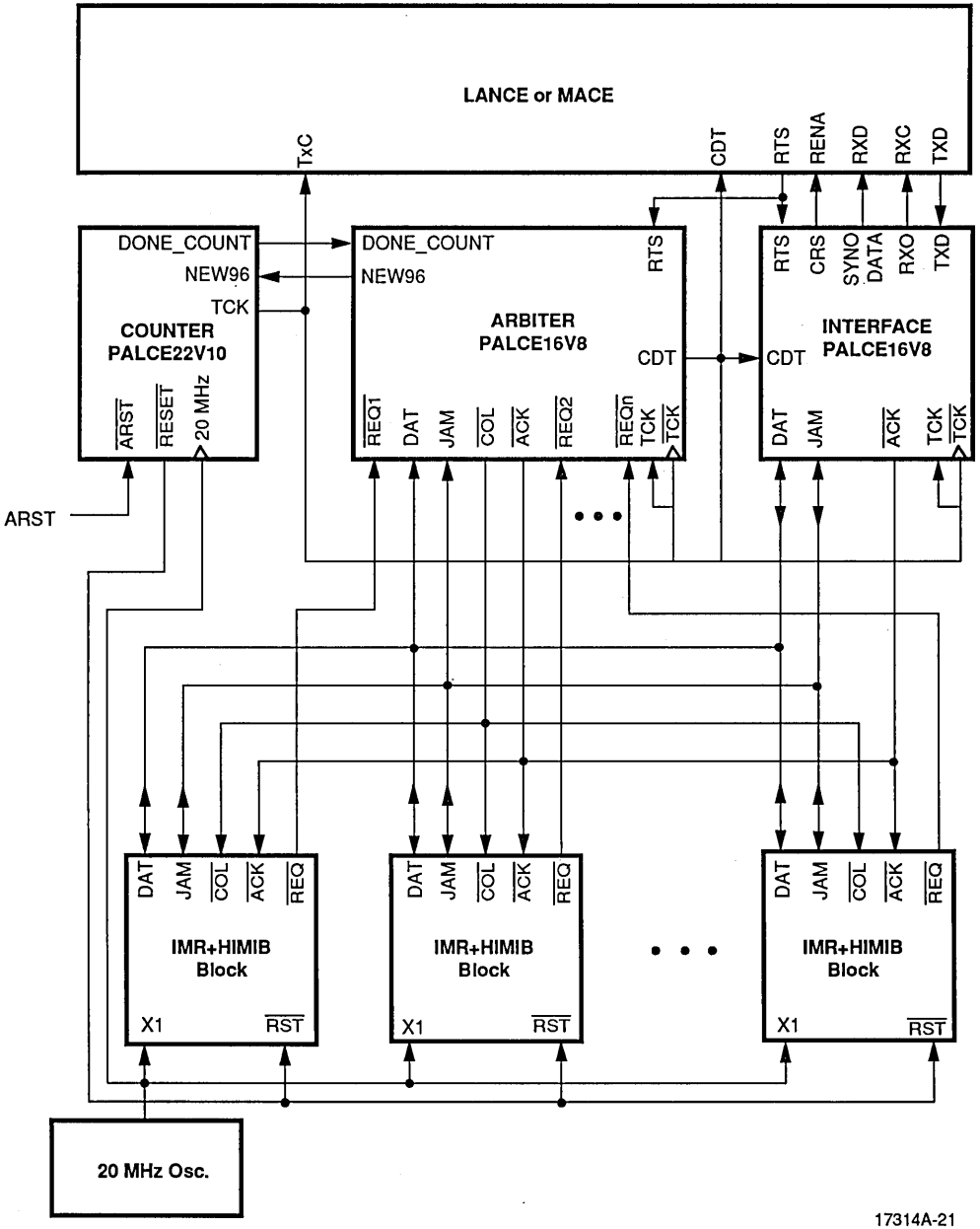
Using the IMR+ Expansion Port, a design based on the IMR+ chip can be cascaded to provide increased port density, whilst only incurring the delay of a single logical repeater. A common expansion bus is formed by implementing a simple arbitration logic circuit between the Expansion Port on each IMR+ device. Essentially, each IMR+ must request access (using its $\overline{REQ}$ output) to the common DAT and JAM bi-directional lines. Providing only one IMR+ device asserts $\overline{REQ}$, the arbiter will return a common $\overline{ACK}$, which signals the requestor it can drive the DAT/JAM lines as outputs, and signals all non-requesting devices that they should configure DAT/JAM as inputs. If multiple devices request the bus, the arbiter returns a collision indication ($\overline{COL}$), which causes each IMR+ device to output jam on its network ports independently.

All repeated data in an IMR+ or multi-IMR+ device design is available on the expansion bus. Data on the DAT line of the expansion bus is passed in NRZ format, that is already decoded from the Manchester encoded data passed over the network. Because of this, any Media Access Controller (MAC) device with a General Purpose Serial Interface (GPSI) capability can be connected to the expansion bus. Most popular LAN controllers support a derivative of the GPSI, including the LANCE (Am7990), MACE (Am79C940) and PCnet-ISA (Am79C960) devices.

In a repeater based on the IMR+/HIMIB chip-set, the MAC is not used to gather statistics since this function is performed by the HIMIB device(s). The MAC provides in-band network addressability that allows a remote Network Management Station (NMS) to request access to statistical data that the repeater maintains (in the Repeater MIB), and to execute specific actions such as enabling and disabling a port.

Figure 5-5 shows how to interface a repeater based on two or more IMR+/HIMIB chip-sets to an Ethernet controller such as the MACE or LANCE chips, via the GPSI and the common expansion bus. The IMR+/HIMIB block is shown in detail in Figure 5-3. Note that the buffering shown in the example of Figure 5-3 may be omitted for a solution based on three IMR+/HIMIB chip-sets, providing the capacitance of the PCB traces plus the PALCE devices on the DAT and JAM lines does not exceed 46 pF.

Figure 5-5 Ethernet MAC Interfaced to Expansion Bus for In-Band Management



5.5.2 Principle of Operation

The basic function of the three PALCE devices shown in Figure 5-5 is to interface multiple IMR+ devices and the MAC together to form one logical repeater. A major function of the logic is to make the MAC appear to the common expansion bus as if it is another IMR+ device.

The arbiter device controls the $\overline{COL}$ and $\overline{ACK}$ signals of the expansion bus, asserting $\overline{ACK}$ when one and only one request signal is active. In this design a request signal is the assertion of either a $\overline{REQ}$ line from any IMR+ device or the RTS line from the MAC device.

If more than two requests are active at once the arbiter will signal a collision to all devices by asserting the $\overline{COL}$ signal to the IMR+ devices and the CDT signal to the MAC.

The arbiter grants access to the DAT/JAM resources of the expansion bus by asserting $\overline{ACK}$ to all IMR+ devices. The IMR+ that has asserted its $\overline{REQ}$ will see $\overline{ACK}$ asserted and start to repeat data over the expansion bus via the DAT signal.

The CDT signal sent to the MAC is always asserted when $\overline{COL}$ is true, but is also asserted in other cases. These cases (described in the logic equations below) are basically the same as those that occur in the internal logic of each IMR+ device. The most complex of these cases is to hold CDT asserted for 96 bit times after any new event occurs on any repeater port. This is necessary because each IMR+ chip will ensure that the minimum length event repeated to the network on any of its 10BASE-T ports or its AUI port is 96-bit times long.

If the 96-bit counter is not implemented, the solution may generate collisions or unacceptable short IPG times when the MAC attempts to transmit a packet, as described below.

Assuming one IMR+ device receives a short event on one of its ports, and successfully arbitrates for the expansion bus. It starts to repeat the incoming data on the bus, but data reception stops before 96 bit times have elapsed. The receiving IMR+ device will de-assert its request when the event terminates prematurely, will enter the receive collision state (as will all other IMR+ devices connected to the same expansion bus), and extend the received short event to a minimum size fragment (96 bits) on all active transmit ports. The expansion bus will appear to have no activity whilst all $\overline{REQ}$ lines are inactive. If the MAC has a pending transmit packet, it would commence its IPG timer when the expansion bus appeared to become inactive, and subsequently assert its RTS signal and successfully gain control of the bus.

In this case, a collision may result, or an extremely short IPG could be generated. Despite the expansion bus appearing inactive, each IMR+ device will be in the process of extending the short event to 96 bits, by transmitting a jam pattern to all active ports.

This example design eliminates this problem by having a 96-bit counter that is reset to the beginning each time a new event is detected on any repeater port.

The arbiter detects the events, and passes a signal (NEW96) to the counter to tell it to restart the count. The arbiter holds CDT to the MAC asserted, at least until the counter asserts DONE_COUNT.

The other functions of the arbiter are to generate the synchronous reset signal required by the IMR+ and HIMIB devices, and to generate the 10 MHz bit clock. The bit clock is used to generate the transmit and receive clocks of the MAC, and is synchronized with $\overline{RESET}$ so that it is in phase with the internal 10 MHz bit clocks of the IMR+ devices.

The interface device essentially translates the GPSI compatible signals of the MAC into the IMR+ compatible signals of the expansion bus.

