An Introduction to the ControlNet Network

This chapter introduces the ControlNet network. It contains these sections:

Section	Page
The ControlNet Network Mission	1
The ControlNet Network as a Part of an Allen-Bradley Architecture	2
Object Modeling	3
The ControlNet Communication Model	4
ControlNet Product Overview	7
ControlNet Enablers	9
ControlNet Media	10

The ControlNet Network Mission

Control and I/O data is information critical to the operation of a process.

The ControlNet network's mission is to provide reliable, high-speed transport of two basic types of application information:

- **control** and **I/O** data
- non-time critical messaging data related to the controlled system

Control and I/O data are given highest priority. Other information, such as programming or parameter upload and download operations, does not interfere with the transport of control and I/O data.

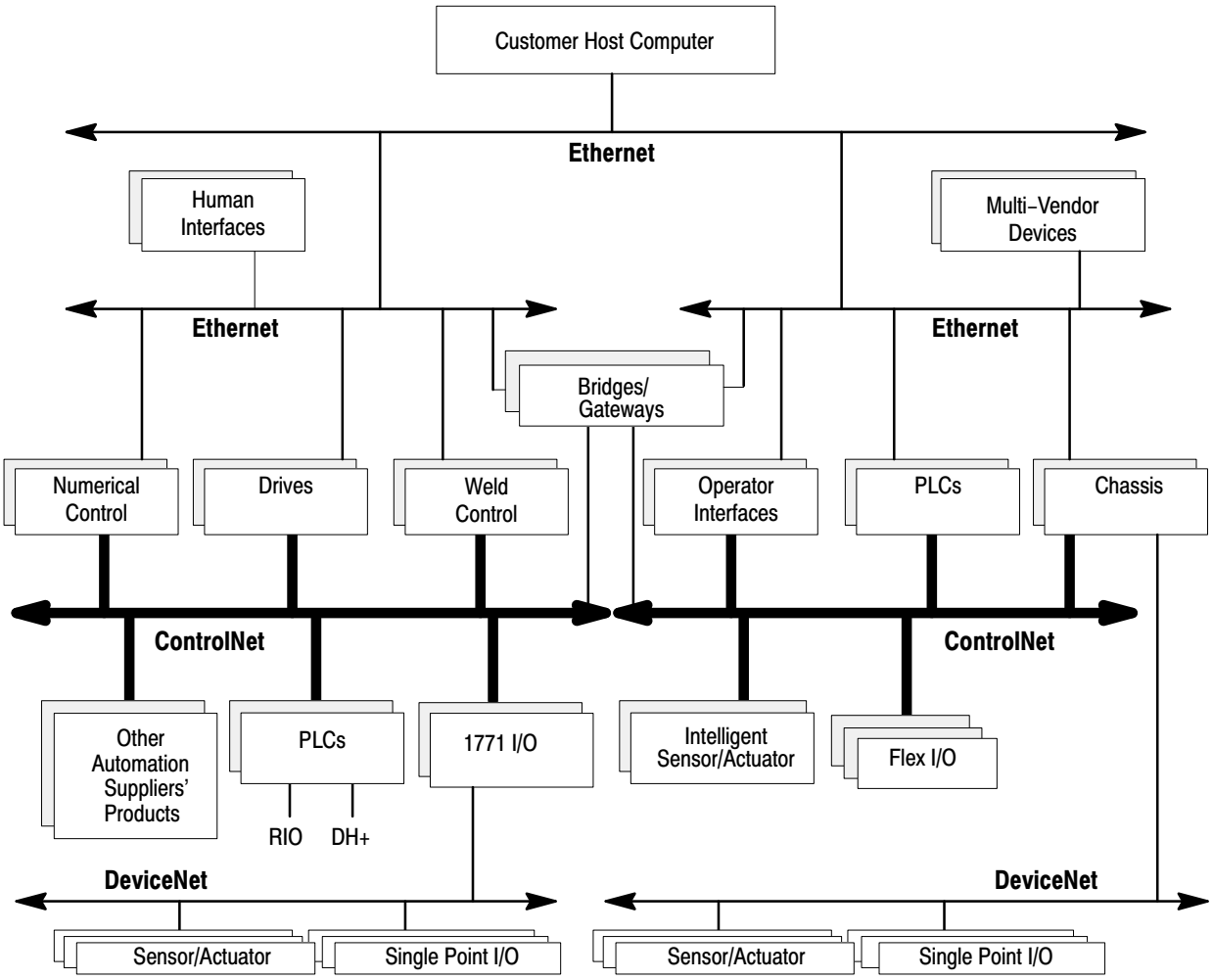
The ControlNet network:

- provides a high-speed control and I/O network
- facilitates deterministic data transfer over the network
- supports transparent media redundancy (optional)
- combines the functions of RIO and DH+™ networks into a single LAN
- supports easy configuration, maintenance, and troubleshooting

The ControlNet Network as a part of an Allen-Bradley Architecture

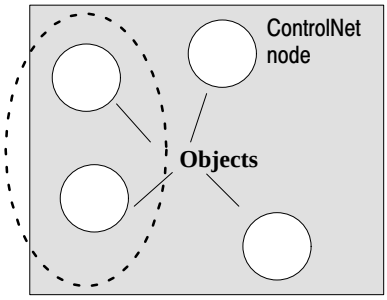
The ControlNet network is designed to be the central communication architecture for interoperating automation products. Figure 1.1 represents the ControlNet network's broad environment.

Figure 1.1
ControlNet-based Architecture



Object Modeling

The ControlNet network is designed through object modeling. Object modeling organizes related data and procedures into one entity: the *object*.



A *Class* of Objects

An object is a collection of related *services* and *attributes*. Services are procedures an object performs. Attributes are characteristics of objects represented by values, which can vary. Typically, attributes provide status information or govern the operation of an object. Attributes often represent the state of an object. The value associated with an attribute may or may not affect the *behavior* of an object. An object's behavior is an indication of how the object responds to particular events.

Classes categorize objects. A class defines a particular type of object and defines the characteristics shared by all the objects within a class. For example, the human race could be represented by the class "Human" with millions of objects within it.

Objects within a class are called *object instances*. An object instance is the actual representation of a particular object within a class. Each instance of a class has the same set of attributes, but has its own set of *attribute values*, which makes each instance in the class unique.

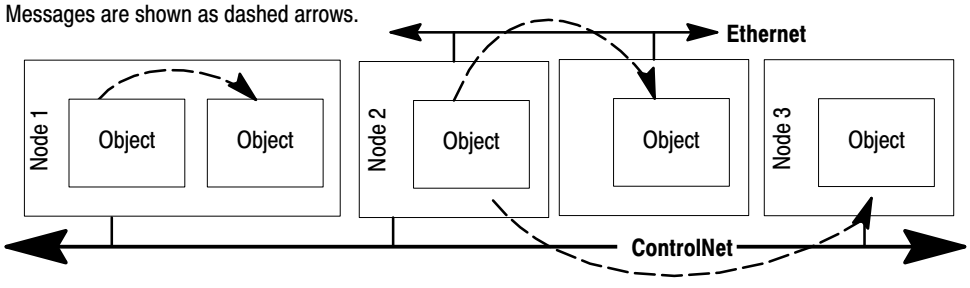
Each person in the human race can be represented by an instance within the class "Human." All humans have the same set of attributes: eyes, ears, age, gender, etc. Yet, because the values of each attribute vary, each of us looks different and performs in distinct ways.

Class	Instances	Attributes	Attribute Values
Human	Christy	Sex	Female
		Age	31
	Charles	Sex	Male
		Age	50

A message is a signal sent to an object to request a service.

Application objects communicate with each other by sending **messages**. As shown in Figure 1.2, messages can be sent between objects within a single node and across networks.

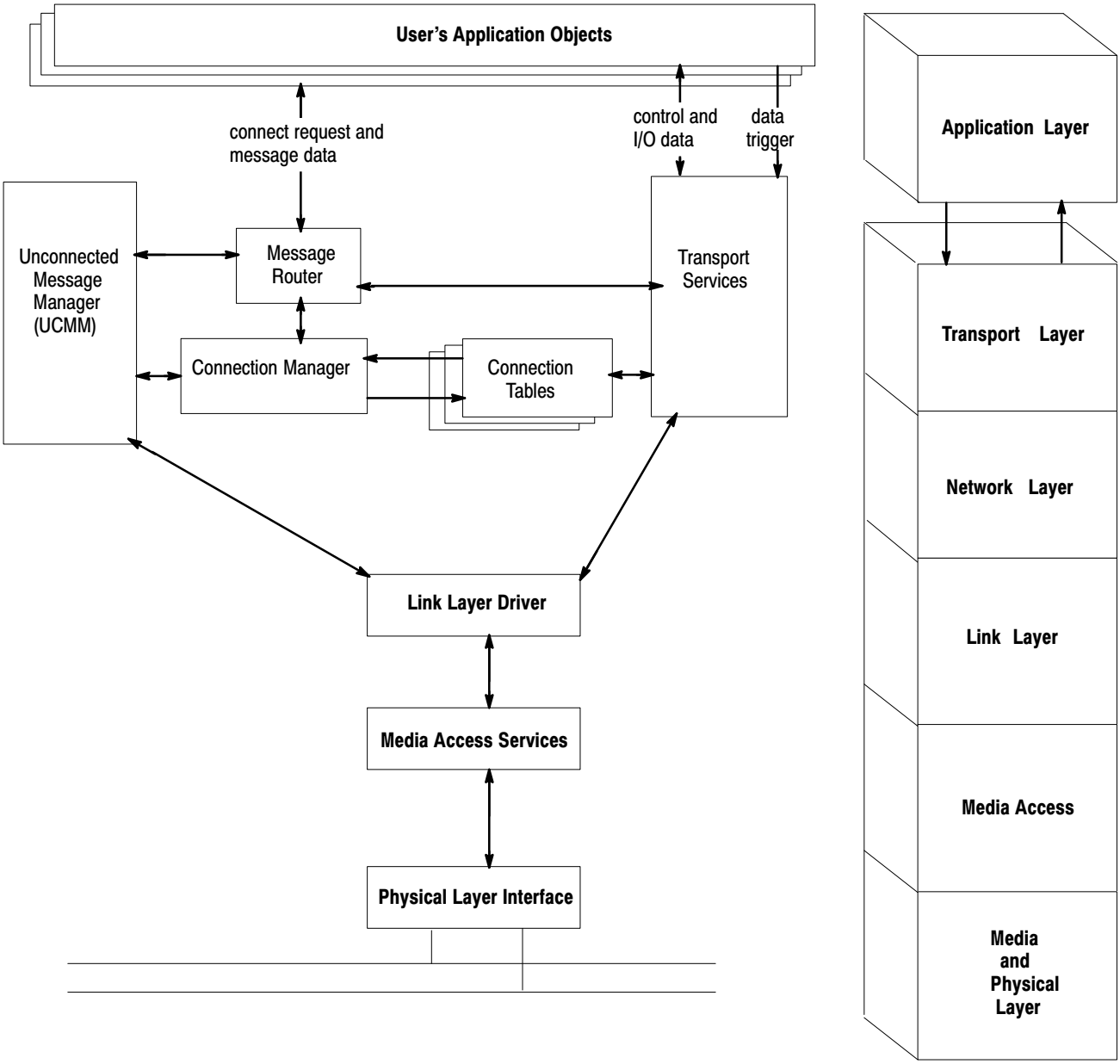
Figure 1.2
Application Object Message Paths



The ControlNet Communication Model

General relationships between the functions that comprise the ControlNet communication model are illustrated in Figure 1.3. All of the figure's blocks below the *User's Application Objects* reside within either a ControlNet ASIC or ControlNet class-specific Example Code.

Figure 1.3
ControlNet Communication Model



Communication Model Components

This section provides a bottom-up definition of each of the ControlNet Communication Model's components illustrated in Figure 1.3, on page 4.

Physical Layer Interface

The **Physical Layer Interface** consists of the hardware components which are necessary to get on and off the network's physical media. It is responsible for:

- providing transceiver components for the redundant coax media ports
- providing transceiver components for the Network Access Port (NAP)

Media Access Service

Media Access Service allows an application to transmit on the network's media. This service is contained within the ControlNet ASICs. It is responsible for:

- providing transmit and receive data flow-control on and off the media
- checking received data for errors
- adding the proper header and trailer information on transmit frames
- deleting header and trailer information from received frames
- controlling the media redundancy switch-over algorithm

Link Layer Driver

The **Link Layer Driver** assembles data into correctly-formatted ControlNet frames and is responsible for:

- setting up the Media Access Service
- moving data to and from the Media Access Service
- controlling the Media Access Service's transmit and receive data flow
- holding received data in temporary storage until checked for errors by the Media Access Service

Unconnected Message Manager (UCMM)

The **Unconnected Message Manager (UCMM)** provides the ability to send a message *without* an established connection. It is responsible for:

- receiving incoming UCMM messages
- sending and receiving unconnected messages to and from UCMM objects on other nodes
- sending and receiving unconnected messages to and from its local Message Router

The **Message Router (MR)** allows an application to open connections to objects within the same node and is responsible for:

- interpreting the part of a message that indicates the local destination object
- routing the message to the appropriate object for execution

The **Connection Manager (CM)** opens and closes connections as well as maintains the network's connection tables. It is responsible for:

- setting up a connection within its node
- processing all connection requests sent on the local service
- forwarding an open connection service via the UCMM to the next node in a connection path
- establishing a connection to a target node

Connection Tables hold information about all connections in which the node participates.

