Bulletin: 30598-713-01

Date: Dec. 8, 1988

Instruction Bulletin

Subject: SY/MAX®

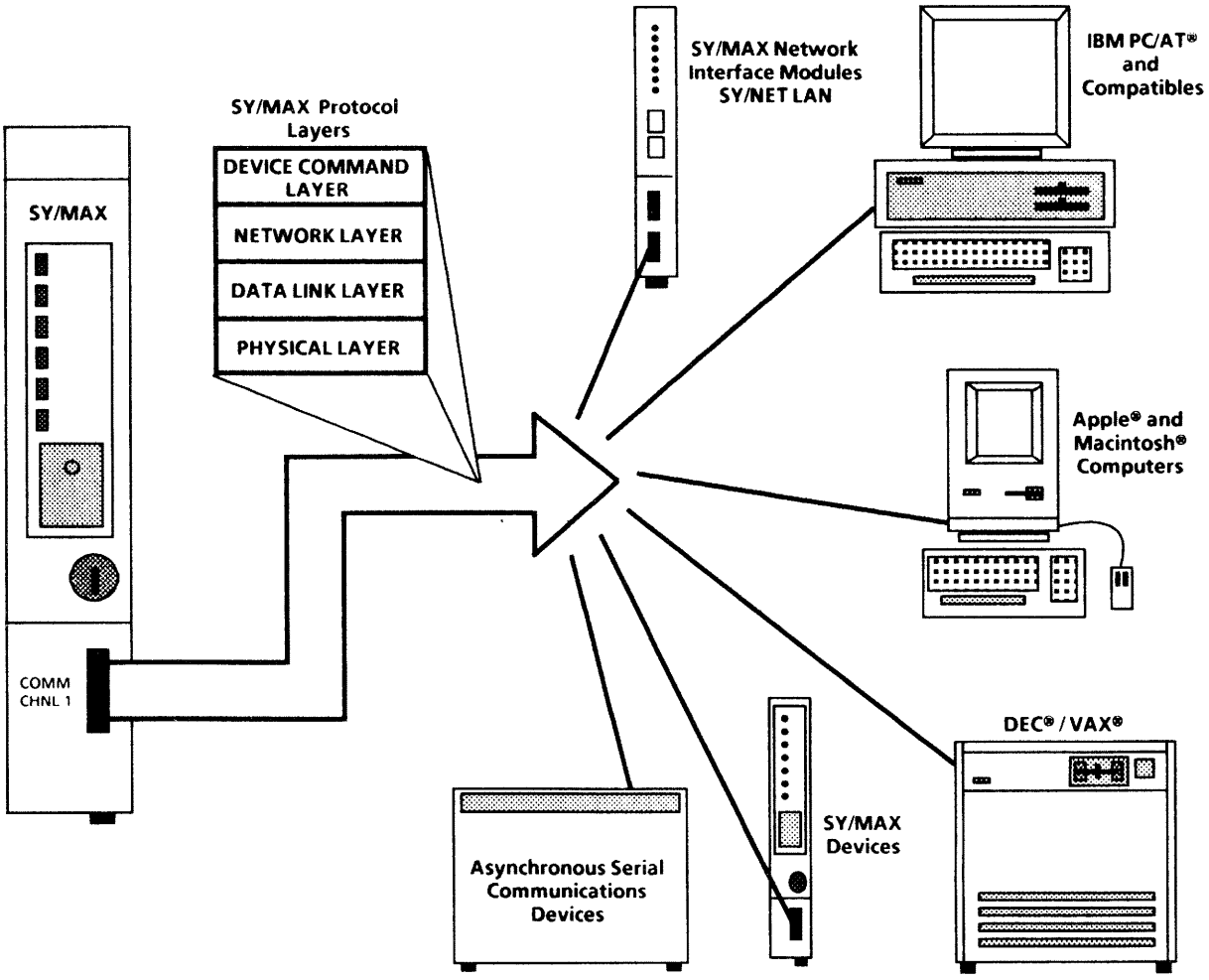
Point-to-Point Communications Protocol for Data Transfer Operations

DESCRIPTION:

This instruction bulletin describes the SY/MAX Protocol for Point-to-Point Data Transfer Communications. SY/MAX Protocol is used in all SY/MAX programmable controllers and SY/MAX devices that support serial communications.