It multiplexes the unidirectional transmit data (TXD) and receive data (RXD) signals of the MAC onto the bi-directional DAT line of the expansion bus, generates the receive enable (RENA) signal to the MAC and synchronizes RXD with the receive clock (RXC)

5.5.3 Counter Logic Equations

Upon $\overline{\text{RESET}}$, the counter is initialized to the terminal value of 60H (96), and remains there until triggered. If (re)triggered by NEW96, the counter is reset to 00H. The counter will count at 10 MHz until it reaches the terminal value, and will stop there until triggered. TCK and $\overline{\text{RESET}}$ are also generated here.

```
TCK    := /TCK * /RESET * /ARST                                  Generates TCK for MAC use

RESET  := ARST                                                   Synchronizes the async
                                                                 reset signal

Q0     := /Q0 * /TCK * /DONE_COUNT * /RESET * /NEW96              Toggle if still counting LOW
                                                                 at reset. LOW when triggered
                                                                 by NEW96
       + Q0 * (TCK + DONE_COUNT) * /RESET * /NEW96                Stop at terminal value

Q1     := /Q1 * Q0 * /TCK * /DONE_COUNT * /RESET * /NEW96         LOW at reset, LOW when
                                                                 (re)triggered by NEW96
       + Q1 * (/Q0 + TCK + DONE_COUNT) * /RESET * /NEW96

Q2     := /Q2 * Q1 * Q0 * /TCK * /DONE_COUNT *                   LOW at reset, LOW when
          /RESET * /NEW96                                        (re)triggered by NEW96
       + Q2 * (/Q1 + /Q0 + TCK + DONE_COUNT) * /RESET * /NEW96
Q3     := /Q3 * Q2 * Q1 * Q0 * /TCK * /DONE_COUNT *
       /RESET * /NEW96                                           LOW at reset, LOW when
                                                                 (re)triggered by NEW96
       + Q3 * (/Q2 + /Q1 + /Q0 + TCK + DONE_COUNT) *
       /RESET * /NEW96

Q4     := /Q4 * Q3 * Q2 * Q1 * Q0 * /TCK * /DONE_COUNT *         LOW at reset, LOW when
       /RESET * /NEW96                                           (re)triggered by NEW96
       + Q4 * (/Q3 + /Q2 + /Q1 + /Q0 + TCK + DONE_COUNT) *
       /RESET * /NEW96

Q5     : = /Q5 * Q4 * Q3 * Q2 * Q1 * Q0 * /TCK *                 Toggle if all lower bits HIGH
       /DONE_COUNT * /NEW96                                      And still counting LOW if
                                                                 (re)triggered by NEW96
       + Q5 * (/Q4 + /Q3 + /Q2 + /Q1 + /Q0 + TCK +               Stay if any lower bit is LOW or
       DONE_COUNT) * /NEW96                                      if done counting
       + RESET                                                   HIGH at reset

Q6     := /Q6 * Q5 * Q4 * Q3 * Q2 * Q1 * Q0 * /TCK *             LOW when (re)triggered by
          /DONE_COUNT * /NEW96                                   NEW96
       + Q6 * (/Q5 + /Q4 + /Q3 + /Q2 + /Q1 + /Q0 + TCK +
       DONE_COUNT) * /NEW96
       + RESET                                                   HIGH at reset

DONE_    COUNT    = Q6 * Q5 * /Q4 * /Q3 * /Q2 * /Q1 * /Q0         Terminal count (96)
```

5.5.4 Arbiter Logic Equations

```
ACK    = REQ1 * /REQ2 * ... /REQn * /RTS                         ACK is asserted when one and
                                                                 only one expansion bus request
                                                                 is active
       + /REQ1 * REQ2 * ... /REQn * /RTS
                                                                 %
                                                                 %
       + /REQ1 * /REQ2 * ... REQn * /RTS
       + /REQ1 * /REQ2 * ... /REQn * RTS
       + /REQ1 * /REQ2 * ... /REQn * RTSCLS * ACKCLK * /CDT      Term to extend ACK when MAC
                                                                 device finishes sourcing data
                                                                 onto expansion bus
ACKCLK := ACK                                                    Clock delayed ACK signal
                                                                 (ACK * ACKCLK defines first
                                                                 clock period with valid DAT
                                                                 and JAM)
COL    = /ACK * (REQ1 + REQ2 + ... REQn + RTS)                   COL is asserted when more than
                                                                 one expansion bus request is
                                                                 active
COLCLK := COL                                                    Clock delayed COL signal
CDT    :=                                                        CDT causes the MAC device to
                                                                 back off/stay off the
                                                                 expansion bus
       COL                                                       Arbiter detects collision
       + ACK * ACKCLK * JAM                                      Single IMR collision
       + COLCLK * ACK                                            Holds CDT active during 'dead'
                                                                 clock cycle
```

```
      + /DONE_COUNT * CDT                    clock cycle
                                             Maintains CDT (suppresses new
                                             MAC requests) until 96 bit
                                             time-out
XCOL  :=                                     Present transmit collision
                                             state
      ACK * ACKCLK * JAM * /DAT              Ongoing single IMR transmit
                                             collision
      + COL                                  Ongoing multiple IMR transmit
                                             collision
      +COLCLK * ACK                          Ongoing multiple IMR transmit
                                             collision ('dead' clock
                                             cycle)
RTSCLK := RTS                                Clock delayed RTS signal
                                             (RTS*RTSCLK defines first
                                             recognized RTS from MAC)
NEW96 :=                                     Triggers or re-triggers
                                             counter
                                             (96 bit times)
      ACK * /ACKCLK * /CDT                   New repeater data (start of
                                             repeated packet, ACK/ with no
                                             existing collision) (Trigger
                                             only)
      + ACK * ACKCLK * JAM * /DAT * /XCOL    New transmit collision
                                             (Single IMR)
      + COL * /XCOL                          New transmit collision
                                             (Multi-IMR)
```

5.5.5 Interface Logic Equations

```
DAT.TRST = ACK * ACKCLK * RTSCLK             Enabled if MAC request gets
                                             arbiter ACK (delayed)
DAT   = SYNCDATA * /CLK                      Synchronized and latched
                                             MAC data
                                             (if enabled) guarantees
                                             hold time
                                             for slave IMRs
      + DAT * CLK
      + DAT * SYNCDATA
      + CDT                                  Overriding HIGH if collision
                                             exists (if enabled)
JAM.TRST = ACK * ACKCLK * RTSCLK             Enabled if MAC request gets
                                             arbiter ACK (delayed)
JAM   = CDT                                  HIGH if collision exists
ACKCLK        := ACK                         Clock delayed ACK signal
                                             (ACK * ACKCLK defines
                                             first clock
                                             period with valid DAT and JAM)
CRS   = ACK * ACKCLK * /JAM * /CDT           Single active IMR or MAC
                                             sourcing data
      + RCKEN                                Term to extend CRS at end of
                                             MAC receive
SYNCDATA :=
      /RTS * DAT * ACK * ACKCLK              Synchronized receive data
                                             for the MAC
      + TXD * RTS                            Synchronized TxD data for
                                             MAC transmit
RXC   = RCKEN * /CLK                         Inverted clock for
                                             synchronized
                                             data for MAC receive
RCKEN := ACK * ACKCLK * /RTS * /RTSCLK * /JAM RCLK enabled if non-MAC data
                                             transfer (MAC is slave device)
      + ACK * RTS * /CDT                     RCLK enabled on transmit
                                             to allow
                                             data loopback to MAC
RTSCLK        := RTS                         Clock delayed RTS signal
                                             (RTS * RTSCLK defines first
                                             recognized RTS from MAC)
```

5.6 HIMIB DEVICE MICROPROCESSOR INTERFACE

Figure 5-6 shows one example of how the HIMIB device may be interfaced to a popular microprocessor such as the 80C188. This is simply one way that this may be accomplished, and there are many other possible solutions

A 10 MHz 80C188 is employed since this can use the same 20 MHz clock as the HIMIB device. The RDY output of the HIMIB device must be connected to one of the ready

inputs of the CPU so that the HIMIB device can insert wait states in the CPU cycle. Both the ARDY and SRDY inputs of the 80C188 require some kind of synchronization to the CLKOUT clock of the CPU. In this case SRDY was chosen, and a D type flip flop such as half a 74LS74 is used to guarantee that the setup and hold times for this input are met.

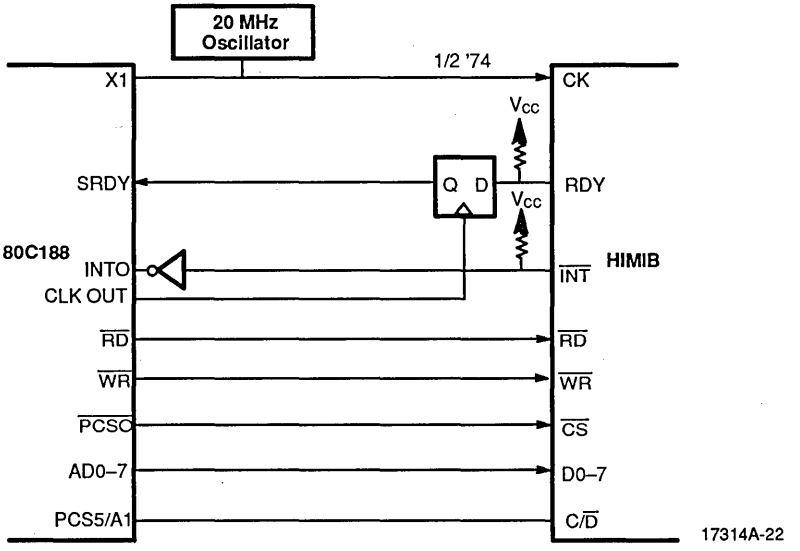
The 80C188 has several programmable chip select outputs, which may be used to eliminate external address decode logic. In this case $\overline{PCS0}$ was used although any of these outputs would work.

The HIMIB requires one non-multiplexed address line from the CPU to drive the C/$\overline{D}$ pin. This design uses the PCS5/A1 line to provide this signal, thus avoiding the need for an external latch.

The interrupt inputs of the 80C188 are positive true, while the $\overline{INT}$ output of the HIMIB device is negative true. An inverter such as an LS04 is thus needed.

Note that because the RDY and $\overline{INT}$ outputs of the HIMIB device are open drain, pullups are required.