Transport Services define the type of data delivery an application object requires. They are responsible for:

- notifying the transmitting application of the data's arrival
- detecting duplicate data delivery

Important: Once a connection is established, objects such as the MR, UCMM and CM are no longer required. Data can go directly to the destination object.

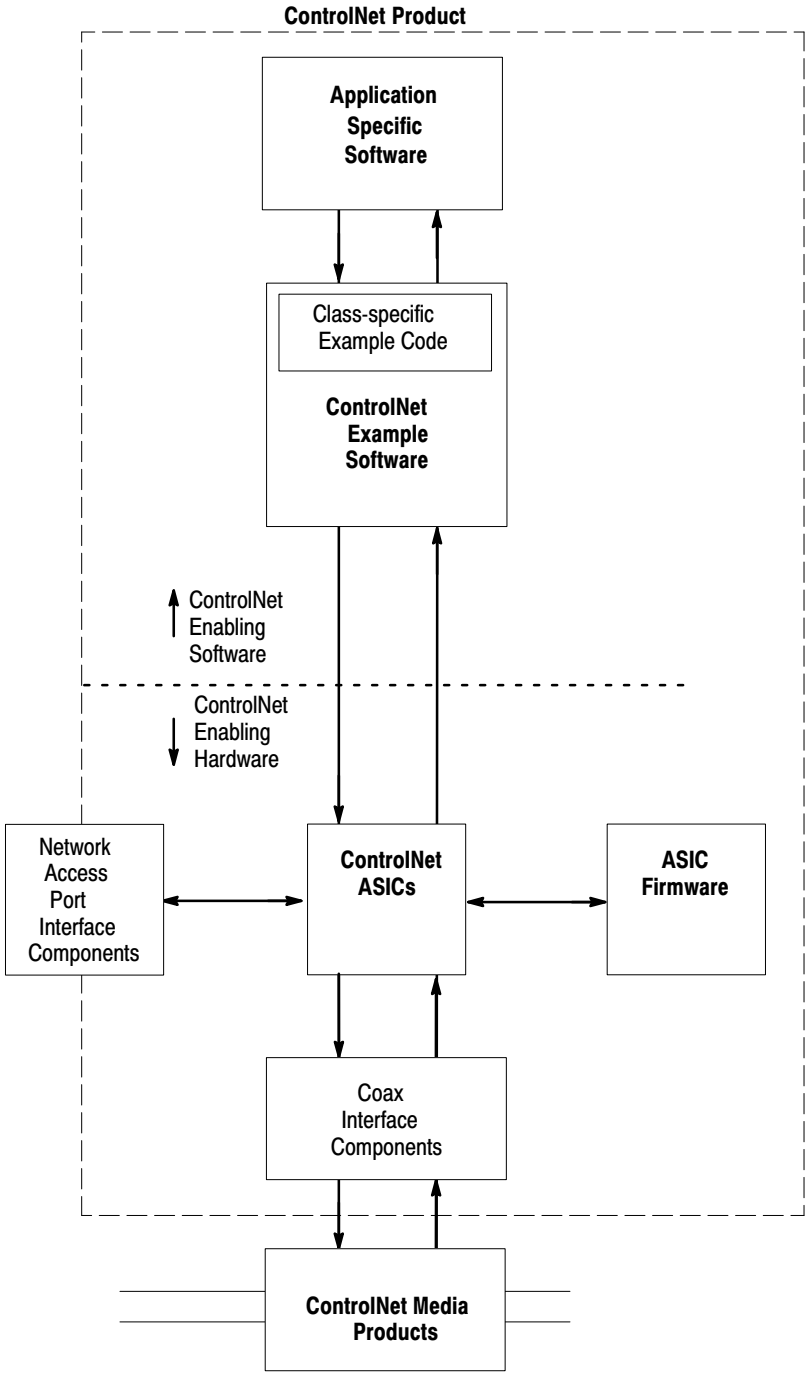
The **User's Application Objects** are the functions for which data is transmitted or received. They are responsible for:

- making a connection request to establish a path that allows the exchange of information between application objects
- selecting the connection type that defines *how many* application objects can use the data
- deciding the connection priority which defines the time-critical *nature of the data* to be sent
- defining the trigger mode, which determines *when* new data should be sent
- choosing the transport service which defines the *quality* of delivery required
- generating the data that has been requested by other applications

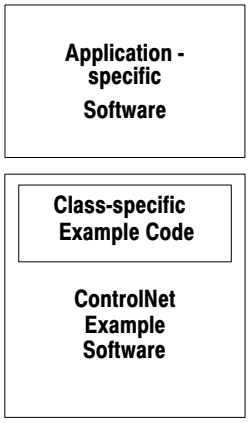
ControlNet Product Overview

A ControlNet product is made up of several components. In general, a product developer is responsible for supplying the host micro-processor and application specific software that resides above the ControlNet enabling hardware and software. The general components required to implement ControlNet-based communication within a product are illustrated below:

.

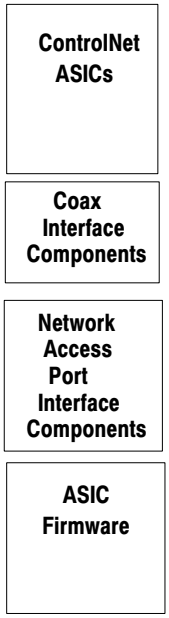


ControlNet Product Components



ControlNet Enabling Software provides access to enabling hardware as well as defines a product's behavior.

- **Application-specific Software**, provided by the product designer, resides above the ControlNet enabling components; it provides the added value of the customer's specific product or application.
- **ControlNet Example Software** is an example implementation that demonstrates the required protocol for the network. It includes:
- opening/closing connections
- maintaining connections
- directing data transmission and reception
- supporting specific, product-based requirements
- supporting interfaces to several product classes:
  - Message Class (CNA20)
  - Adapter Class (CNA30)
  - Scanner Class (CNA40)



ControlNet Enabling Hardware implement lower levels of ControlNet's protocol and provide physical connections to the network. These components include: ASICs, transceivers, transformers, crystals, and connectors.

- A **ControlNet ASIC** is an **A**pplication **S**pecific **I**ntegrated **C**ircuits developed for the ControlNet network.
- **ControlNet Coax Interface Components** allow a ControlNet ASIC to access the network's physical signalling.
- **Network Access Port** (**NAP**) **Interface Components** include an RJ45 connector and transceiver that provide a tapless, full-speed, temporary network connection at every node.
- **ASIC Firmware** provides a product class-specific personality to the ControlNet node. Much of the computationally intensive parts of the ControlNet protocol are done here, offloading the host CPU.

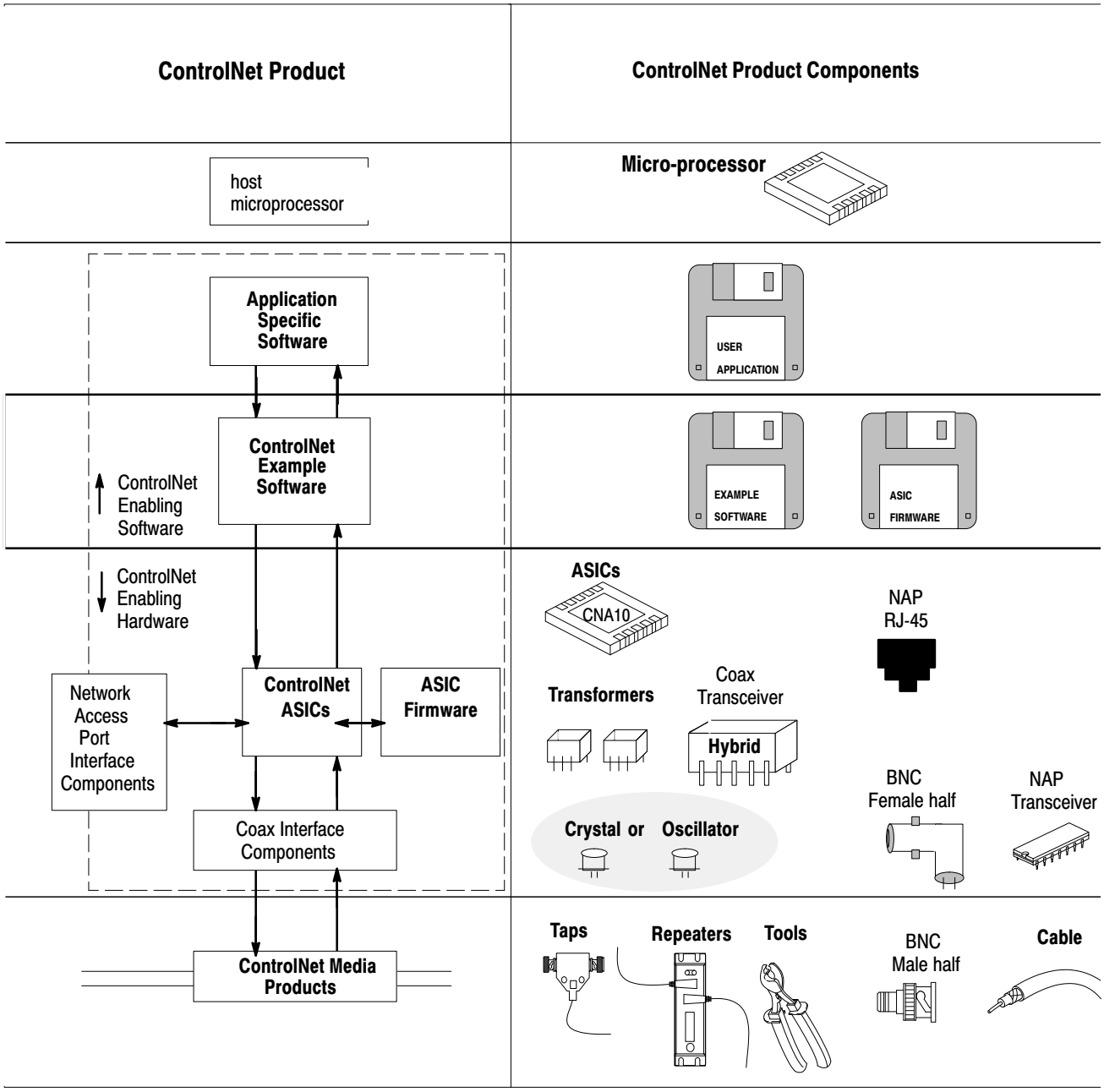
ControlNet Media Products are items that compose the network's physical environment such as the cable and taps. (See Chapter 6, "ControlNet Physical Media.").

ControlNet Enablers

ControlNet enablers are the software and hardware components necessary for a product to function on the ControlNet network. Figure 1.4 illustrates these components.

Important: All application-specific software resides in a user-provided host micro-processor, not on ControlNet ASICs.

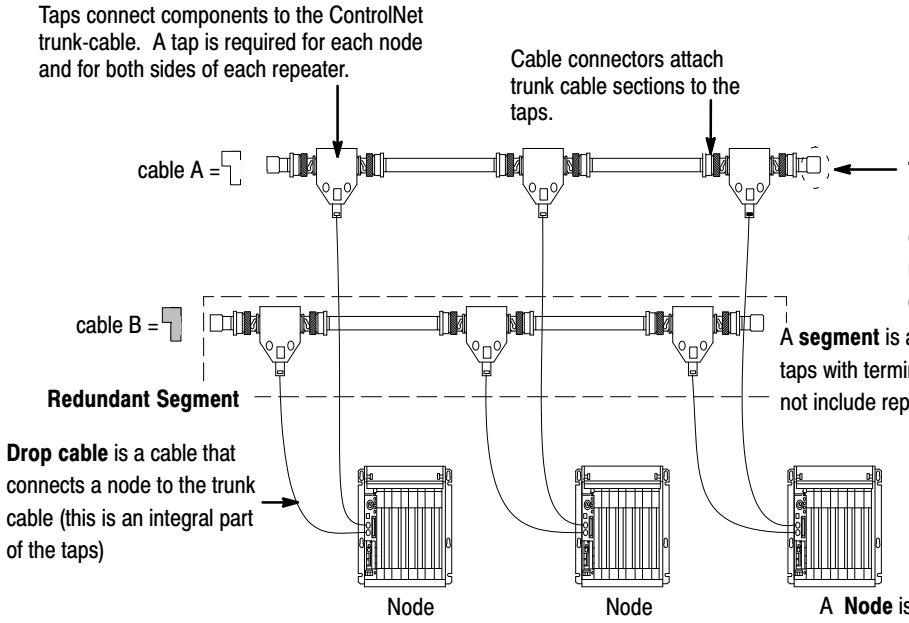
Figure 1.4
Components that Enable Communication on the ControlNet Network



ControlNet Media

ControlNet media consists of the trunk-cable, connectors, taps, and repeaters through which data travels. In contrast to the logical components such as software or protocols, the media are the physical components that make up a ControlNet network.