SY/MAX Protocol provides the following seven facilities for transferring data from device to device through standard serial communications or via the SY/NET® Local Area Network:

- Priority READ Registers
- Non-Priority READ Registers
- Priority WRITE Registers
- Non-Priority WRITE Registers
- READ Multiple Registers
- Alarms
- Print ASCII



NEW LISTING
Printed in U.S.A.

SQUARE D

Dedicated to Growth • Committed to Quality

NOTICE

The products and services described in this manual are useful in a wide variety of different applications. Therefore, the user and others responsible for applying the products and services described herein are responsible for determining their acceptability for each application. While efforts have been made to provide accurate information within this manual, revisions and updates make it impossible to guarantee that the information is the newest available for a particular product.

Under no circumstance will the Square D Company or any of its subsidiaries be responsible or liable for any damages or losses , including indirect or consequential damages or losses, arising out of either the use of any information contained within this manual or the use of any products or services referenced herein.

No patent liability is assumed by the Square D Company with respect to the use of any of the information, products, circuits, programming, or services referenced herein.

The information contained in this manual is subject to change without notice.

TABLE OF CONTENTS

This page intentionally left blank.

1 INTRODUCTION

1.1 General

This document describes SY/MAX point-to-point communications and the protocol associated with it. Written for the technically-oriented reader, its primary objective is to serve as a complete and comprehensive source of information necessary to understand the protocol and design the software capable of handling that protocol. This document assumes basic understanding of data communications concepts and, although some of the material presented here does require understanding of software design for data communications, those portions of the document can be omitted, if desired.

CAUTION

Please note, that in order to design software capable of handling SY/MAX communications CORRECTLY and EFFICIENTLY, it is of vital importance that ALL aspects and features of the protocol are understood, especially when making decisions concerning implementation of a SUBSET of the protocol. In any implementation of SY/MAX Protocol, it is recommended that the Data Link Layer be implemented in Full, as described in Section 6.

1.2 Organization of the Document

This document is organized into eight major sections, plus the appendixes.

Section 2 – SY/MAX and SY/NET Communications Concepts
introduces the terminology used throughout the rest of the document and explores the relationship between SY/MAX Point-to-Point and SY/NET network communications.

Section 3 – SY/MAX Point-to-Point Communications Overview
presents an overview of the protocol in form similar to the seven-layer OSI [1] model.

Section 4 – Device Command Layer
presents a complete definition of the Device Command Layer of SY/MAX protocol. Both the syntactic and the semantic aspects are presented, including the complete set of commands.

Section 5 – Network Layer
presents a definition of the Network Layer of SY/MAX protocol, including its functions and responsibilities.

Section 6 – Data Link Layer
presents a complete definition of the Data Link Layer of the protocol, including syntax and semantics. Startup, handshake, data transfer and error recovery sequences are defined here as well.

Section 7 – Physical Layer
contains a complete definition of the Physical Layer of the SY/MAX Point-to-Point communications protocol, including a description of the physical media used and a specification of the control signals and pin configuration.

Section 8 – Software for SY/MAX Communications
presents suggestions on the design and architecture of software developed to implement SY/MAX Point-to-Point communications. This section should be of interest to the reader aiming to develop software and includes examples, guidelines, and application-specific considerations.

Appendixes
contain quick-reference material, organized mostly into tables and charts. It is highly recommended that the reader gets acquainted with contents of the appendixes as early as possible to make the best use of information presented there.

(1) OSI - Open Systems Interconnect, as defined by the International Standards Organization (ISO).

This page intentionally left blank.

2 SY/MAX AND SY/NET COMMUNICATIONS CONCEPTS

2.1 General

SY/NET is a high speed local area network that allows SY/MAX family devices, as well as other devices, to communicate with each other. It consists of up to 100 Network Interface Modules (NIM), which are used to connect the end-devices to the network, and a twin-ax cable up to 15,000 feet long. Each of the modules can have 2 end-devices (i.e. Processors, D-Logs, etc.) connected to it, so up to 200 end-devices can be communicating over a single SY/NET LAN (see Figure 2.1). However, since network to network communications are easily facilitated by connecting one network's NIM to a NIM from another network, a virtually unlimited number of devices can be interconnected to form a multi-network system (see Figure 2.2).

Typically, two distinct types of communication are taking place when a network is operating. First, high speed communications are occurring between the NIMs on a twin-ax cable. Throughout the document, these types of communications will be referred to as "SY/NET" or "network" communications.