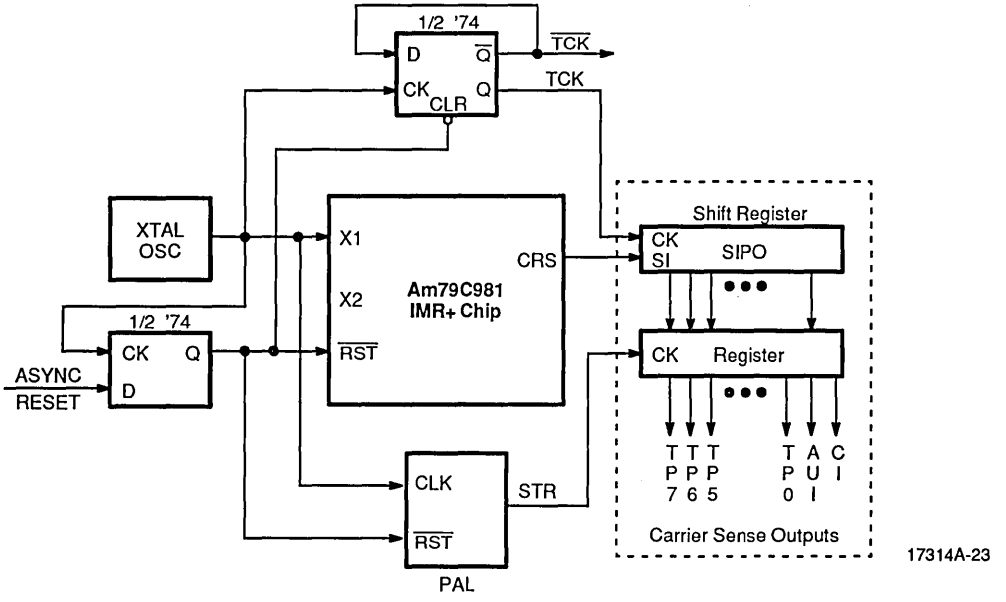
Figure 5-6 80C188 to HIMIB Connection Example



5.7 MANAGED REPEATER STATUS INDICATORS

When the IMR+ and HIMIB devices are used in conjunction, after a hardware reset the HIMIB chip will reprogram the IMR+ device's STR signal to become an input. Applications requiring that the IMR+ chip's carrier sense status be brought out must externally generate the original STR signal. This is done as shown in Figure 5-7 by counting the clock cycles from the time the IMR+ device is reset.

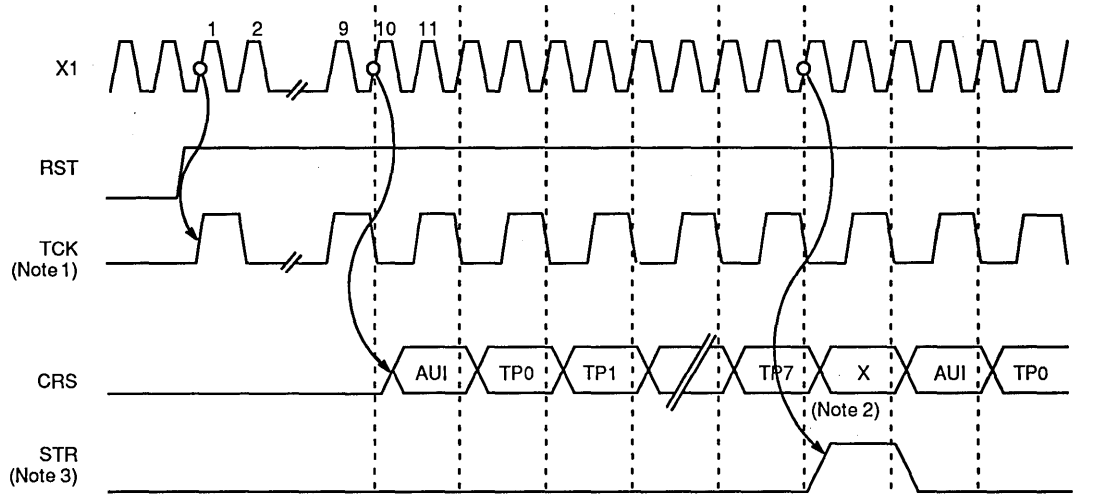
Figure 5-7 Port Activity Monitor Implementation Using Externally Generated STR Signal



Note that once the HIMIB device detects the presence of the IMR+ chip, the HIMIB will enable the CI reporting (C bit in the IMR+ Chip Programmable Options Register) function in the IMR+ chip. The IMR+ device will therefore report AUI CI activity using the bit position immediately after the last twisted pair carrier status (TP7) and prior to the combined AUI CI/DI status. Therefore the AUI CI activity is effectively reported during the bit position corresponding to the occurrence of the STR signal.

Figure 5-8 shows the timing relationship between the reset, with respect to the clock input, and the carrier sense (CRS) output from the IMR+ chip. The carrier sense outputs are valid following the tenth XCLK after reset becomes inactive, and the CRS bits will be clocked out at the rate of one bit every two XCLK periods thereafter.

Figure 5-8 Carrier Sense (CRS) and Strobe (STR) Timing Relationship



17314A-24

Notes:

1. *Externally generated signal illustrates internal IMR+ chip clock phase relationship.*
2. *IMR+ chip standalone, X will be low. When attached to a HIMIB device, X reflects the state of the CI pair.*
3. *STR signal is not available when the IMR+ chip is attached to HIMIB device, and must be generated externally.*

5.8 MODULAR REPEATER COLLISION ARBITRATION

The IMR+ Expansion Port allows for modular expansion. An expandable repeater based on modular 'plug-in' cards can be built by sharing the arbitration duties between a backplane bus architecture and several separate repeater modules. Each repeater module performs the local arbitration function for the IMR+ devices on that module, and provides signals to the backplane for use by a global arbiter.

Figure 5-9 and Figure 5-10 show an expandable repeater based on 3 modules that each use 3 IMR+ devices. In this design, each module provides 24 10BASE-T ports and requires a single PALCE16V8-15 to perform the local arbitration function. The backplane portion of the modular repeater consists only of the global arbiter, implemented with a single PALCE16V8-15.

Note that in each module care must be taken to ensure that the IMR+ devices do not drive a load greater than 100 pF with their DAT and JAM pins.

A module similar to that shown in Figure 5-10 may be designed with up to four IMR+/HIMIB chip-sets provided that the capacitance seen by the DAT and JAM pins of the IMR+ devices does not exceed 28 pF. This capacitance will include the PCB traces and the input capacitance of the buffer device.

Figure 5-9 Modular Repeater

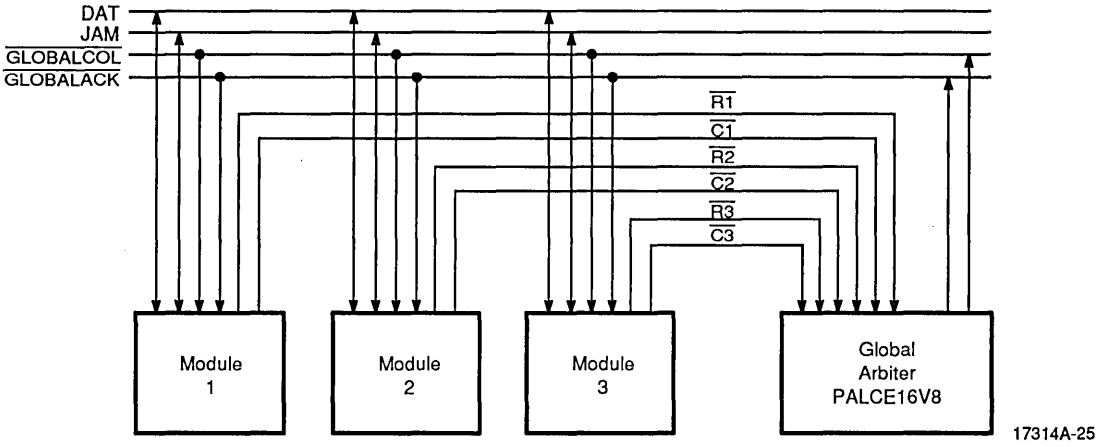
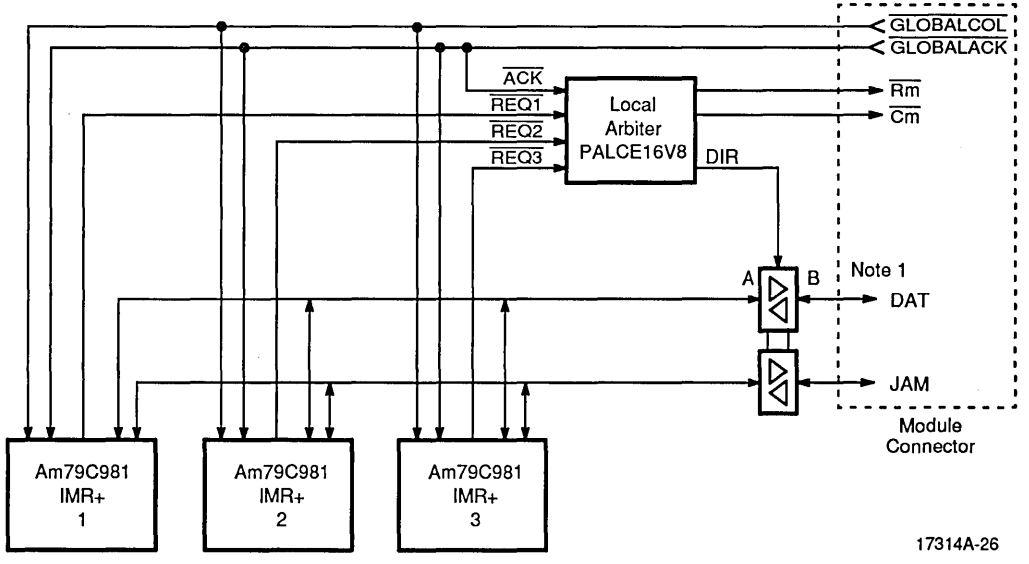


Figure 5-10 Repeater Module with Three IMR+ Devices



Note 1:

Direction	*DIR*
B → A	*LOW*
A → B	*HIGH*

Local Arbiter Logic Equations (for n IMR/IMR+ devices per module):

```
Rm      = REQ1 * /REQ2 * /REQ3 * ... /REQn
Rm      = /REQ1 * REQ2 * /REQ3 * ... /REQn
Rm      = /REQ1 * /REQ2 * REQ3 * ... /REQn
                          %
                          %
                          %
Rm      = /REQ1 * /REQ2 * /REQ3 * ... REQn
Cm      = /Rm * (REQ1 + REQ2 + REQ3 + ... REQn)
DIR     = Rm * GLOBALACK
```

The DIR signal is HIGH when the module drives the DAT and JAM backplane bus signals. A single PALCE16V8 can support up to eight IMR/IMR+ devices per module.