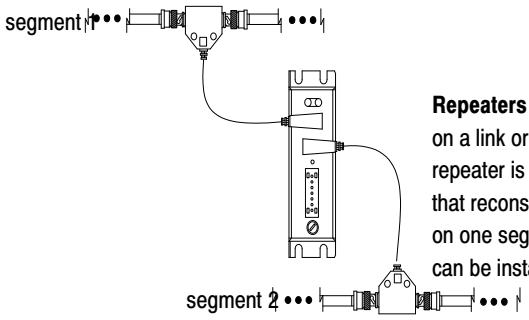
A **link** is a collection of nodes with unique addresses (in the range of 1-99). Segments connected by repeaters make up a link; links connected by bridges make up a network.



Terminators are 75-ohm resistors (mounted in BNC plugs) placed on the ends of segments to prevent reflections from occurring at the ends of cables.

A **segment** is a trunk-cable section connected via taps with terminators at each end; a segment does not include repeaters.

A **Node** is the port of a physical device connecting to the network which requires a network address in order to function on the network; a link may contain a maximum of 99 nodes.



Repeaters increase the number of taps permitted on a link or extend the total length of a link. A repeater is a two-port active physical-layer device that reconstructs and retransmits all traffic it hears on one segment to another segment. A repeater can be installed on *any* tap.

AkntIkICNt Aknœdoll tekn

This chapter discusses how the ControlNet network is configured and maintained. This chapter contains these sections:

Section	Page
Network And Station Management Introduction	1
ControlNet Configuration Manager Node	2

Network And Station Management Introduction

The ControlNet configuration method prevents any single node from disrupting the network's normal operations. This method includes a common set of operating rules for all nodes:

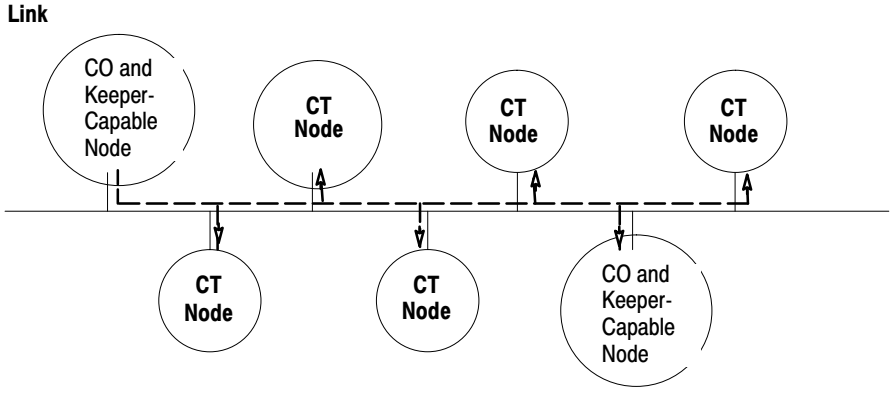
- all nodes on a link must agree on a common set of network configuration parameters before transmitting **scheduled** or **unscheduled** data
- each connection originator (CO) is required to have specific scheduling information before requesting or transmitting scheduled data
- each connection target (CT) can transmit unscheduled data immediately after power–up and a request from a connected originator

ControlNet Configuration Keeper Node

The ControlNet Configuration Keeper node is a device that holds the network parameters and scheduling information in its non-volatile memory for distribution to other nodes.

The **ControlNet Configuration Keeper node** is responsible for maintaining and distributing the network and scheduling information. The Keeper node must hold the network attributes and scheduling information in non-volatile storage, allowing re-distribution after a power-fail event. Every ControlNet link must have a Keeper-capable node to:

- support scheduled traffic
- save and distribute the network parameters and scheduling information



A network may contain more than one Keeper.

Aonuroll eu Nhwtldcl Ceelc

This chapter provides a summary of ControlNet physical media. Also, it provides related specifications and parameters on a general level. This chapter contains these sections:

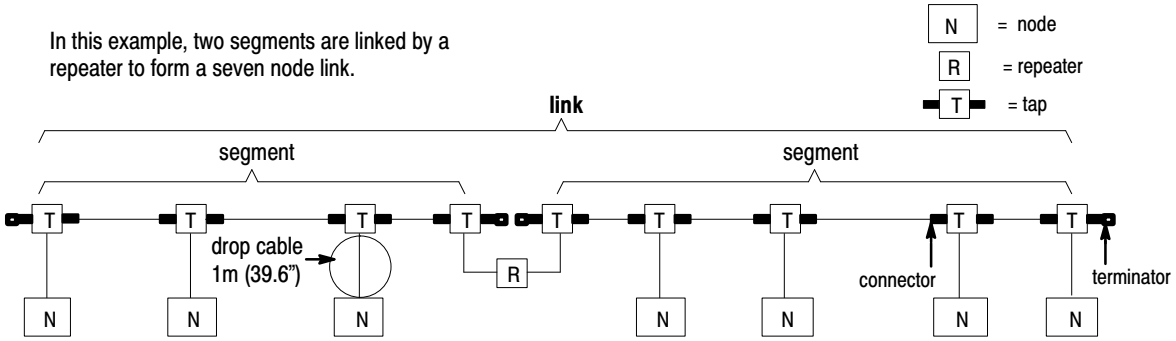
Section	Page
Overview of Physical Media	1
Network Specifications And Parameters	2
Tap Types	3
Physical Layer Repeaters	3
Redundant Media	4
Alternate Network Connection	4

Overview of Physical Media

The primary physical media for the ControlNet network is coaxial cable. A ControlNet physical network is comprised of this cable and a combination of connecting, transceiving, and terminating devices. Secondary is a fiber optic repeater device and cabling which is available to support long distances, such as building-to-building installation, as well as intrinsically safe environments.
The following sections are a general description of the primary physical system, its limitations, and components.

Figure 1.1 illustrates the basic components of ControlNet's physical media.

Figure 1.1
General Components of a ControlNet Cable System



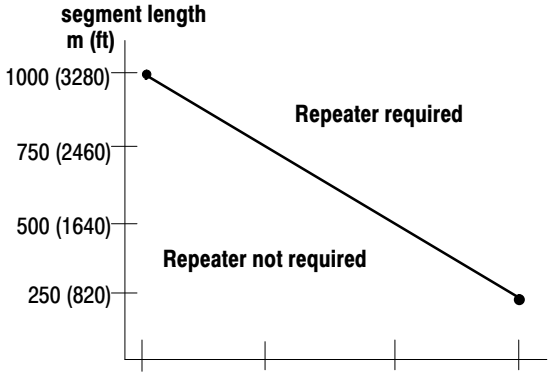
Network Specifications And Parameters

A tap connects a node to the network's trunk cable.

A segment is trunk-cable sections connected via taps with terminators at each end; a segment does not include repeaters.

Figure 1.2 graphs the network's physical parameters. It also illustrates the relationship between the length of a cable **segment** and the number of **taps** allowable within that distance, indicating when **repeaters** are necessary.

Figure 1.2
ControlNet Cable System Requirements



maximum allowable segment length =
1000m – 16.3m x [*number of taps* – 2]
or
3280ft – 53.4ft x [*number of taps* – 2]

Important: Networks that stay within the limits detailed in Table 1.A may connect additional nodes at all available NAPs. (See "Alternate Network Connection," on page 4.)

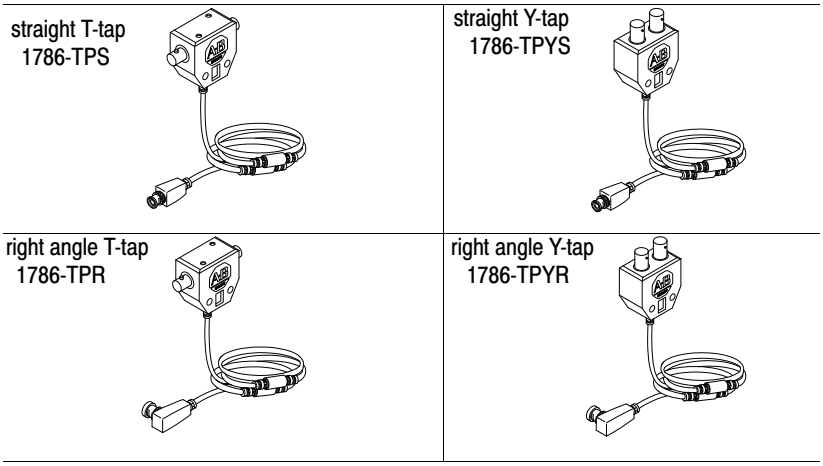
Table 1.A
ControlNet Network Parameters

Parameter	Limit
Maximum number of nodes per link	99 nodes
Maximum number of repeaters in a series (link)	depends on amount of cable used
Data rate	5.0 MBit/Sec (1.6 μ sec/Byte)
Undetected error rate	< 1 error in 20 years

Tap Types

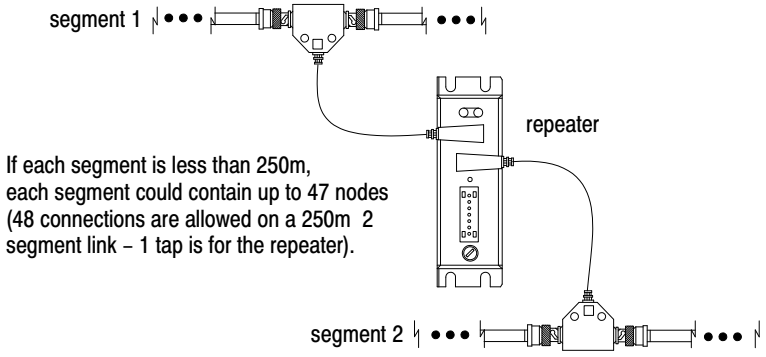
The ControlNet network uses a passive tap with four different configurations. The taps' names are derived from the unique combination of their connecting end configurations.
The following end configurations and their resulting combinations are shown in Figure 1.3.

Figure 1.3
Tap Configurations



Physical Layer Repeaters

Physical layer repeaters extend the length and/or node count of the network when a system requires more than 48 taps per link or a longer cable than specifications allow (reference Figure 1.2). Also, repeaters do not occupy any of the 99 network addresses allowed per link; they do not require an address. (See Figure 1.2, ControlNet Cable System Limitations and Requirements.)



Redundant Media

A node is the port of a physical device connecting to the network which requires a network address in order to function on the network.

A link is a collection of nodes with unique addresses (1-99). Segments connected by repeaters make up a link; links connected by bridges make up a network.

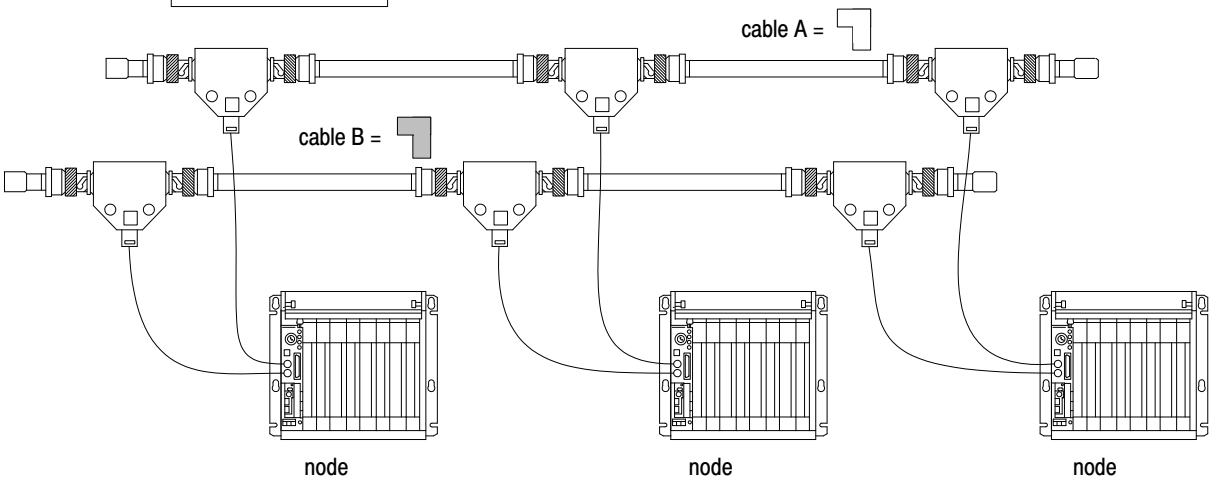
The ControlNet network provides the option of installing a second cable between **nodes**. With redundant media, nodes send identical signals on two separate segments. The receiving node continuously compares the signals' quality and selects the better cable to receive the next message. This also provides a backup cable in case one is improperly installed, not connected, or fails.