There is also data being transferred between the NIM and the end-device attached to it. This type of communications is commonly known as "SY/MAX" or "Point-to-Point" and is marked on Figures 2.1 and 2.2 with a solid line, representing a dual twisted-pair cable. The twin-ax cable and connections are denoted by dashed lines in Figures 2.1 and 2.2.

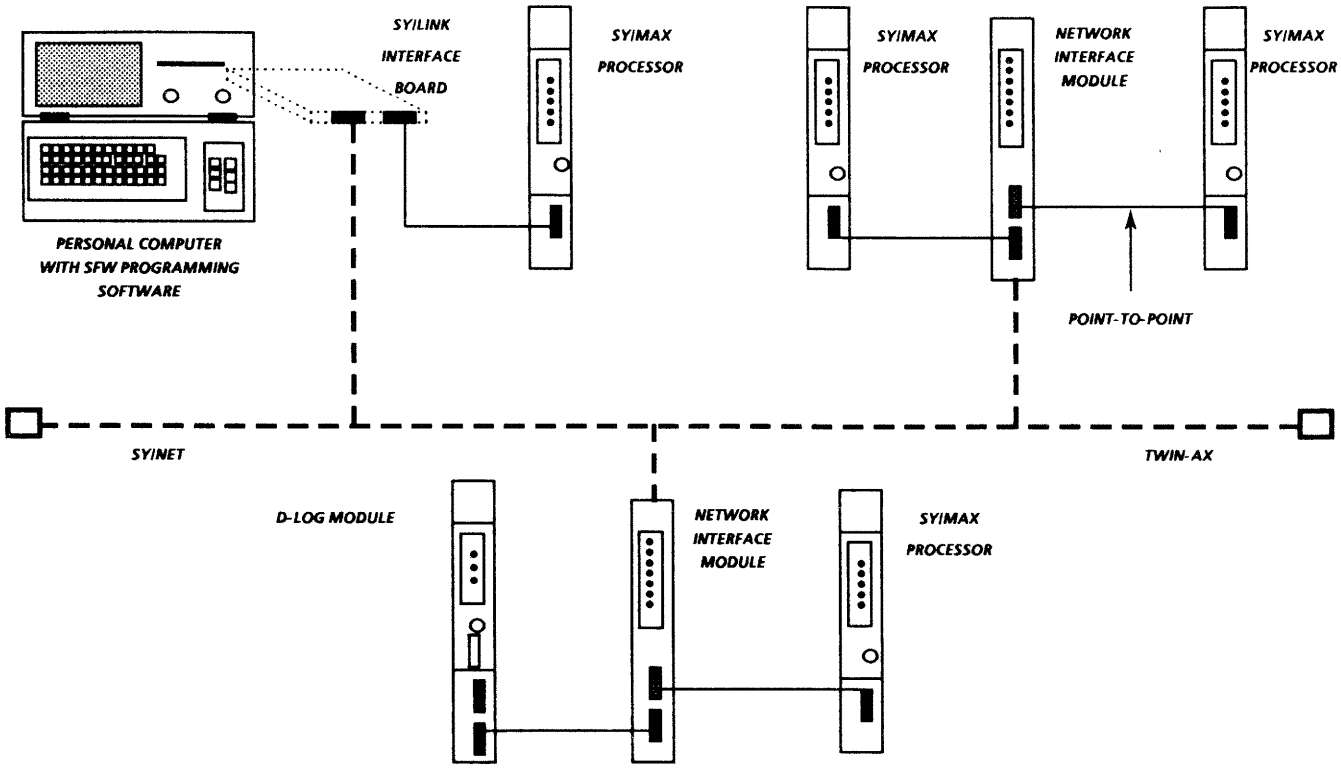


Figure 2.1 - Typical SY/NET Local Area Network Configuration

Note that, as illustrated in Figure 2.2, selected end-devices such as ASCII printers, Class 8881 Processors and some other devices which do not use the SY/MAX Point-to-Point communication protocol, may be connected to the network.

As previously indicated, this document describes only the Point-to-Point communications. It is not intended to address the SY/NET issues. Thus, any SY/NET information presented here serves a sole purpose of illustrating the relationship between the Point-to-Point and the Network communications, and should only be used as such. For more information on SY/NET, see bibliography notes in the appendix. For more information on the NIM, refer to the NIM instruction bulletin 30598-257-02 (see Appendix A).