MODULARR.CDR
Global Arbiter Logic Equations (for m modules):

```
GLOBALACK   = /C1 * /C2 * /C3 * ... /Cm * R1 * /R2 * /R3 * ... /Rm
            + /C1 * /C2 * /C3 * ... /Cm * /R1 * R2 * /R3 * ... /Rm
            + /C1 * /C2 * /C3 * ... /Cm * /R1 * /R2 * R3 * ... /Rm
                          %
                          %
                          %
            + /C1 * /C2 * /C3 * ... /Cm * /R1 * /R2 * /R3 * ... Rm
GLOBALCOL   = /GLOBALACK * (R1 + R2 + ... Rm)
            + (C1 + C2 + C3 + ... Cm)
```

A single PALCE22V10 can perform the global arbitration for up to eight modules.

A distributed version of the global arbitration scheme is relatively easily achieved by using the $\overline{REQ}$ from each IMR+/HIMIB chip-set (or group) to drive a unique "signature" onto a backplane. This signature can then be used to arbitrate at each module, similar to the arbitration scheme employed in the Microchannel Architecture. The primary issue in any such design is the $\overline{REQ}$ to ACK arbitration must resolve in 100 ns. See the "Slow $\overline{ACK}$ or $\overline{COL}$ Signals" section for additional detail.

SECTION

6 LAYOUT RECOMMENDATIONS

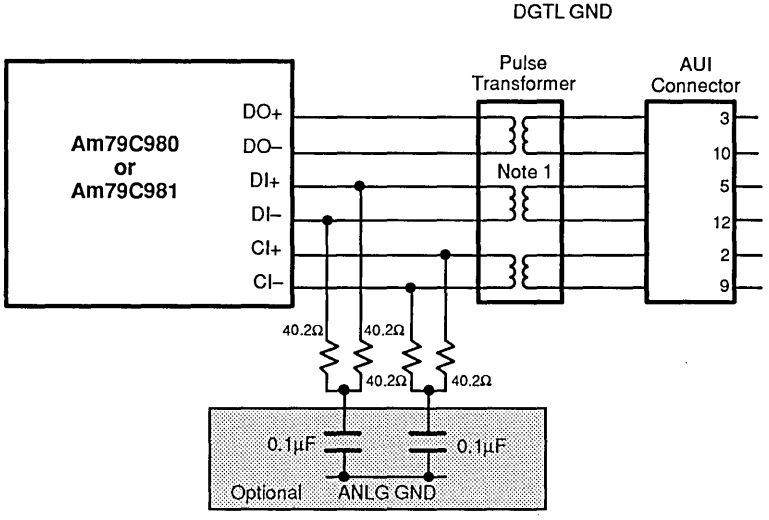
The following layout recommendations are based upon the experiences of numerous IMR customers and IMR+ beta sites. These guidelines are somewhat conservative and are intended to minimize the engineering required to achieve an appropriate EMI/RFI emission certification, rather than component count.

6.1 THE ATTACHMENT UNIT INTERFACE

The AUI port of the IMR+ device can drive a full length (50 m) AUI cable via an isolation transformer and 15 pin D-Type connector. An external transceiver can be connected to the AUI for remote MAU applications. Alternatively, an integrated MAU can be incorporated into the repeater, by connecting a suitable transceiver chip (for 10BASE-T, 10BASE2 or 10BASE-F/FOIRL) to the AUI via an isolation transformer.

To provide a connection capability to a remote MAU requires a 15-pin D connector, 75 µH isolation transformer, termination resistors, and capacitors to reduce common mode noise (Figure 6-1). The IEEE 802.3 Section 7 requirements dictates that the DI± and CI± inputs be terminated with 78.0 ± 5 Ω. Two matched 40.2 Ω ± 1% resistors are used in parallel to accomplish this. The 0.1 µF capacitor tying the center point of the resistors to ground is a filter for common mode noise that may be induced from external sources.

Figure 6-1 IMR/IMR+ Device With External AUI Connector



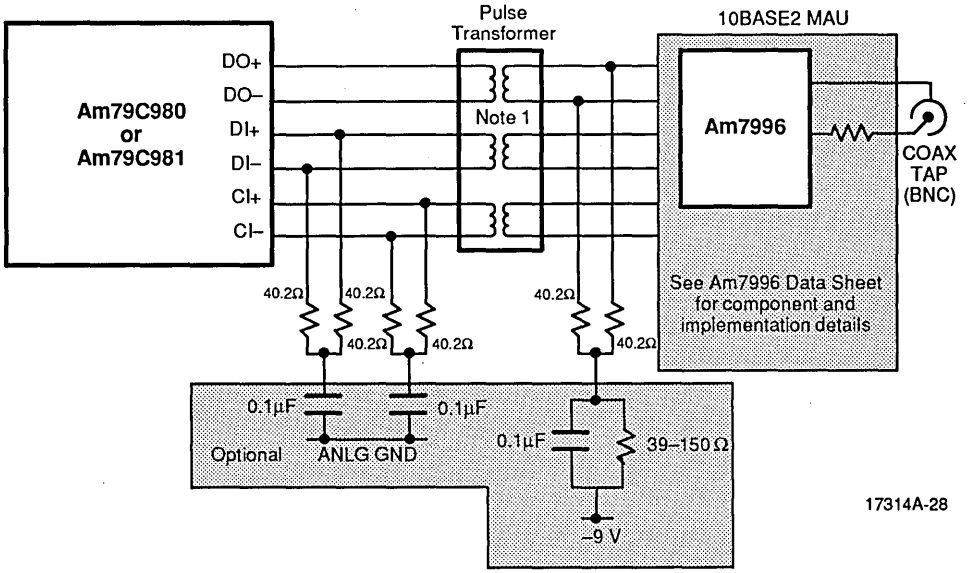
Note:

1. *Compatible AUI transformer modules, with a brief description of package type and features are included in the Appendix.*

As an example of an integrated MAU, a detailed AUI to Cheapernet attachment is described in the Am7996 IEEE 802.3 Ethernet/Cheapernet Transceiver Data Sheet, in the AMD Ethernet/IEEE 802.3 Family Data Book (PID #14287B). Note that the SQE Test function of any transceiver connected to a repeater must be disabled. In the case of the Am7996, the $\overline{\text{SQE TEST}}$ pin must be connected to the positive logic supply (0 V) for the transceiver.

Special attention should be observed in order to minimize potential coupling of noise generated by the DC/DC switching convertor, into the transceiver circuit, and hence to the outside world (via the BNC connector). One approach which has been used in conjunction with the Am7996, is to modify the termination of the DO± circuit. Figure 6-2 shows a 0.1 µF capacitor connecting the parallel 40.2 Ω resistors to –9 volts. This capacitor helps to minimize FCC problems associated with noise from the DC/DC power converter. While this capacitor helps filter out common mode noise, the normal bias point of the Am7996 DO± receiver is altered by the AC coupling effect (the recommended circuit for the Am7996 DO± circuit normally has the center point of the DO± termination resistors tied directly to the –9V generated by the DC/DC convertor). To mitigate this, a resistor is added in parallel to provide a DC bias path. The value of the parallel resistor is not critical, but it should be small enough to maintain 78.0 ± 5 Ω when combined with the 40.2 Ω resistors, and large enough not to reduce the effect of the common mode filter. A value from 39–150 Ω is suggested. It may be easiest to use 40.2 Ω so that a single resistor pack can be used for all terminations.