Important: Redundant media is not "Hot-Backup."

Actual ControlNet products are labeled with these icons  .

Cables on a redundant cable **link** are defined by the segment number and the redundant cable letter.

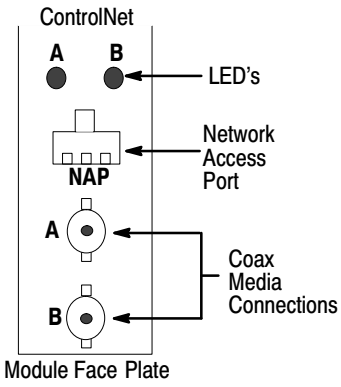
Figure 1.4
Redundant Media



Alternate Network Connection

All ControlNet nodes, except repeaters, are equipped with a Network Access Port (NAP). This feature provides a tapless, full-speed, temporary network connection at every node. Each connection uses a 10-foot modular cable with connectors to access the system's coaxial media. This cable can be extended to a length of 10m.

Important: This cable is available as catalog number 1786-CP.



AkntIkeCct Ak onNtdkn

This chapter presents the fundamentals of ControlNet connections. It contains these sections:

Section	Page
ControlNet Network Packet Format	1
Connections	4
The Producer/Consumer Model	7

ControlNet Network Packet Format

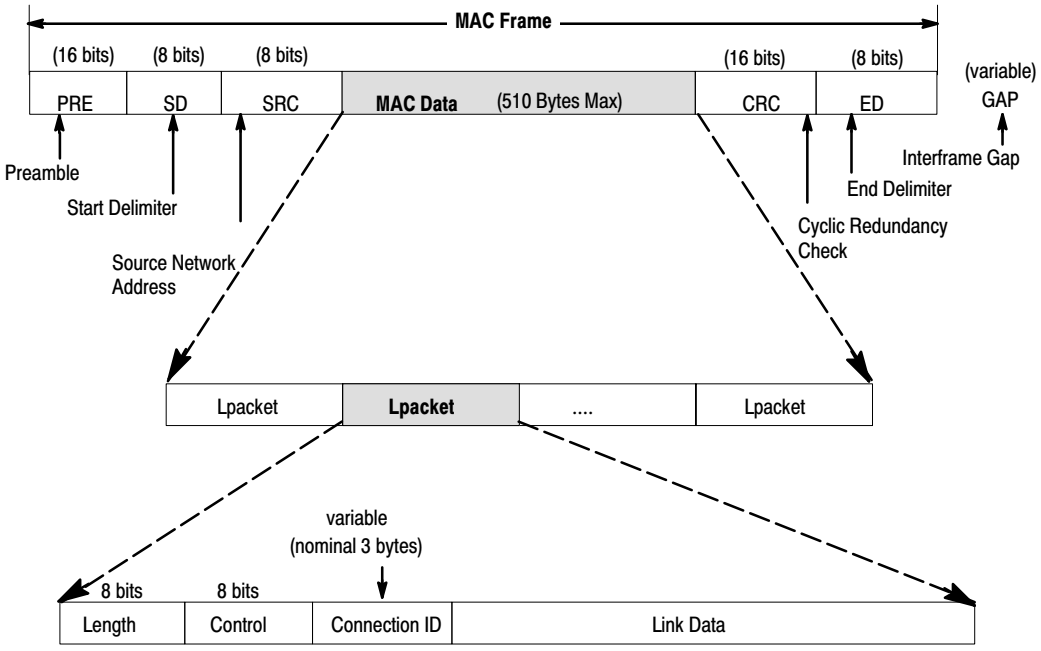
When a node sends data over the ControlNet network, it packages that data into a **MAC frame**. Each MAC frame can contain multiple **LPackets** which are transmitted together. (See Figure 1.1.)

Important: Each node can send only one MAC frame of up to a maximum of 510 bytes during each transmission opportunity.

Media Access Control (MAC) frames are the form in which a node transmits a group of data. This group of data is in the form of LPackets. A single MAC frame can contain several LPackets.

A **link packet**, or **LPacket** is data that has been packaged and labeled by a node in preparation for transmission.

Figure 1.1
MAC Frame and Lpacket Formats



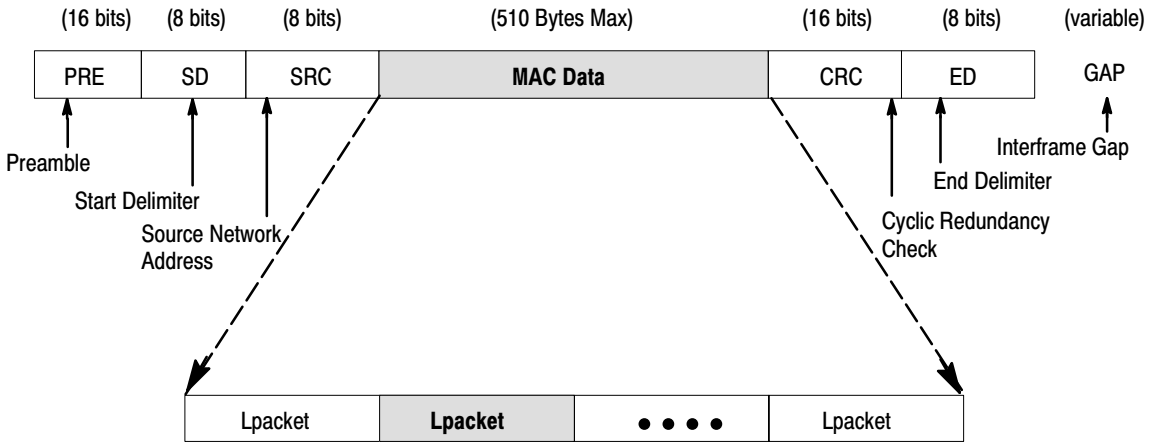
MAC Frame Format

The following items are inserted into MAC frames before transmission on the ControlNet network:

- preamble
- start delimiter
- source network address
- CRC
- end delimiter

The network address is written to a register in the ASIC. Once this is done, the system needs to provide only the contents of the MAC data. The MAC frame allows several pieces of information, called Lpackets, to be sent in a single transmission. Each Lpacket within the MAC data field could be destined for different consumers. (See Figure 1.2.)

Figure 1.2
MAC Frame Containing Multiple Lpackets

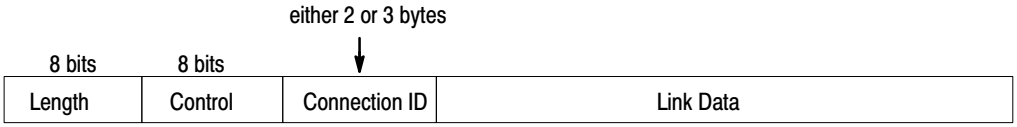


Lpacket Format

A **connection ID (CID)** is a numeric identifier on a network. It names the information to follow.

As shown in Figure 1.2, the MAC data field can contain several Lpackets, which are the packets of interest to a user's application. Each Lpacket contains a single piece of application information destined for one or more nodes on the network. Packets sent or received on the physical wire contain data of interest to the application layers as well as a **connection ID** (**CID**), control byte, and length byte. (See Figure 1.3.) An application uses the CID to determine whether or not a particular Lpacket contains the data it needs.

Figure 1.3
LPacket Format

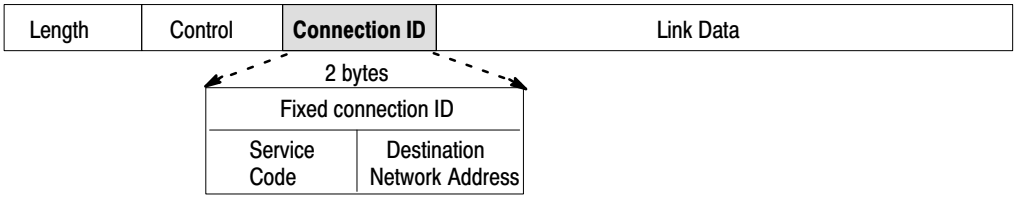


The CID is an identifier or name automatically created by ControlNet nodes. A CID is a shorthand method for referring to a particular object message. In the figure above, the CID can be one of two formats:

- fixed connection ID (2 bytes)
- general connection ID (3 bytes)

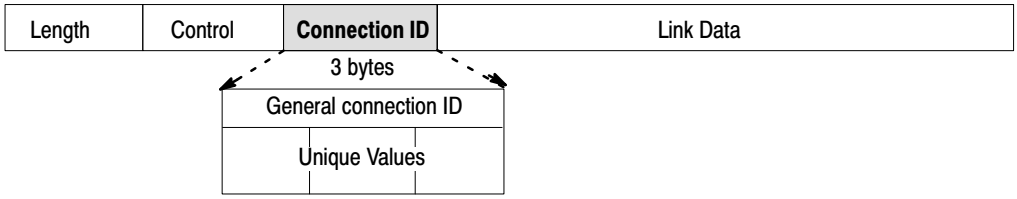
A **Fixed Connection ID** is two bytes long and contains a *service code* and *destination network address byte.* (See Figure 1.4.) The code byte is used to indicate the service required, while the destination byte indicates to which network address it is to be delivered.

Figure 1.4
Fixed CID Format



A **General Connection ID** is three bytes long and contains a number used to identify a specific data packet. (See Figure 1.5.) A general CID must be a unique value on any ControlNet network.

Figure 1.5
General CID Format



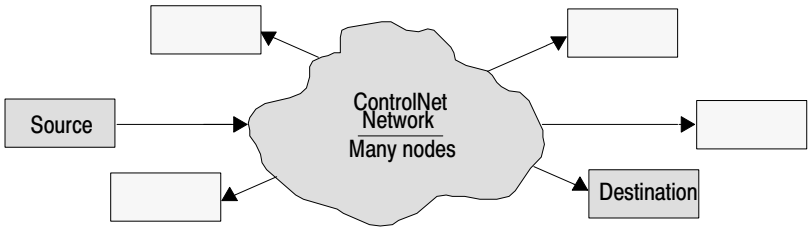
Connections

In some ways, a ControlNet connection can be compared to a telephone circuit. When you place a call, the telephone system selects a path for your call and sets up each switching station in the route to handle it. As long as the call continues, the resulting *virtual circuit* remains open, carrying data or voice traffic. In the telephone system, a single call can traverse multiple and different-type links. Through all of this, the connection appears the same to both the caller and the party called: sound in one end becomes sound in the other.

A connection also provides a path between two end points. Once the connection manager determines the virtual circuit, the route between end points is fixed. (See “Connection Manager,” on page 7.)
The end points of connections are *applications* that need to share data. Figure 1.6 illustrates a virtual circuit that traverses one or more intermediate nodes between the source and destination.

Important: The terms *source* and *destination* imply that a connection has been established and currently exists.

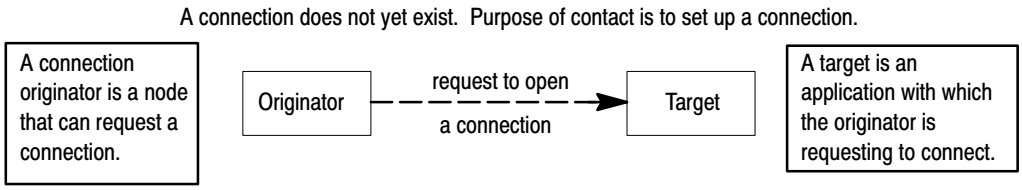
Figure 1.6
Virtual Circuit



Establishing a Connection

Every node contains the following:

- Unconnected Message Manager (UCMM)
- Message Router (MR)
- Connection Manager (CM)
- ControlNet Object
- Identity Object (ID)



In addition, a connection originator node contains a Connection Scheduler (CS) Object

The following sections about establishing a connection provide:

- a brief functional description on each connection object listed above
- a graphic as well as written explanation of how these components interact while a connection is initiated and established

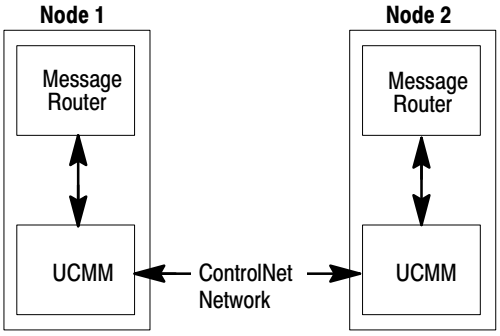
Unconnected Message Manager (UCMM)

The UCMM facilitates the exchange of information used to establish, open, or close a connection between applications. In addition, it is used to convey non-repetitive, non-time critical data on a single link. To establish a connection, the Connection Manager provides the UCMM with the network address and path to the target node. Once the connection is established, the address and path are no longer required. Opening the connection establishes a CID, which will be used to exchange application information.