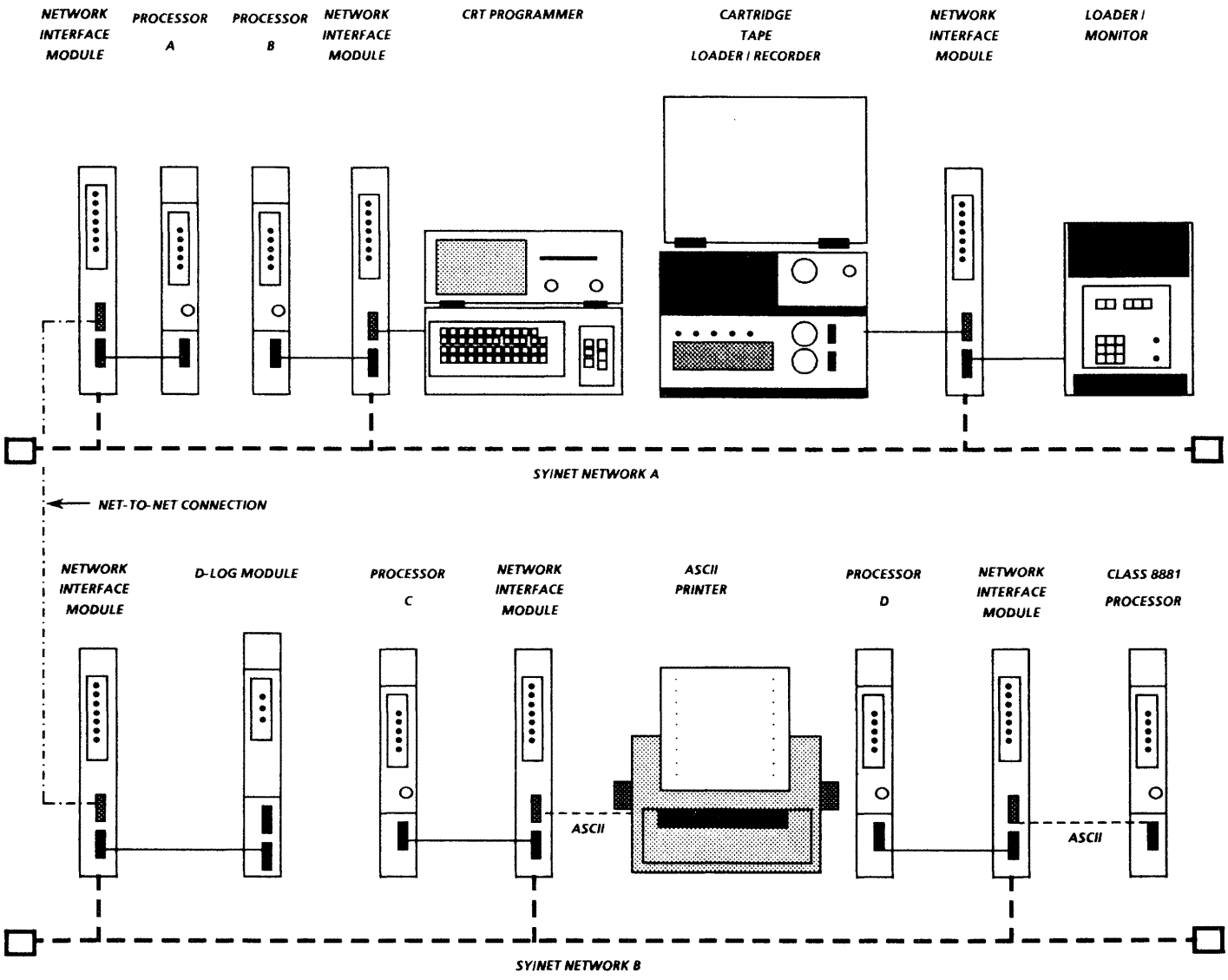


Figure 2.2 - TYPICAL Multi-Network Configuration

3 SY/MAX POINT-TO-POINT COMMUNICATIONS OVERVIEW

3.1 Introduction

Protocols provide the rules for data communications systems. According to the Open System Interconnection (OSI) model, communication protocols can have a multi-layer structure. It is necessary therefore to define the protocols at each level precisely by specifying both the syntax (format) and the semantics (procedural rules) for the protocol. In this Section, the protocol layers for SY/MAX Point-to-Point communications are defined. Wherever possible, the definitions follow the form of the OSI model.