Figure 6-2 IMR/IMR+ Device With Integrated 10BASE2 MAU



Note:

1. *Compatible AUI transformer modules, with a brief description of package type and features are included in the Appendix.*

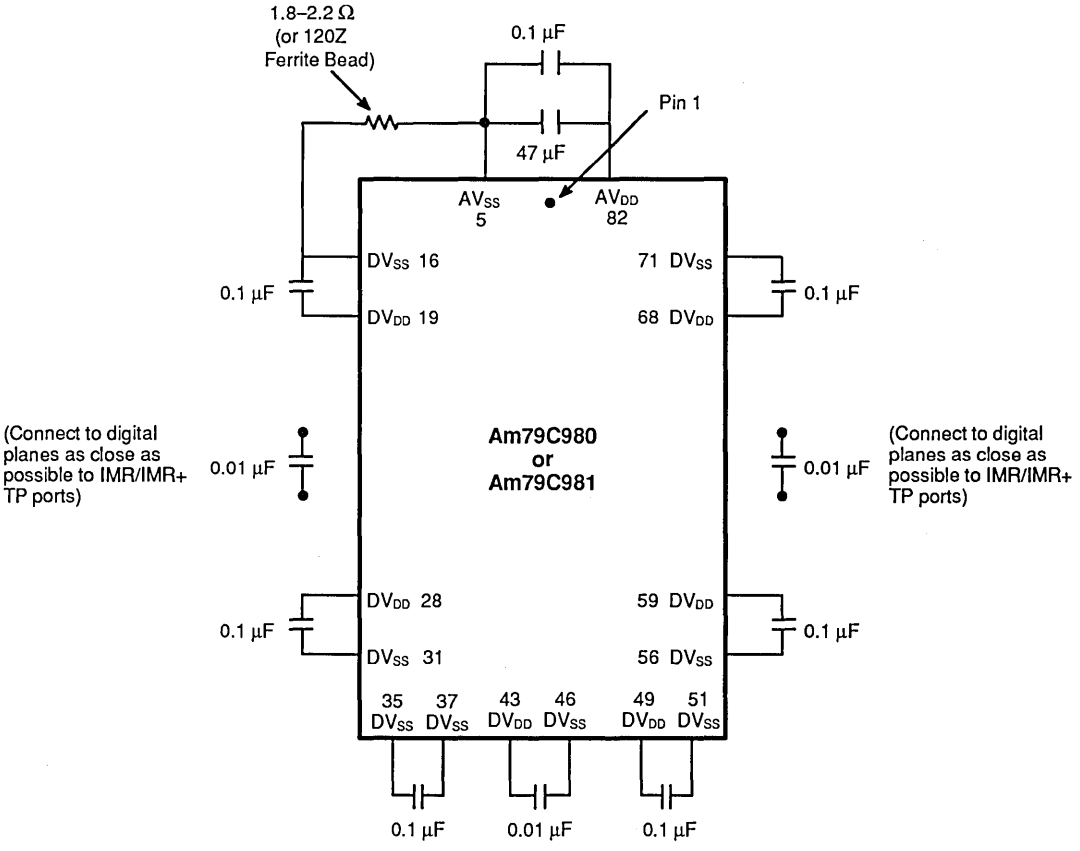
6.2 DECOUPLING

The IMR+ chip contains both analog and digital elements. Separate power pins are provided for the analog sections, the digital portion of the TP line drivers, and the digital core logic.

DV_{CC} Pin #	DV_{SS} Pin #	Function
19	16	TP ports 0 & 1 drivers
28	31	TP ports 2 & 3 drivers
43, 49	35, 37, 46, 51	Core logic & expansion control pins
59	56	TP ports 4 & 5 drivers
68	71	TP ports 6 & 7 drivers

Care should be taken in the board design to minimize the coupling of noise from the power supply and digital logic to the analog power pins. Decoupling capacitors should be placed as close to the appropriate V_{DD} and V_{SS} pins as possible. Figure 6-3 shows the recommended decoupling values for the IMR+ chip.

Figure 6-3 IMR/IMR+ Device Power Supply Decoupling Recommendations



17314A-29

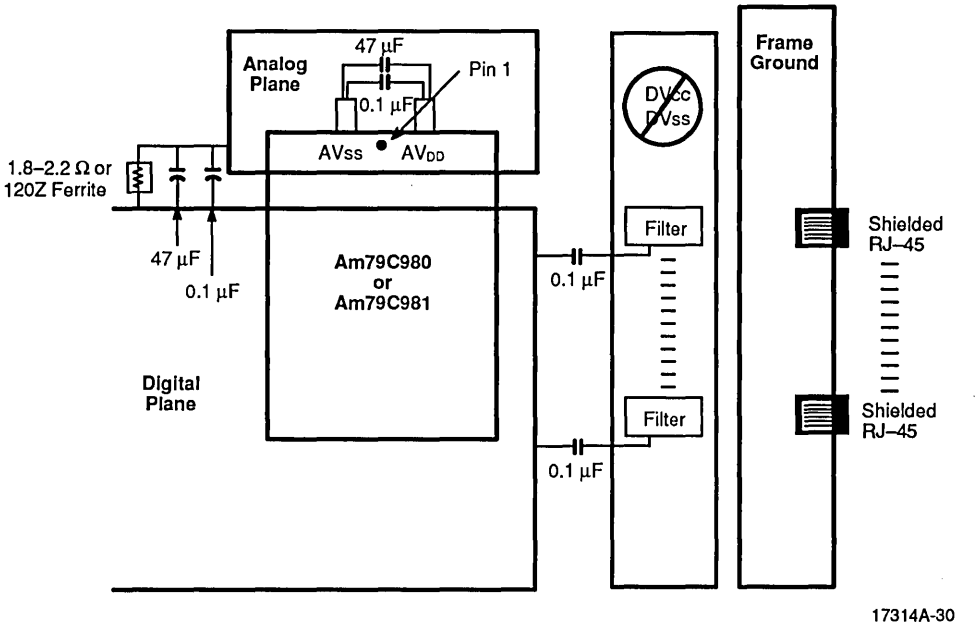
The supply pins for the TP ports are directly decoupled with 0.1 μF capacitors. Additional 0.01 μF capacitors are connected to the digital planes near the TP ports to increase the effective range of filtering. This combination of 0.01 μF and 0.1 μF capacitors are used for the core logic as well. X7R type capacitors are recommended if surface mount technology is used. Experience has shown that the X7R is more effective above 5 MHz than the less expensive Z5U type.

The analog Phase Locked Loop (PLL) of the IMR/IMR+ device is inherently more sensitive to noise in the 50–100 kHz range. The IMR chip recommended that a 47 μF capacitor be used in parallel with a 0.1 μF capacitor to provide filtering in this range for the PLL power pins. The effectiveness of this filter can be significantly improved by isolating DV_{SS} from AV_{SS} with a 1.8–2.2 Ω resistor. Some experimentation has also shown that a 120Z ferrite bead is effective in place of this resistor. The IMR+ chip incorporates additional decoupling capacitance internally. Comparative testing indicates that the necessity for the 47 μF capacitor can be removed if the board design is relatively quiet (typically less than 300 mV of power supply noise). However, there will be no adverse technical impact on the IMR+ device operation if the 47 μF capacitor is retained.

6.3 POWER PLANES

The board power planes must be separated into analog and digital portions. The +5V and ground planes can be laid out as shown in Figure 6-4. The analog portion should be located under the analog power pins of the IMR+ device, and the AUI logic. The digital portion should be located under all of the digital logic. The digital ground should be extended close enough to the 10BASE-T filters to attach a 0.1 μF capacitor to the filters ground pin. Extending the digital power plane under the 10BASE-T filters is not recommended. The analog and digital power planes should be tied at a single point with either a 1.8–2.2 Ω resistor or a 120Z ferrite bead. A 47 μF capacitor in parallel with a 0.1 μF capacitor is used for decoupling.

Figure 6-4 IMR/IMR+ Device Power Plane Recommendations



Shielded RJ-45 connectors are recommended. The shield pins should be tied to the frame ground. Depending on the characteristics of the 10BASE-T filters, either the frame ground or a void in the planes should be extended under the filters. Consult the filter manufacturer to determine if the frame ground is needed to minimize the effects of crosstalk within the filters.

6.4 FILTER MODULES

10BASE-T ports are by definition unshielded. Any noise on these ports will be emitted to the outside world. EMI studies of 10BASE-T designs show that special attention needs to be paid to common mode noise, since this is usually the greatest contributor to EMI problems. For this reason, it is recommended that 10BASE-T filter/transformer modules that incorporate a common mode choke in the transmit (and preferably receive) path are used. A common mode choke in the receive circuit is useful for keeping noise induced anywhere in the receive link from coupling to the transmit path.

For a list a representative filter/transformer modules, refer to Appendix A.

An additional method to further reduce any high frequency common mode noise effects is to add 68 pF capacitors at 2000V to the network side of the filter modules outputs. Calculations show that capacitors in this range do not violate the IEEE 802.3 requirements for insertion loss and return loss.

SECTION

7 GLOSSARY

AUI
Attachment Unit Interface. IEEE specification for a node or repeater connection interface to an external medium attachment unit (MAU). The AUI cable between the DTE/repeater and the MAU may be up to 50 m in length. In systems where the MAU is embedded into the DTE or repeater (such as 10BASE-T or 10BASE2) a physical implementation of the AUI may not be present. Defined in Section 7 of ISO/IEC 8802-3: 1990 (ANSI/IEEE Std 802.3). See also "PLS".

CI
Control In. AUI differential pair circuit, operating at pseudo ECL levels. The MAU drives a 10 MHz signal on the CI circuit to indicate to the DTE or repeater that a collision has been detected on the network, that the MAU is the the jabber state, or that an SQE Test from the MAU to the DTE is in progress. See also "Jabber" and "SQE Test".

CMIP
Common Management Information Protocol. The ISO defined transport protocol to move management information through the network. Defined by ISO/IEC 9595/6, Information Processing Systems – Open Systems Interconnection – Common Management Information Protocol Specification.

Concentrator
A general term frequently used instead of repeater. Typically, a concentrator supports more than one network protocol, such as 802.3/Ethernet as well as 802.5/Token Ring. The terms "hub", "concentrator" and "intelligent hub" are frequently used interchangeably to reference a multiport, multiprotocol device, capable of statistics gathering, fault monitoring and/or network management activities.

CRC
Cyclic Redundancy Check. A CRC is calculated by the MAC transmit process, and checked by the MAC receive process of a station (DTE), to ensure integrity of the frame contents. The mathematical CRC is computed on the Destination Address, Source Address, Length/Type and Data/Pad fields (all the frame except the Preamble, SFD and FCS fields). The computed 32-bit CRC is appended as the last 4-bytes of every frame. See also "FCS".

CSMA/CD
Carrier Sense Multiple Access/Collision Detect. The Media Access Control protocol used in Ethernet and 802.3 networks, which determines the transmit and receive characteristics of each station. Defined in Section 4 of ISO/IEC 8802-3:1990 (ANSI/IEEE Std 802.3). See also "MAC".