As shown in Figure 1.7, each message the UCMM receives is forwarded to the Message Router where it is analyzed and sent to its specified function or object. The UCMM keeps a transaction record of each message received so that a reply can be sent to the proper location. Open and Close connection request messages are always sent via the UCMM. In addition, the UCMM provides:

- duplicate detection
- automatic retries
- message time-out

Figure 1.7
UCMM and Message Router Interaction

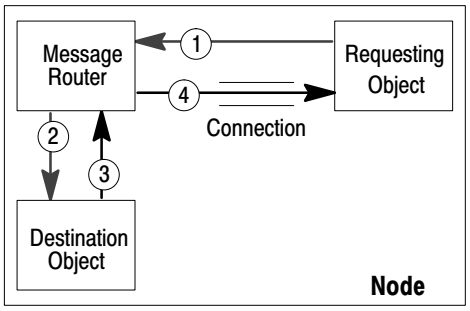


Important: UCMM messages are always sent in the unscheduled portion of the network interval.

Message Router

The Message Router (MR) allows an application to open connections to multiple objects within the same node. It acts much like an internal switchboard for a node's objects. Other nodes may establish a connection with the MR through the UCMM and connection manager. Figure 1.8 is an example of how the MR functions.

Figure 1.8
Message Router Functions



1. The MR determines which object is to perform the specified service by interpreting the identifying portion of the message.
2. The message is forwarded to the destination object.
3. A response from the destination object for the requesting object is received.
4. The MR forwards the response to the requesting object via an established connection.

Important: Connections can be created without a MR connection; a messaging connection to the MR is only mandatory when the originating application requires access to multiple internal objects through the same connection.

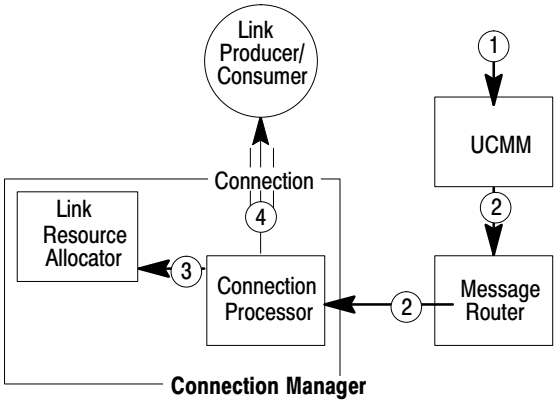
Connection Manager

The Connection Manager (CM) allocates internal resources necessary for each connection. Connection requests are originated by:

- other nodes via the UCMM
- an application on a node

Figure 1.9 is an example of how the the CM of a target node functions upon receiving a connection request by an originating node.

Figure 1.9
Object Components of the Connection Manager



1. The originator's UCMM contacts the target's UCMM with a connection request.
2. The request is routed through the target's MR to the CM.
3. The CM allocates the needed resources.
4. A connection is made to the originator node.

The Producer/Consumer Model

The ControlNet network uses the *Producer/Consumer* model to exchange application information. This model is the basis for understanding all ControlNet transactions. The following builds a producer/consumer model by discussing its basic components:

- object messages
- connection ID (CID)
- producer/consumer connection types
- transport services
- transport connection types

Object Messages

An **object message** is a piece of information that interests one or more nodes on the network.

In the ControlNet network's Producer/Consumer model, **object messages** are used to exchange information. An object message is a piece of information that interests one or more nodes on the network. It carries a value or set of values with a description of the value's meaning. The ControlNet network transfers object messages between producers and consumers to convey information.

The network view of messages, shown below, is simplified to facilitate minimal processing, high performance, and small code-requirements. It contains only that part of an object's value that changes with time.

A **producer** is a transmitting node.

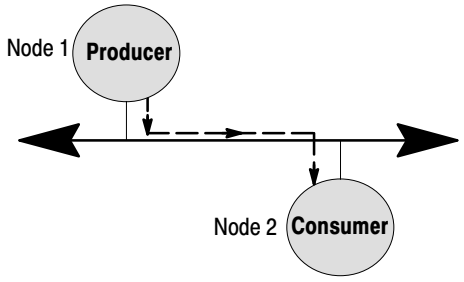
A **consumer** is a receiving node.

Nodes watch for certain CIDs transmitted by **producer** nodes. Once a node recognizes a CID, it *consumes* the message, therefore becoming a **consumer**. The network assumes that each object message has exactly one meaning but may have one or more consumers.

Producer/Consumer Connection Types

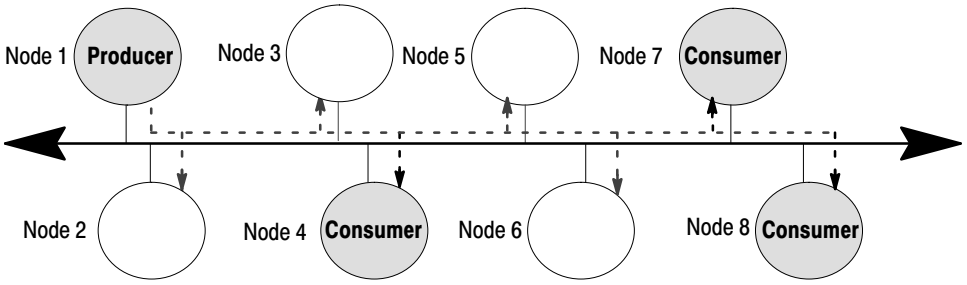
The concept of one producer and one consumer, illustrated in Figure 1.10, is known as a *point-to-point* connection.

Figure 1.10
Point-to-Point Producer/Consumer Connection



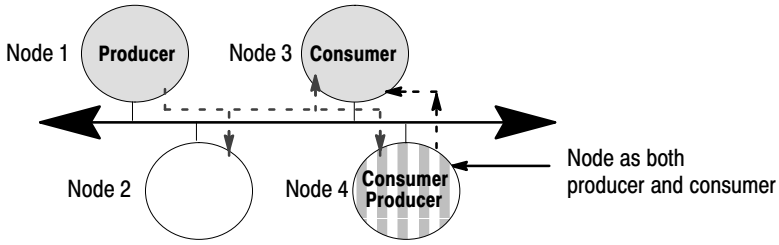
The concept of more than one consumer per message, or *multicast*, is shown in Figure 1.11. The dashed arrow represents an object message. In Figure 1.11, Node 1 *produces* a message on the network. Nodes 4, 7, and 8 *consume* the message. Nodes 2, 3, 5, and 6 see the message but are not interested in it and do not consume that message.

Figure 1.11
Multicast Producer/Consumer Connection



Important: Nodes can be producers, consumers, or both. For example, in Figure 1.12, Node 4 may want to send an object message to Node 3 based on the information it just consumed from Node 1.

Figure 1.12
A Single Node as Both a Producer and Consumer



Transport Services

Table 1.A lists the two general-purpose transport classes that have been defined for the ControlNet network. Each transport class provides a different level of service. It is these services that allow communication between applications. Transport classes with higher numbers incorporate and build upon the functions of lower transport classes. The originating application must determine which transport class best suits its needs for a particular data transfer.

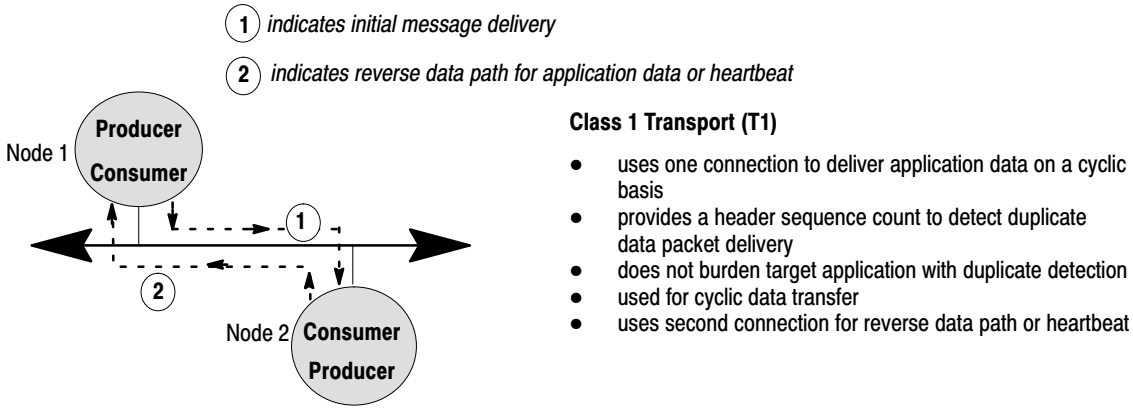
Table 1.A
Transport Classes

Class Number	Class Name
1	Duplicate detection
3	Verified

Transport Class 1 (T1)

Transport class 1 is illustrated in Figure 1.13. Class 1 provides the minimum level of service only, duplicate data detection.

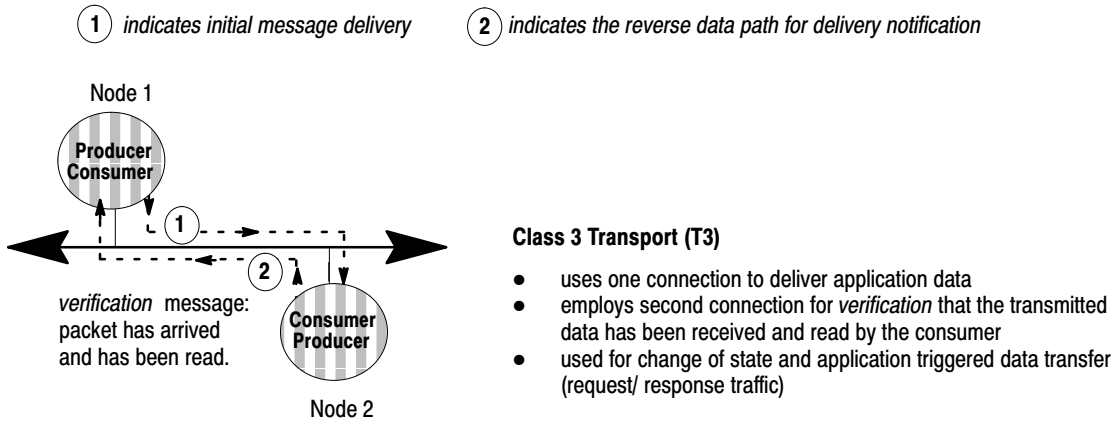
Figure 1.13
Transport Class 1



Transport Class 3 (T3)

Figure 1.14 illustrates transport classes 3. Class 3 provides data verification.

Figure 1.14
Transport Class 3



Transport Connection Types

The ControlNet network supports two types of connections:

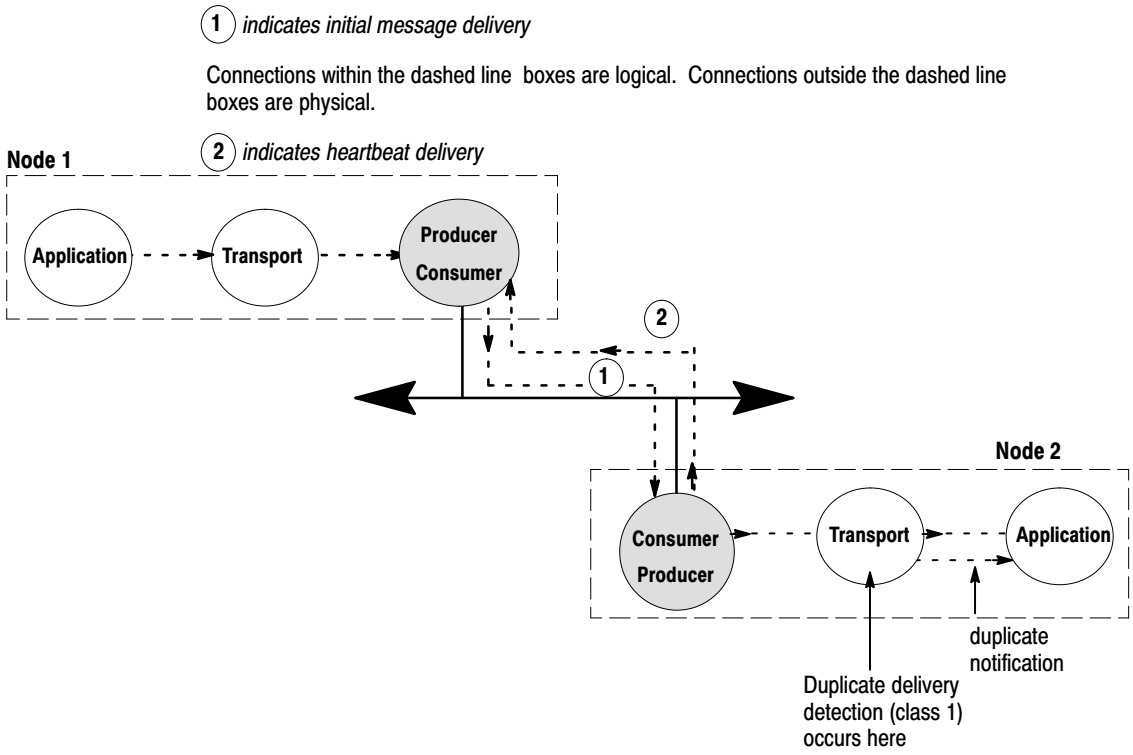
- *point-to-point* connections use one producer and only one consumer. No additional connections can be added.
- *multicast* connections allow one producer of data with more than one consumer.