DA
Destination Address. The 48-bit field within the 802.3/Ethernet packet format which identifies the physical address of the intended recipient. The field immediately follows the Preamble/Start Frame Delimiter, and precedes the Destination Address (DA) field. The 802.3 protocol supports individual, multicast and broadcast addressing. See also "Source Address".

DI
Data In. AUI differential pair circuit, operating at pseudo ECL levels. Data received by the MAU from either the media or the DO circuit (or its logical equivalent), is driven onto the DI circuit for use by the DTE or repeater.

DO
Data Out. AUI differential pair circuit, operating at pseudo ECL levels. The DTE or repeater drives Manchester encoded data out on the DO circuit (or its logical equivalent), which is transmitted by the MAU over the physical media and the DI circuit.

DTE
Data Terminal Equipment. Communication station (or node) capable of reception and/or transmission of data. Generally includes the MAC and PLS sublayer functions, but may also include an embedded MAU.

ENDEC
Encoder/Decoder. The Manchester encoder/decoder, effectively the physical implementation of the 802.3 PLS sublayer. NRZ data output by the MAC, is passed to the PLS function and Manchester encoded, and sent over the AUI DO circuit (or its logical equivalent) to the MAU. Manchester encoded data from the network (from other MAC devices), received on the AUI DI circuit, is passed to the PLS function and decoded to extract clock and NRZ data, as well as the indication of receive carrier, which are returned to the MAC.

FOIRL
Fiber Optic Inter Repeater Link. IEEE specification for inter repeater communications link, primarily aimed at significantly increasing the distance capabilities of an 802.3/Ethernet network. Defined in Section 9.9 of ISO/IEC 8802-3:1990 (ANSI/IEEE Std 802.3).

FCS
Frame Check Sequence. A 4-byte field appended to the end of each 802.3/Ethernet frame. The field contains a 32-bit CRC, mathematically computed on the Destination Address, Source Address, Length/Type and Data/Pad fields (all the frame except the Preamble, SFD and FCS fields). The CRC is appended by the MAC transmit process and checked by the MAC receive process, and used to verify the integrity of the frame contents. See also "CRC".

HIMIB
Hardware Implemented Management Information Base. The HIMIB device (Am79C987) effectively acts as a "management co-processor" to the IMR+ device, providing support for all hardware intensive Repeater Management functions. The HIMIB device incorporates hardware registers which directly support the mandatory and optional attributes required to implement an IEEE 802.3 repeater MIB.

HMI
Hub Management Interface. A specification developed by Novell, Inc., which defines the MIB variables for a 10BASE-T repeater, as well as the driver independent protocol by which the variables are accessed in a Novell software environment.

Hub
A general term frequently used instead of repeater. See also "Concentrator" and "Repeater".

IAB
Internet Advisory Board. The group within the Internet community responsible for the administration of protocol standards.

IETF
Internet Engineering Task Force. The group within the Internet community responsible for the development of protocol standards.

ILACC
Integrated Local Area Communication Controller (Am79C900). A general purpose 32-bit bus mastering Ethernet controller, with integrated SIA function. Largely software compatible with the LANCE device. See also "LANCE".

IMR
Integrated Multiport Repeater (Am79C980). Single chip repeater incorporating 8 10BASE-T and 1 AUI compatible ports, all 802.3 repeater requirements, management, diagnostics and port expansion facilities.

IMR+
Enhanced Integrated Multiport Repeater (Am79C981). An enhanced pin, timing and software compatible version of the IMR device. The IMR+ chip incorporates additional features used by the HIMIB (Am79C987) device for managed 802.3 repeater applications.

IPG
Inter Packet Gap. The minimum time permitted between back-to-back packets on the 802.3 network, specified as 96 bits (9.6 µs) minimum. Note that a phenomenon known as IPG shrinkage can cause the IPG to be reduced below 96 bits.

Jabber
A MAU is required to interrupt a station which is transmitting for an excessive period of time, to prevent the network from disruption. The MAU is required to enter the jabber state if it detects continuous DO circuit (or its logical equivalent) activity for 20–150 ms. During the jabber state, transmission to the network is disabled and collision is returned to the DTE via the CI circuit (or its logical equivalent). The MAU will remain in this state until DO circuit activity stops for a period of 0.25–0.75 s.

Jam
A DTE that detects a collision, is required to complete the Preamble/SFD sequence (if it has not already done so), and subsequently continue to transmit for an additional 32 bit times. The Jam sequence can be any arbitrary pattern, providing it is not the calculated CRC. On a repeater, if a receive collision is detected, it will ensure that at least 96-bits are sent to all ports except the receiving (colliding) port, if necessary by appending an additional Jam sequence to the frame. When a repeater detects entry into the transmit collision state, a new 96-bit Jam sequence will be transmitted to all ports. In this case, the Jam sequence must start with the first bit as a one and continue with 62-bits of alternating "1, 0, 1, 0.....", with the remainder being an arbitrary pattern.

LANCE
Local Area Network Controller for Ethernet (Am7990). A general purpose, 24-bit address/16-bit data, bus mastering Ethernet controller. The Novell NE2100 architecture is based on the LANCE device. See also "PCnet-ISA".

Length
A 2-byte field in the 802.3 frame, immediately following the Source Address field, and preceding the Data field, which defines the total number of data bytes contained in the frame. The valid range is 46–1500 bytes. For values less than 46-bytes, Pad characters are added to the Data field of the transmit frame to ensure that the minimum size frame is observed (64-bytes); these are removed at the receiving station. The field is transmitted with the high order byte first, in LSB to MSB order. See also "Type".

MAC
Media Access Control. The MAC sublayer defines the medium independent capability for frame transmission and reception using the CSMA/CD access method. Defined in Section 4 of ISO/IEC 8802-3:1990 (ANSI/IEEE Std 802.3).

MACE
Media Access Controller for Ethernet (Am79C940). A general purpose, 16-bit data bus, slave Ethernet controller, incorporating a high speed bus interface, Ethernet controller, SIA, 10BASE-T transceiver and AUI port.

Manchester Encoding
Manchester encoded data is used for data transmission across the AUI (and across all of the common media currently defined for 802.3). Each bit of information is converted into a "bit-symbol", which in turn is divided into two halves. During the first half of the bit-symbol, the representation is the complement of the data bit being encoded, and during the second half of the bit-symbol, the representation is identical to the data bit value. In this way, a transition is guaranteed in the center of every bit-symbol, hence clock and data information are encoded into a single serial representation. Manchester encoding/decoding is performed in the PLS sublayer. See also "ENDEC" and "PLS".

MAU
Medium Attachment Unit. The physical and electrical interface between a DTE or repeater and the actual medium. The MAU is connected to the DTE by an AUI, although this may not be visible if the MAU is embedded within the DTE or repeater. A different MAU is required to support each different type of medium (cable type).

MIB
Management Information Base. Network devices which can be managed embed a MIB. The MIB is effectively an internally located table of variables which an external device can access. Several published MIB standards exist for various network devices. The work to create a MIB for 802.3 repeaters is defined in IEEE 802.3 Section 19, Layer Management for 10 Mb/s Baseband Repeaters. This information was also used as the basis of the definition for the "Repeater MIB" in RFC 1368, which defines the SNMP encoding for an 802.3/Ethernet repeater. In addition, many vendors publish their own MIBs and/or extensions to an existing MIB.

MIB-I
Management Information Base. Defines the core set of managed objects for the Internet suite of protocols. Primarily focussed on monitoring network interfaces and the protocol stack. Defined in RFC 1156. See also "MIB", "MIB-II" and "RMON MIB".

MIB-II
Management Information Base. Defines extended capabilities over MIB-I, primarily focussed on monitoring the network interfaces and the protocol stack (including SNMP). Defined in RFC 1213. See also "MIB", "MIB-I" and "RMON MIB".

PCnet-ISA
Single Chip Ethernet Controller for ISA (Am79C960). An ISA based, 24-bit address/ 16-bit data, bus mastering Ethernet controller. Basically a single chip implementation of the Novell NE2100 adapter card, incorporating a full ISA bus interface, Ethernet controller, SIA, 10BASE-T transceiver and AUI port. Driver compatible with the Novell NE2100 architecture, and based on the LANCE software model. See also "LANCE" and "SIA".

PLS
Physical Signalling. IEEE specification which defines the signalling scheme between the MAC sublayer, through to the AUI, which provides the interface for a medium attachment unit (MAU). Defined in Section 7 of ISO/IEC 8802-3: 1990 (ANSI/IEEE Std 802.3). The PLS sublayer is physically implemented by the Manchester encoder/decoder function. NRZ data output by the MAC, is passed to the PLS function and Manchester encoded, and sent over the AUI DO circuit (or its logical equivalent) to the MAU. Manchester encoded data from the network (from other MAC devices), received on the AUI DI circuit (or its logical equivalent), is passed to the PLS function and decoded to extract clock and NRZ data, as well as the indication of receive carrier and collision, which are returned to the MAC sublayer. See also "AUI" and "ENDEC".

PMA
Physical Medium Attachment. The portion of the MAU which contains the active circuitry responsible for the interface of the AUI circuits to the specific network medium.

Preamble
An alternating "1, 0, 1, 0....." sequence at the start of each frame transmission. The preamble for 802.3 networks is defined as 7-bytes long, followed by a 1-byte SFD; whereas the preamble for Ethernet networks is defined as 62-bits long, with a 2-bit "Synch" character (hence both have an overall preamble/start delimiter length of 64-bits). When Manchester encoded, the preamble sequence produces a 5 MHz frequency at the start of each frame, which is used by a receiving station or repeater as a reference to decode the receive clock and data.

Repeater
An 802.3/Ethernet repeater in its most generic form is an "n" port device, which supports the 802.3 protocol only. A repeater is used to extend the physical topology of the network, allowing two or more cable segments to be coupled together. No more than four repeaters are permitted between the end-to-end path of two stations. When data is received on a single port, the repeater retransmits the incoming bit stream to all other ports, performing signal retiming and amplitude restoration. When data appears simultaneously on more than one port, the repeater transmits a collision to all ports, including the receiving ports. In addition, the repealer can isolate a port if it detects faults, such as excessive number or duration of collisions, to prevent disruption of the rest of the network. In a 10BASE-T network, the repeater provides a central point of connectivity, ideally suited to the incorporation of statistics gathering and network administration functions. Defined in Section 9 of ISO/IEC 8802-3:1990 (ANSI/IEEE Std 802.3).

Repeater Management
The generic term used to describe the 802.3 Supplements "Layer Management for 10 Mb/s Baseband Repeaters" (Section 19). The Repeater Management Standard defines the MIB variables for an 802.3 repeater. Defined in IEEE Std 802.3k–1992 (Supplement to ISO/IEC 8803-3:1992 [ANSI/IEEE Std 802.3, 1992 Edition]). See also "Repeater MIB".

Repeater MIB
The generic term used to describe RFC 1368, which defines the MIB variables and their encoding for use by SNMP. See also "Repeater Management".

RFC
Request For Comment. A specification administered and developed by the Internet community (IAB/IETF). The specifications are public domain, and detail the TCP/IP Internet protocol suite.

RMON MIB
Remote Network Monitoring Management Information Base. Defines the MIB attributes for a network monitoring device, with detailed statistic, fault diagnostic, performance monitoring and historical trending capabilities.Defined in RFC 1271. See also "MIB-I" and "MIB-II".

SA
Source Address. The 48-bit field within the 802.3/Ethernet packet format which identifies the senders unique physical address. The field immediately follows the Destination Address (DA) field. See also "Destination Address".

SFD
Start of Frame Delimiter. The SFD immediately follows the alternating "1, 0, 1, 0....." preamble sequence. The SFD for 802.3 networks is defined as a 1-byte field consisting of the pattern "1, 0, 1, 0, 1, 0, 1, 1" whereas for Ethernet networks a 2-bit pattern of "1, 1" at the end of preamble signifies the "Synch" character (although the byte containing the SFD/Synch character is identical in both cases). The first but of the 802.3 frame (the Destination Address field) immediately follows the SFD. See also "Preamble" and "Synch".

SNMP
Simple Network Management Protocol. The Internet specified transport protocol for the movement of management data in a heterogeneous network environment. Defined in RFC 1157.

SIA
Serial Interface Adaptor (Am7992). A Manchester Encoder/Decoder IC, which performs the Physical Signalling (PLS) sublayer functions of the IEEE 802.3 Standard. The device encodes data and clock from the MAC for transmission over the network, and drives the DO circuit of the AUI. It receives data from the network via the DI circuit of the AUI, extracts the data and clock into separate paths, and passes these back to the MAC.

SQE
Signal Quality Error. A 10Mb/s pulse train passed from the MAU (using the CI circuit) to a DTE or repeater to indicate an error condition on the network, such as collision or excessive transmit duration (jabber).

SQE Test
Signal Quality Error Test. Frequently referred to as "Heartbeat". An 802.3 MAU, when connected to a station (DTE), is required to send a brief collision indication back to the DTE after the end of each transmission. When the transmission from the DTE completes on the DO circuit, within 0.6–1.6 μs the MAU should start to return a collision indication to the DTE using the CI circuit, of duration 10 ± 5 bit times. The test is intended to notify the DTE that the MAU collision circuitry is operational, and that the AUI cable is intact. Note that the SQE Test function must be disabled when a MAU is connected to a repeater.

Synch
Synchronization. The "Synch" immediately follows the alternating "1, 0, 1, 0....." preamble sequence in an Ethernet frame, and indicates that the first bit of the frame (the Destination Address field) will follow immediately. The Synch for Ethernet networks is a 2-bit pattern of "1, 1" at the conclusion of the preamble. For 802.3 networks, a 1-byte SFD field is defined, consisting of the pattern "1, 0, 1, 0, 1, 0, 1, 1" (hence the byte containing the Synch/SFD character is identical in both cases). See also "Preamble" and "SFD".

TCP/IP
Transmission Control Protocol/Internet Protocol. The Internet protocol suite, which defines a wide range of network services, to allow heterogeneous network system operation. The suite is a layered set of protocols, and covers all aspects of network service and communication. Primarily defined in RFC 791 and RFC 793, although many other related RFCs are applicable to the protocol suite.

TPEX
Twisted Pair Ethernet Transceiver (Am79C98 or Am79C100). A transceiver IC that converts the electrical signals of the AUI to those of the 10BASE-T Standard.

Type
A 2-byte field in the Ethernet frame, immediately following the Source Address field, and preceding the Data field, which defines the protocol type of the frame. The Type specification has no meaning at the MAC level, and is passed to higher level protocols. In the 802.3 frame, this field defines the length of the data portion of the frame. Note that Type identifiers fall outside the range for valid packet Length values. The field is transmitted with the high order byte first, in LSB to MSB order. See also "Length".

10BASE-FB
10 Mb/s Baseband Fiber Optic Backbone. Covered by Section 17 (Draft) of IEEE 802.3. Uses 802.3 protocol, dual fiber point-to-point cabling with synchronous signalling to provide an inter-repeater "backbone" link. No defined maximum node count, maximum fiber distance 2 km, depending on system configuration.

10BASE-FL
10 Mb/s Baseband Fiber Optic Link. Covered by Section 18 (Draft) of IEEE 802.3. Uses 802.3 protocol, dual fiber point-to-point cabling and repeaters to provide the network architecture. No defined maximum node count, maximum fiber distance 1–2 km, depending on system configuration.

10BASE-FP
10 Mb/s Baseband Fiber Optic Passive. Covered by Section 16 (Draft) of IEEE 802.3. Uses 802.3 protocol, dual fiber point-to-point cabling and passive optical star to provide the network architecture. No defined maximum node count, maximum fiber distance 0.5 km, depending on system configuration.

10BASE-T
10 Mb/s Baseband Twisted Pair. Covered by Section 14 of IEEE 802.3. Uses 802.3 protocol, point-to point twisted pair cabling and repeaters to provide network services. No defined maximum node count, maximum cable distance 100 m. Defined in Section 13 and 14 of IEEE Std 802.3i–1990 (Supplement to ISO/IEC 8802-3: 1990 (ANSI/IEEE Std 802.3)).

10BASE2
10 Mb/s Baseband 200 m (Cheapernet). A low cost version of 10BASE5 (frequently referred to as Cheapernet), eliminates the external AUI requirement, relaxes the network electrical interfaces and allows use of thin 75 Ω coaxial cable. Maximum 30 nodes (or mating connectors) on cable segment, 185 m per segment. Defined in Section 10 of ISO/IEC 8802-3: 1990 (ANSI/IEEE Std 802.3).

10BASE5
10 Mb/s Baseband 500 m (Ethernet). Based on the original Ethernet specification proposed by DEC, Intel and Xerox, for multi-drop communication scheme using the CSMA/CD access protocol, over thick 75Ω coaxial cable. 802.3 is the corresponding IEEE standard which varies in minor electrical and protocol specifications. Maximum 100 nodes on cable segment, 500 m per segment. Defined in Section 8 of ISO/IEC 8802-3: 1990 (ANSI/IEEE Std 802.3).

APPENDIX

A 10BASE-T INTERFACE FILTER AND TRANSFORMER MODULES

The table below lists the recommended resistor values and filter and transformer modules for the IMR+ device.

IMR+ Device Compatible 10BASE-T Media Interface Modules

Manufacturer	Part #	Package	Description
Bel Fuse	A556-2006-DE	16-pin 0.3" DIL	Transmit and receive filters and transformers.
Bel Fuse	0556-2006-00	14-pin SIP	Transmit and receive filters and transformers.
Bel Fuse	0556-2006-01	14-pin SIP	Transmit and receive filters, transformers and common mode chokes.
Bel Fuse	0556-6392-00	16-pin 0.5" DIL	Transmit and receive filters, transformers and common mode chokes.
Halo Electronics	FD02-101G	16-pin 0.3" DIL	Transmit and receive filters and transformers.
Halo Electronics	FD12-101G	16-pin 0.3" DIL	Transmit and receive filters and transformers, transmit common mode choke.
Halo Electronics	FD22-101G	16-pin 0.3" DIL	Transmit and receive filters, transformers and common mode chokes.
Halo Electronics	FD22-101R2	16-pin 0.3" DIL	Transmit and receive filters, transformers, common mode chokes, plus termination resistors.
PCA Electronics	EPA1990A	16-pin 0.3" DIL	Transmit and receive filters and transformers.
PCA Electronics	EPA2013D	16-pin 0.3" DIL	Transmit and receive filters and transformers, transmit common mode choke.
PCA Electronics	EPA2116	16-pin 0.3" DIL	Transmit and receive filters and transformers, transmit common mode choke, plus termination resistors.
Pulse Engineering	PE-65434	10-pin SIP	Transmit and receive filters, transformers, and common mode choke.
Pulse Engineering	PE-65445	16-pin 0.3" DIL	Transmit and receive filters and transformers (for SMT use PE-65446)
Pulse Engineering	PE-65467	16-pin 0.3" DIL	Transmit and receive filters, transformers, common mode chokes, plus termination resistors.
Pulse Engineering	PE-65424	16-pin 0.3" DIL	Transmit and receive filters, transformers, and common mode chokes.
TDK	TLA 470	14-pin SIP	Transmit and receive filters and transformers.
Valor Electronics	PT3877	16-pin 0.3" DIL	Transmit and receive filters and transformers.
Valor Electronics	PT3983	8-pin 0.3" DIL	Transmit and receive common mode chokes.
Valor Electronics	FL1012	16-pin 0.3" DIL	Transmit and receive filters and transformers, transmit common mode choke.
Valor Electronics	FL1010/17	62-pin	4 channels (Quad Mod). Transmit and receive filters, transformers, common mode chokes, plus termination resistors.
Valor Electronics	FL1020	16-pin 0.3" DIL	Transmit and receive filters, transformers and common mode chokes, plus termination resistors.
Valor Electronics	FL1003	18-pin SIP	Transmit and receive filters, transformers and common mode chokes.