Both types of connections can be defined further by the application, depending on the particular services that application requires.

Point-to-Point Class 1

Figure 1.15 illustrates a point-to-point connection between applications. In this example, data is simply sent from one application to another. No services other than data delivery and duplicate detection are provided by class 1. This type of connection is commonly used for cyclic I/O data transfers (class 1).

Figure 1.15
Point-to-Point Connection Using Transport Class 1



A **client** is an application that requests data from another application on an existing connection.

A **server** is an application that responds to a client's request by sending the requested data on an existing connection.

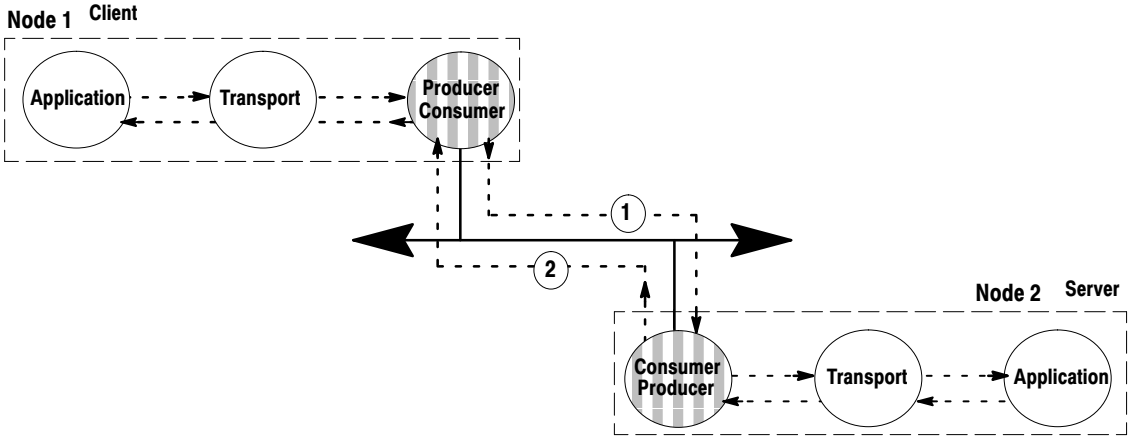
Point-to-Point Class 3

Figure 1.16 illustrates a point-to-point connection with delivery verification. In this type of connection, the application can specify a class 3 transport. The delivery notification is used for message verification (class 3). A typical use for this type of connection is **client/server** message traffic.

Figure 1.16
Point-to-Point Connection Using Transport Class 3

Connections within the dashed line boxes are logical. Connections outside the dashed line boxes are physical.

(1) *indicates initial message delivery* (2) *indicates the reverse data path for delivery verification*



Important: Delivery verification is not a point-to-point connection requirement, only an available augmentation.

Multicast Class 1

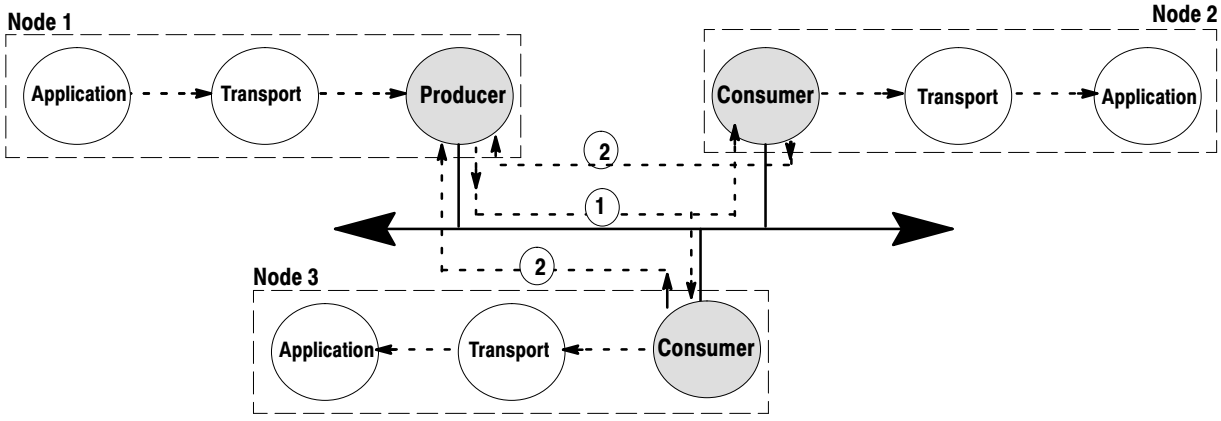
Figure 1.17 illustrates a multicast connection that can be defined further as class 1 by the application. This example is similar to Figure 1.15. The application may specifies that duplication detection is required. A common use for this type of multicast connection could be an adapter sending inputs to multiple scanners.

Figure 1.17
Multicast Connection Using Transport Class 1

(1) *indicates initial message delivery*

Connections within the dashed line boxes are logical. Connections outside the dashed line boxes are physical.

(2) *indicates heartbeat delivery*



Notes:

Aonl koeCcl Cclwokd I kolo Noe

This chapter describes the method by which the ControlNet network:

- provides determinism for control and I/O data
- facilitates unscheduled messaging
- allocates resources
- allows access to the network's media

This chapter contains these sections:

Section	Page
The Media Access Method	1
Scheduled Service	2
Unscheduled Service	3
Guardband	4

The Media Access Method

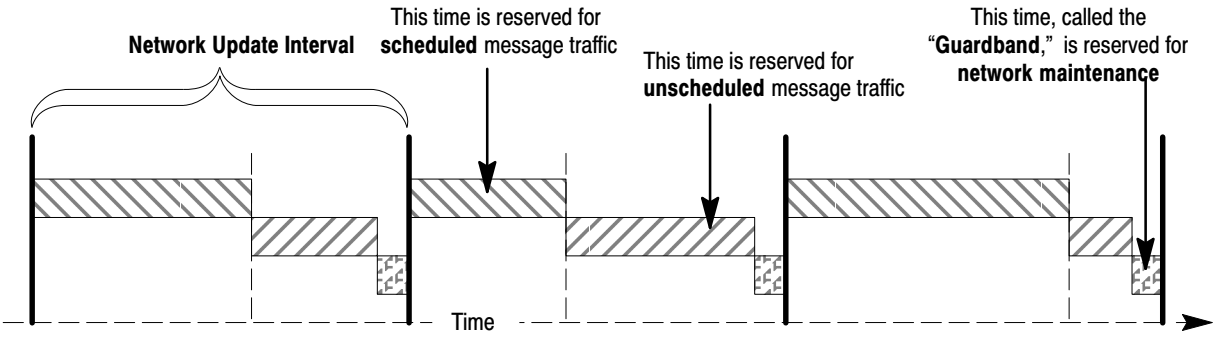
Concurrent Time Domain Multiple Access (CTDMA) is the algorithm with which the ControlNet network transmits data.

The **Network Update Time (NUT)** is the repetitive time interval in which data can be sent on the ControlNet network.

Access to the network is determined by time. Each node may transmit only during its assigned turn, which falls within a specified time frame. An algorithm called **Concurrent Time Domain Multiple Access** (**CTDMA**) regulates the opportunity to transmit. This opportunity repeats itself at precise intervals. The **Network Update Time** (**NUT**) is the fixed and known rate at which this network update interval (NUI) is repeated. Figure 1.1 illustrates the repeating NUI in which nodes can transmit their scheduled and unscheduled messages. As depicted below, the NUI is divided into three sections:

- Scheduled
- Unscheduled
- Guardband (network maintenance)

Figure 1.1
The ControlNet Network's Media Access Method



Scheduled Service

Scheduled traffic is time-critical data that is sent at a fixed and repetitive rate.

SMAX is the highest network address that can reserve bandwidth in the scheduled portion of the NUI.

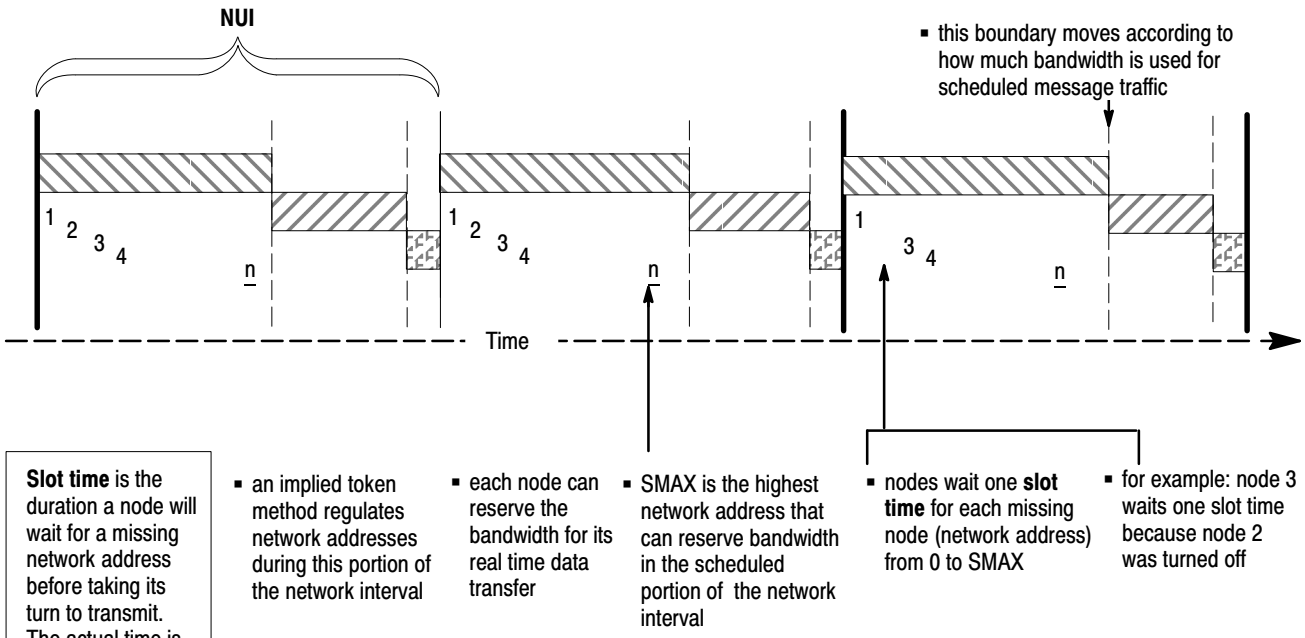
The first portion of the NUI is reserved for **scheduled** message traffic. Message delivery in this portion of the NUI is deterministic and repeatable. Every node with a network address falling between one and **SMAX** is guaranteed exactly one opportunity to transmit per NUT.

Each node can transmit up to 510 bytes during its turn. The bandwidth in this portion is reserved and configured in advance to support real-time data transfers. Typical types of scheduled messages include:

- digital data
- analog data
- peer-to-peer inter-locking data

Important: The scheduled service is from 0 to SMAX. Nodes with network addresses above SMAX will *not* send messages during the scheduled portion of the interval.

Figure 1.2
ControlNet Scheduled Media Access Service



Slot time is the duration a node will wait for a missing network address before taking its turn to transmit. The actual time is based on cable length and the number of repeaters.

Slot time is the duration a node will wait for a missing network address before taking its turn to transmit. The actual time is based on cable length and the number of repeaters.

Important: An *implied token* is the manner in which a network address determines when to transmit in relation to the other nodes on the network. No actual token is passed; the pass is implied because it is based on time. Each network node either waits to hear the end of the previous address' transmission or a slot time for each missing node before sending its message. Each node remains silent until its opportunity to transmit.

Important: Network address 0 is reserved for future use. A node must not use address 0.

Unscheduled Service

UMAX is the highest network address that can communicate during the unscheduled portion of the NUI. The default is SMAX + 8.

Unscheduled traffic is data which has no time-critical constraints.