APPENDIX

B AUI ISOLATION TRANSFORMERS

IMR/IMR+ Device Compatible AUI Isolation Transformers

Manufacturer	Part #	Package	Description
Bel Fuse	A553-0506-AB	16-pin 0.3" DIL	50 µH
Halo Electronics	TD01-0756K	16-pin 0.3" DIL	75 µH
Halo Electronics	TG01-0756W	16-pin SMD	75 µH
PCA Electronics	EP9531-4	16-pin 0.3" DIL	50 µH
Pulse Engineering	PE64106	16-pin 0.3" DIL	50 µH
TDK	TLA 100-3E	16-pin 0.3" DIL	100 µH
Valor Electronics	LT6031	16-pin 0.3" DIL	50 µH

Contact the following companies for further information on their products:

Bel Fuse	Phone:	(317) 831-4226
	FAX:	(317) 831-4547
Halo Electronics	Phone:	(415) 989-7313
	FAX:	(415) 367-7158
PCA Electronics	Phone:	(408) 954-0400
	FAX:	(408) 954-1440
Pulse Engineering	Phone:	(619) 674-8218
	FAX:	(619) 675-8262
Valor Electronics	Phone:	(619) 458-1471
	FAX:	(619) 458-0875
TDK	Phone:	(213) 530-9397
	FAX:	(213) 530-8127

Sales Offices

North American

ALABAMA ..... (205) 882-9122
ARIZONA ..... (602) 242-4400
CALIFORNIA,
- Culver City ..... (310) 645-1524
- Newport Beach ..... (714) 752-6262
- Sacramento(Roseville) ..... (916) 786-6700
- San Diego ..... (619) 560-7030
- San Jose ..... (408) 452-0500
- Woodland Hills ..... (818) 992-4155

CANADA, Ontario,
- Kanata ..... (613) 592-0060
- Willowdale ..... (416) 222-7800

COLORADO ..... (303) 741-2900
CONNECTICUT ..... (203) 264-7800
FLORIDA,
- Clearwater ..... (813) 530-9971
- Ft. Lauderdale ..... (407) 361-0050
- Orlando (Longwood) ..... (407) 862-9292

GEORGIA ..... (404) 449-7920
IDAHO ..... (208) 377-0393
ILLINOIS,
- Chicago (Itasca) ..... (708) 773-4422
- Naperville ..... (708) 505-9517

MARYLAND ..... (301) 381-3790
MASSACHUSETTS ..... (617) 273-3970
MINNESOTA ..... (612) 938-0001
NEW JERSEY,
- Cherry Hill ..... (609) 662-2900
- Parsippany ..... (201) 299-0002

NEW YORK,
- Brewster ..... (914) 279-8323
- Rochester ..... (716) 425-8050

NORTH CAROLINA
- Charlotte ..... (704) 875-3091
- Raleigh ..... (919) 878-8111

OHIO,
- Columbus (Westerville) ..... (614) 891-6455
- Dayton ..... (513) 439-0268

OREGON ..... (503) 245-0080
PENNSYLVANIA ..... (215) 398-8006
TEXAS,
- Austin ..... (512) 346-7830
- Dallas ..... (214) 934-9099
- Houston ..... (713) 376-8084

International

BELGIUM, Antwerpen ..... TEL ..... (03) 248 43 00
FAX ..... (03) 248 46 42
FRANCE, Paris ..... TEL ..... (1) 49-75-10-10
FAX ..... (1) 49-75-10-13
GERMANY,
- Bad Homburg ..... TEL ..... (06172)-24061
- FAX ..... (06172)-23195
- München ..... TEL ..... (089) 45053-0
- FAX ..... (089) 406490

HONG KONG, Wanchai ..... TEL ..... (852) 865-4525
FAX ..... (852) 865-4335
ITALY, Milano ..... TEL ..... (02) 3390541
FAX ..... (02) 38103458
JAPAN,
- Atsugi ..... TEL ..... (0462) 29-8460
- FAX ..... (0462) 29-8458
- Kanagawa ..... TEL ..... (0462) 47-2911
- FAX ..... (0462) 47-1729
- Tokyo ..... TEL ..... (03) 3346-7550
- FAX ..... (03) 3342-5196
- Osaka ..... TEL ..... (06) 243-3250
- FAX ..... (06) 243-3253

KOREA, Seoul ..... TEL ..... (82) 2-784-0030
FAX ..... (82) 2-784-8014

International (Continued)

LATIN AMERICA,
- Ft. Lauderdale ..... TEL ..... (305) 484-8600
- FAX ..... (305) 485-9736

SINGAPORE ..... TEL ..... (65) 3481188
FAX ..... (65) 3480161
SWEDEN,
- Stockholm area (Bromma) ..... TEL ..... (08) 98 61 80
- FAX ..... (08) 98 09 06

TAIWAN, Taipei ..... TEL ..... (886) 2-7153536
FAX ..... (886) 2-7122183
UNITED KINGDOM,
- Manchester area (Warrington) ..... TEL ..... (0925) 830380
- FAX ..... (0925) 830204
- London area (Woking) ..... TEL ..... (0483) 740440
- FAX ..... (0483) 756196

North American Representatives

CANADA
Burnaby, B.C. – DAVETEK MARKETING ..... (604) 430-3680
Kanata, Ontario – VITEL ELECTRONICS ..... (613) 592-0060
Mississauga, Ontario – VITEL ELECTRONICS ..... (416) 564-9720
Lachine, Quebec – VITEL ELECTRONICS ..... (514) 636-5951
ILLINOIS
- Skokie – INDUSTRIAL REPRESENTATIVES,INC ..... (708) 967-8430

INDIANA
- Huntington – ELECTRONIC MARKETING CONSULTANTS, INC ..... (317) 921-3450
- Indianapolis – ELECTRONIC MARKETING CONSULTANTS, INC ..... (317) 921-3450

IOWA
- LORENZ SALES ..... (319) 377-4666

KANSAS
- Merriam – LORENZ SALES ..... (913) 469-1312
- Wichita – LORENZ SALES ..... (316) 721-0500

KENTUCKY
- ELECTRONIC MARKETING CONSULTANTS, INC ..... (317) 921-3452

MICHIGAN
- Holland – COM-TEK SALES, INC ..... (616) 335-8418
- Brighton – COM-TEK SALES, INC ..... (313) 227-0007

MINNESOTA
- Mel Foster Tech. Sales, Inc. ..... (612) 941-9790

MISSOURI
- LORENZ SALES ..... (314) 997-4558

NEBRASKA
- LORENZ SALES ..... (402) 475-4660

NEW MEXICO
- THORSON DESERT STATES ..... (505) 883-4343

NEW YORK
- East Syracuse – NYCOM, INC ..... (315) 437-8343
- Hauppauge – COMPONENT CONSULTANTS, INC ..... (516) 273-5050

OHIO
- Centerville – DOLFUSS ROOT & CO ..... (513) 433-6776
- Columbus – DOLFUSS ROOT & CO ..... (614) 885-4844
- Westlake – DOLFUSS ROOT & CO ..... (216) 899-9370

OREGON
- ELECTRA TECHNICAL SALES, INC ..... (503) 643-5074

PENNSYLVANIA
- RUSSELL F. CLARK CO.,INC. ..... (412) 242-9500

PUERTO RICO
- COMP REP ASSOC, INC ..... (809) 746-6550

UTAH
- Front Range Marketing ..... (801) 288-2500

WASHINGTON
- ELECTRA TECHNICAL SALES ..... (206) 821-7442

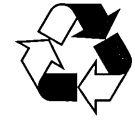
WISCONSIN
- Brookfield – INDUSTRIAL REPRESENTATIVES,INC ..... (414) 789-9393

Advanced Micro Devices, Inc. 901 Thompson Place, P.O. Box 3453, Sunnyvale, CA 94088, USA
Tel: (408) 732-2400 • TWX: 910-339-9280 • TELEX: 34-6306 • TOLL FREE: (800) 538-8450
APPLICATIONS HOTLINE & LITERATURE ORDERING • TOLL FREE: (800) 222-9323 • (408) 749-5703

17314B 2/3/93
BAN-9.7M-3/93-0 Printed in USA

ADVANCED
MICRO
DEVICES, INC.
901 Thompson Place
P.O. Box 3453
Sunnyvale,
California 94088-3453
(408) 732-2400
TWX: 910-339-9280
TELEX: 34-6306

APPLICATIONS HOTLINE &
LITERATURE ORDERING
USA (800) 222-9323
USA (408) 749-5703
JAPAN 011-81-3-3346-7561
UK & EUROPE 44-(0)256-811101
TOLL FREE
USA (800) 538-8450
FRANCE 0590-8621
GERMANY 0130-813875
ITALY 1678-77224



RECYCLED &
RECYCLABLE

Printed in USA
BAN-9.7M-3/93-0
17314B

ADVANCED
MICRO
DEVICES, INC.
901 Thompson Place
P.O. Box 3453
Sunnyvale,
California 94088-3453
(408) 732-2400
TWX: 910-339-9280
TELEX: 34-6306

APPLICATIONS HOTLINE &
LITERATURE ORDERING
USA (800) 222-9323
USA (408) 749-5703
JAPAN 011-81-3-3346-7561
UK & EUROPE 44-(0)256-811101
TOLL FREE
USA (800) 538-8450
FRANCE 0590-8621
GERMANY 0130-813875
ITALY 1678-77224

RECYCLED &
RECYCLABLE

Printed in USA
BAN-9.7M-3/93-0
17314B