The **unscheduled** portion of the NUI begins after all scheduled nodes have had an opportunity to transmit. The time remaining before the beginning of the guardband is available on a sequentially rotating basis to all nodes, with a network address between 0 and **UMAX**. This rotation continues until the beginning of the guardband. *The right to transmit first in the unscheduled portion rotates one node per NUI.*

Important: A node may have the opportunity to transmit many times during the unscheduled portion of the NUI; however, a node is not guaranteed an opportunity every NUI**.**

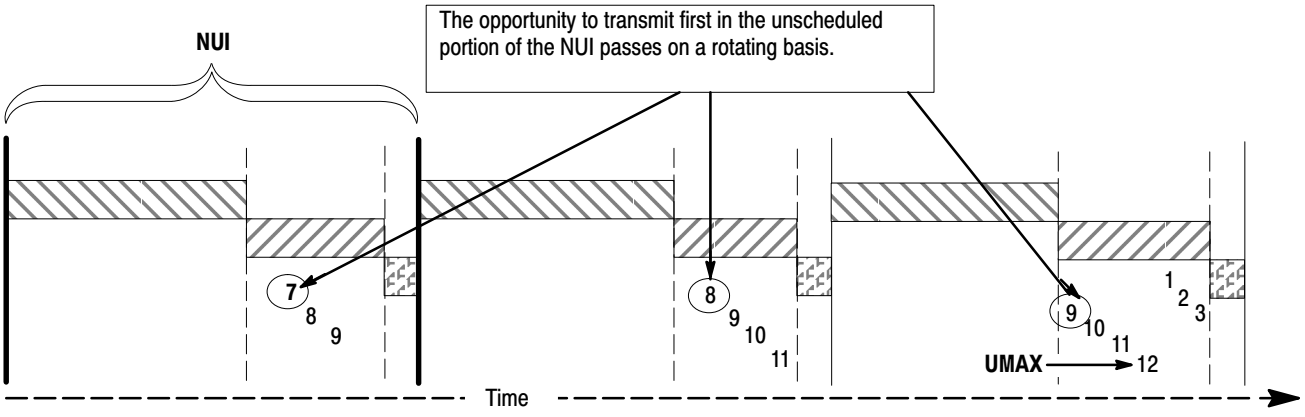
In Figure 1.3, node 7 begins the unscheduled portion in the example's first NUI. Node 8 transmits first in the second NUI regardless of who finished the last interval. Typical types of unscheduled data include:

- connection establishment
- peer-to-peer messaging data
- programming data (uploads and downloads)

Important:

- the unscheduled service is from 0 to the value of **UMAX**. In addition, UMAX is always greater than or equal to SMAX
- nodes with addresses greater than SMAX and less than or equal to UMAX may only send unscheduled messages
- nodes with a network address less than or equal to SMAX may send both scheduled and unscheduled messages
- nodes with network addresses above UMAX may *not* communicate on the ControlNet network

Figure 1.3
ControlNet Unscheduled Media Access Service



- An implied token method regulates network addresses during this portion of the network interval
- unscheduled delivery is not deterministic or repeatable
- nodes wait one slot time for each missing node (network address) from 0 to UMAX
- each node may transmit many times or not at all

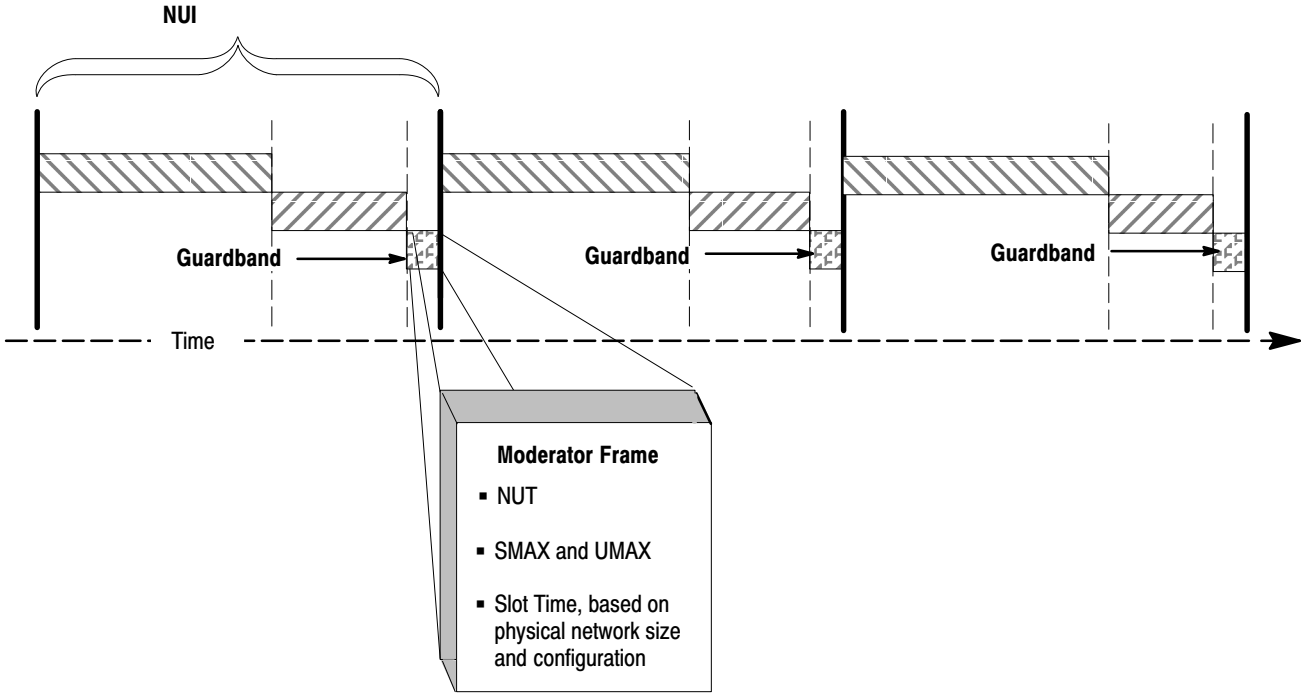
Guardband

The **Guardband** is the final part of the NUI and is reserved for network maintenance. During this time, the **moderator** transmits information, called the **moderator frame**, which keeps all nodes synchronized (Figure 1.4).

The **moderator** is the node with the lowest network address. During the Guardband, this node transmits the **moderator frame**.

Important: As the application dictates, the user can perform a network change algorithm to edit configuration information within a moderator frame. This change may or may not force a change in the NUT.

Figure 1.4
The Moderator Frame Transmitted During a NUI Guardband



Important: Each node's ASIC has the ability to act as a moderator. If the current moderator node ceases to operate, the node with the next higher address assumes the moderator duties.

Aontkod dt CnNceklk

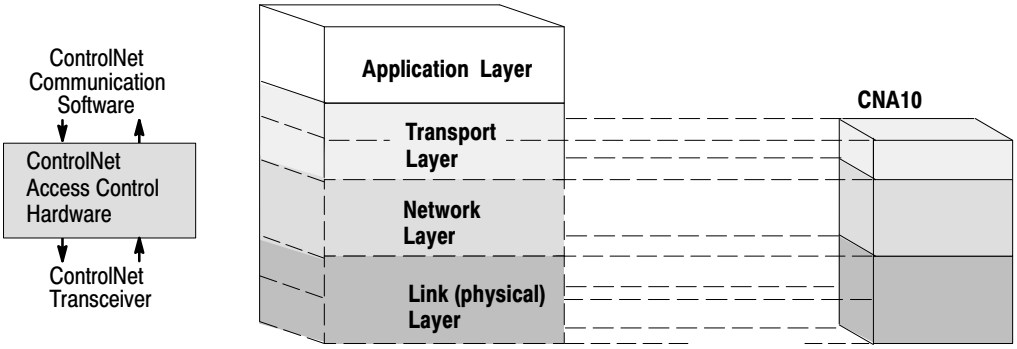
This chapter contains these sections:

Section	Page
The CNA10 ASIC	1
Protocol Functions	4
Communication Processor	5
Dual-port RAM	5
CNA10 Software Components	6
Communication Example Code	8

The CNA10 ASIC

ControlNet media-access control-hardware (**ASICs**) are interface chips that allow a product to access a ControlNet network. The ControlNet ASIC is called CNA10. The CNA10, in conjunction with class-specific Firmware, provides connection layer services, as shown in Figure 1.1.

Figure 1.1
ControlNet CNA10 and the Connection Layer Coverage It Provides



The CNA10 ASIC is an interface chip for Intel™ and Motorola® host processors. It provides the following features:

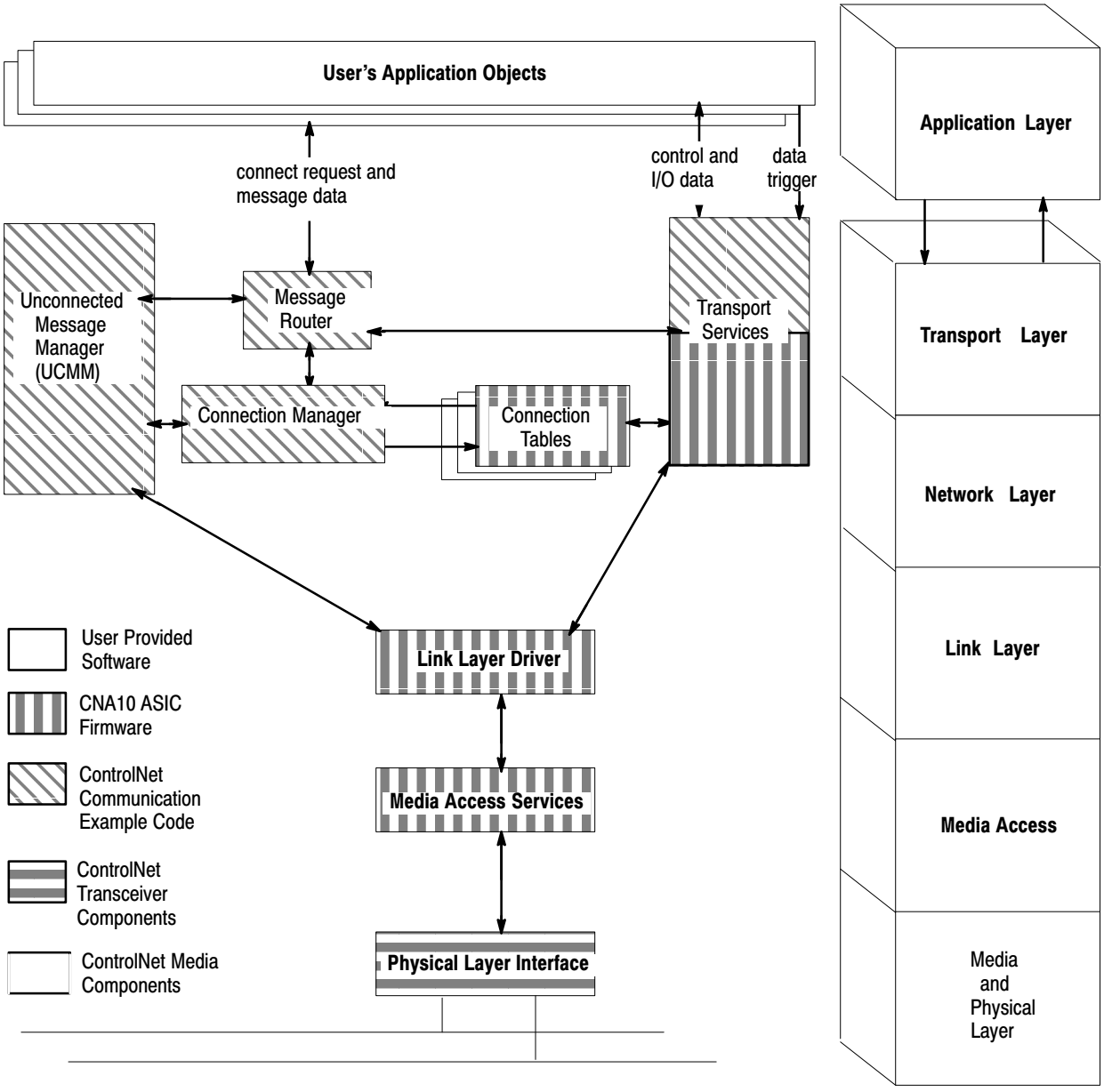
- a 16-bit communication processor
- a 3K byte dual-port RAM interface
- hardware support for 31 connected channels
- a data quarantine service
- transport services

The protocol machine within the CNA10:

- contains media access controller
- receives data from the host processor and network
- transmits data via its modem and a transceiver onto the network
- filters data received from the network
- holds data to be transmitted or received in buffers within the ASIC
- supports optional redundant media
- contains ControlNet Object Powerup and State Transition Logic.

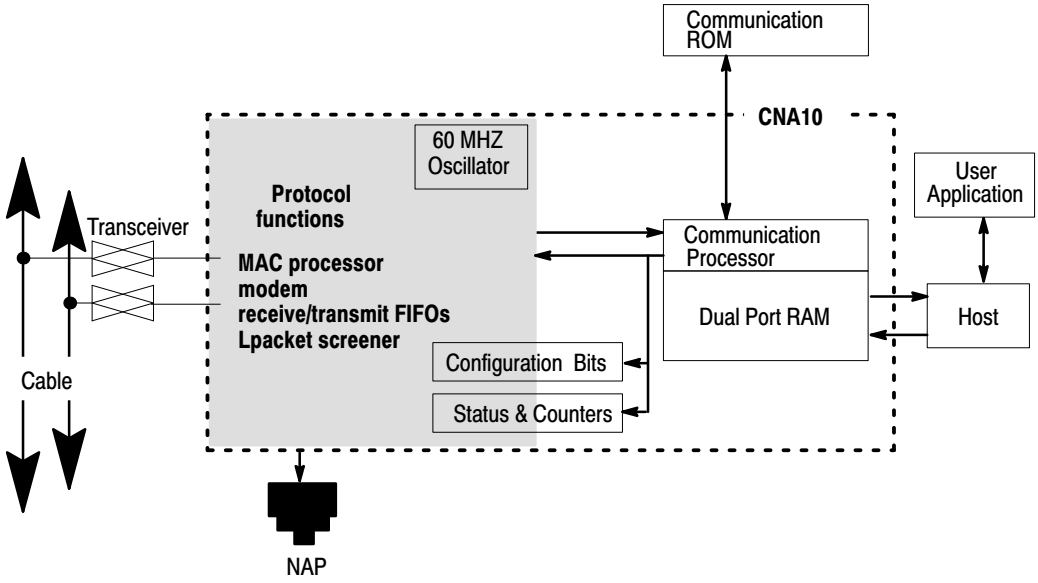
The CNA10, Allen-Bradley's ControlNet ASIC, supports all required media access functions. In addition, through a dedicated *on-chip* communication processor, the CNA10 ASIC supports much of the required transport protocol, shown in Figure 1.2.
This facilitates a simple dual-port RAM interface to a user-supplied host-processor, shown in Figure 1.3

Figure 1.2
The ControlNet Communication Model and the CNA10 Services

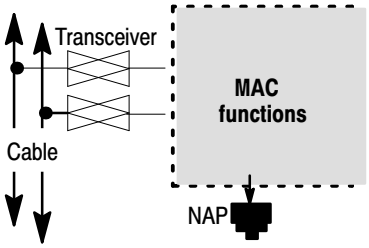


As shown in Figure 1.2, the communication example code runs in a host-processor and is responsible for supporting all layers below it. It is application-specific *example code* and is offered by Allen-Bradley to assist designers in creating their own product software. Because every product has slightly different application requirements and may require a different host-processor, it is not practical to supply a single software package to fit all needs. Therefore, Allen-Bradley has developed several *examples* of how a specific application could be applied using the CNA10. These examples can be used as a foundation for a CNA10-based product.

Figure 1.3
Block Diagram View of the CNA10



Protocol Functions



The CNA10 ASIC protocol engine supports the following features:

- MAC processor
- modem services
- transmit and receive FIFOs
- Lpacket screeners
- cable redundancy
- NAP
- signal integrity checking (**CRC**)

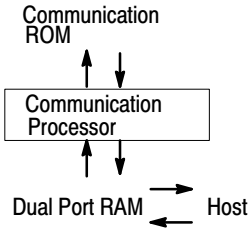
CRC is an acronym for **C**yclic **R**edundancy **C**heck.

The CNA10 *modem* sends and receives data in the form of MAC frames. Its protocol controller constantly tracks which node is next to transmit and manages the receive and transmit process. When a packet is waiting to be sent, it is held in *transmit FIFOs* until its turn. Before a received Lpacket can be processed, it is *screened* by the *Lpacket screener.* The Lpacket's **CID** is read and compared to an internal, host-programmed list of CIDs. If a match is found, the screener attaches an index value to the Lpacket and sends it to the communication processor.

CID is an acronym for **C**onnection **ID**entifier.

Communication Processor

The communication processor is a 16-bit micro-processor responsible for the implementation of communication transport functions. It quarantines packets from the receive FIFO until their CRC can be verified. The communication processor also sends and receives data from the dual-port RAM. To the host, the CNA10 simply appears to be RAM.

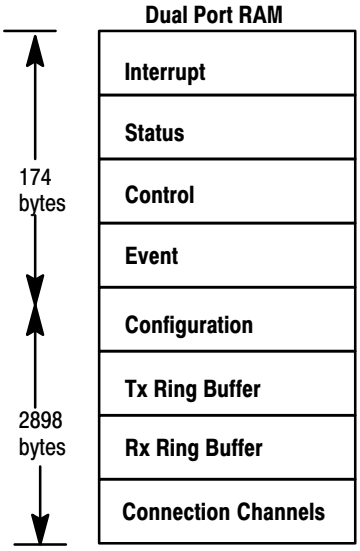


The CNA10 provides one transmit and one receive ring buffer that supports an Unconnected Message Manager (UCMM) transport as well as 31 connected channels. In general, ring buffers are used for unconnected messaging such as open/close requests and client/server messaging, while the connected transport channels are used for real-time I/O data transfers.

For more information, refer to the pages listed in the table below.

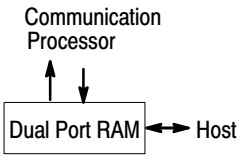
For information about:	See:
MAC Frames and Lpackets	ControlNet Communication, page 1
CTDMA and NUT	ControlNet Network Protocol, page 1
Connection IDs	ControlNet Communication, page 3

Dual-port RAM



The dual-port RAM is the point at which the host processor physically and logically connects to the CNA10. It is 3K-bytes in size and provides buffers for:

- configuration
- interrupt control
- communication services
- one receive and one transmit ring buffers (UCMM and T3)
- each of the 31 connecting channels (two channels required for each T1 connection and one channel required for each T3 connection with its data routed through the ring buffers)



Important: The sum of all areas must not exceed 3K-bytes. In addition, the boundary between the ring buffer and connection buffers is fixed at the time of configuration.

These features facilitate transport support inside the CNA10.

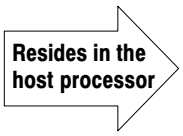
Table 1.A
Connected and Unconnected Buffer Sizes

Buffer	Minimum	Maximum	Default
31 connected channels	4 words per channel	252 words	Not applicable
UCMM Rx	260 words	N/A	260 words
UCMM Tx	1 word	N/A	33 words

CNA10 Software Components

CNA10 software components are located both in the ASIC firmware and in the host processor. These components operate together to provide services to applications. The following description builds a general profile of its capabilities by explaining each of its major software components and processes.

Unconnected Message Manager (UCMM)



Unconnected Message Manager delivers and buffers messages between nodes that do not have an available and established connection between them. The messages may be a request to open a connection or simply non-repetitive, non-time critical data. All ControlNet nodes are required to have UCMM message transmit and receive capability. For this purpose, the CNA10 provides transmit and receive services on an independent channel across a single link. UCMM messages may be serviced anytime the ASIC is **online**. They do not require prior *setup* or *negotiation* by the client and server. The UCMM service is primarily used for:

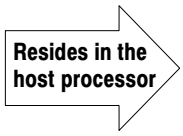
Online means to be actively participating on the communication network.

- establishing real-time data transfers
- non-connection event information between nodes

Because it is not necessary to establish a connection for UCMM messages, they may be sent at any time. *If a response timeout occurs, the application is responsible for any retries.*

Important: These messages have no guarantee of delivery. Also, a communication failure will result in a response timeout.

Message Router (MR)



The Message Router *routes* all incoming messages to the appropriate objects. Valid destinations are any internal object class as well as any application object-class that has been registered with the MR. All received UCMM messages are forwarded to the MR for delivery within the node.

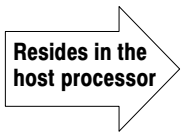
Transports

All message transfers use a specific transport class. Unconnected messages have one class called UCMM transport while the connected messages have two, transport classes 1 and 3. The table below indicates which transport classes are supported by the CNA10. The bulleted items are CNA10 supported transport functions.

- allocation/de-allocation
- start/stop
- write
- trigger

Class Number	Class Name	CNA10 Supported ?
1	Duplicated detection	Yes
3	Verified	Yes

Connection Manager (CM)



The Connection Manager is responsible for establishing and removing connections supported by the CNA10 ASIC. Also, as the CM determines the allocation of resources, it must determine if a particular connection request can be honored by its node.
The availability of connection channels and bandwidth allocation directly impacts the acceptance or rejection of a connection request.

Time Critical Bandwidth Allocation in the Connection Manager

Upon start up, part of the configuration data received specifies how much time-critical bandwidth on the network has been allocated to the node; it is allocated strictly on a *first-come, first-serve* basis. Depending on the sophistication of the application, the CM can be designed for one of three levels of connection request versus bandwidth reviewing:

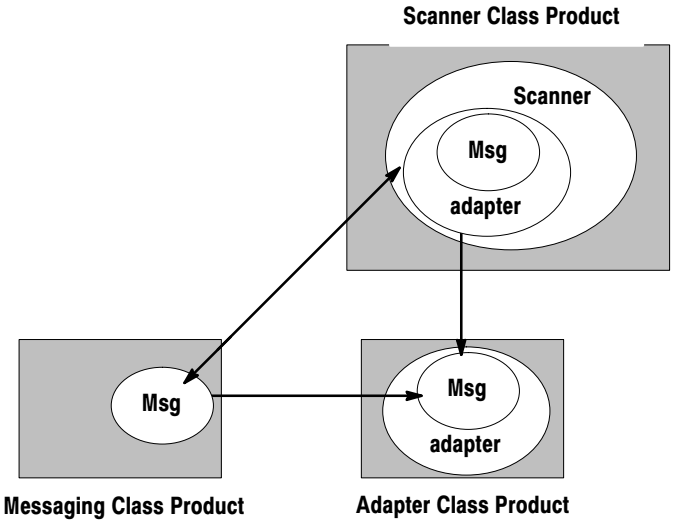
- **low-end**: Connections supported by the node are fixed. The CM can only accept a request that exactly matches one of the available connections.
- **mid-range**: The CM compares connection requests to a variable length list of acceptable connections. The product application has the option to modify the list when desired.
- **high-end**: The CM will attempt to optimize the allocation of ControlNet bandwidth. It minimizes *stale* data by generating a connected-data transmit-schedule with uniform loading over time at the requested throughput.

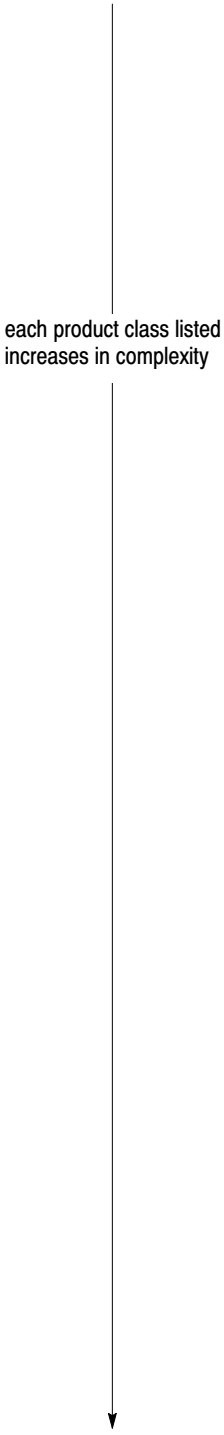
Important: Connections can only be established to either the product application objects or to the MR object.

Communication Example Code

The communication example code, shown in Figure 1.2, supports a set of application objects and interfaces. This set of objects and interfaces may differ, depending on the type of product being developed. Communication software examples have been developed with the appropriate objects and interfaces to support the real-time I/O class and the non-time-critical messaging class. Examples of how products would exchange data with the other classes is shown in Figure 1.4

Figure 1.4
ControlNet Product Classes





Messaging Class

Messaging Class products support only unscheduled messages to all product classes; all other product classes also support unscheduled message traffic. A Messaging Class product is an initiator or responder to connections (T3) and exchanges unconnected client/server messages (UCMM). It is only used in non-time critical applications because response time in the ControlNet unscheduled service is not deterministic. Typical Messaging Class products include:

- monitoring devices
- software-based supervisory products
- operator interface devices
- programming tools
- PCMCIA interface cards
- other computer interface cards
- calibration tools

Adapter Class

An Adapter Class product emulates functions provided by traditional rack-adapter products. This type of node exchanges scheduled real-time I/O data with a Scanner Class product (example, a PLC *scanner*); it does not initiate connections on its own. However, it can support unconnected messages as a server. This facilitates the Adapter Class product's ability to exchange real-time I/O data with a PLC processor as well as unscheduled messages to exchange device calibration, status, and configuration information with a Message Class, Scanner Class, or other Adapter Class node. Typical Adapter Class products may include:

- rack adapters
- drives
- weigh scale
- operator interface devices
- welders
- robots

Scanner Class

A Scanner Class product exchanges scheduled real-time I/O data with Adapter Class and Scanner Class products. This type of node can respond to connection requests and can also initiate connections on its own. In addition, the Scanner Class supports unconnected messages as a client or server. This facilitates emulation of a PLC processor in all respects. It can exchange data with other products in its class as well as directly to racks of I/O cards. Typical Scanner Class products may include:

- PLC processors
- I/O scanners
- drives
- motion controllers
- robots
- CNCs