3.2 OSI Model and SY/MAX Point-to-Point Communications

The Reference Model of Open System Interconnection (OSI), defined by the International Standards Organization (ISO), serves as a basis for defining the conceptual layers necessary for data communications. Although not every layer of the OSI Model is present in the SY/MAX Point-to-Point protocol definition, it is easier to understand SY/MAX protocol by comparing it to the OSI model.

There are seven layers defined in the OSI model. The purpose of the model is "to provide a common basis for coordination of standards development for the purpose of systems interconnection, while allowing existing standards to be placed into perspective with the overall Reference Model." The layers in the OSI Model are illustrated in Figure 3.1. The applications or end users reside above the highest layer (7) of the model.

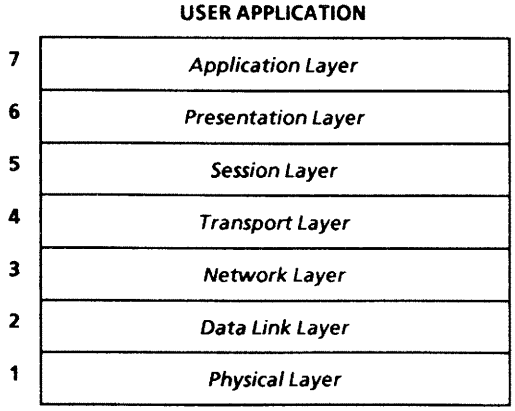


Figure 3.1 - OSI Reference Model

Every layer shown in Figure 3.1 does not necessarily exist in every data communications system. In the case of SY/MAX Point-to-Point communications, the top four layers are collapsed into a single layer which provides most of the functions supplied by the original layers.

It should be noted that only layer 1, the Physical Layer, involves a physical transfer of information between the two communicating systems. Layers 3 and above typically involve information exchange between layers within the same system. Layer 2, the Data Link Layer, controls the transfer of information on the physical medium.

Information flow in the OSI Model occurs as follows:

The data to be transferred is generated at the Application Layer. The block of data from the application is passed down through each layer to the Physical Layer, where it is transferred out of the system. Each layer usually "frames" the data passed to it by a higher layer by adding a header and a trailer. On the receiving end, as the data moves up from one layer to another, the headers and trailers are removed, so the original data sent will arrive correctly at the next higher level. Note that in addition to framing, the form of the data representation may change as well (i.e. ASCII to hex).

3.2.1 OSI APPLICATION LAYER

The Application Layer is typically application dependent. It basically provides the services used directly by the applications. It is important to note that the applications do not reside in this layer. The layer serves, rather, as a window through which the application gains access to the communication services provided by the model. Note that the application itself can be in the form of software running on the system or an end user working on a terminal. The Application Layer is responsible for the overall management of communication transaction.

3.2.2 OSI PRESENTATION LAYER

The Presentation Layer controls data transformation, formatting and syntax, and allows the modification of the information-handling characteristics of one application process to those of another. This layer can perform such functions as data compression/decompression and encryption.

3.2.3 OSI SESSION LAYER

The Session Layer provides administrative services such as setup and termination of the session between the two systems. It also provides the dialogue services to control data exchange and to synchronize the operation of both systems.

3.2.4 OSI TRANSPORT LAYER

The Transport Layer is responsible for providing services such as flow regulation and optimization of communications resources usage. It controls the end-to-end data transfer without the detailed knowledge of the path the data has to follow.

3.2.5 OSI NETWORK LAYER

The Network Layer provides the routing functions and isolates the Transport Layer from the details of data movement through the network. Among other services, it also supplies alternate route selection and adaptive routing, in which the final route is determined dynamically along the way. Error recovery is also a responsibility of this layer.

3.2.6 OSI DATA LINK LAYER

The Data Link Layer controls the data traffic on the physical medium of the Physical Layer. This layer provides services for communication startup, maintenance and termination. It is also responsible for data framing, block sequencing and error detection/recovery. This layer provides control for point-to-point as well as multidrop link configurations.

3.2.7 OSI PHYSICAL LAYER

The Physical Layer defines the electrical and mechanical interfaces to the physical transmission medium. It also provides control and interpretation of all signals and data set controls (hardware handshaking) of the interface.

3.3 SY/MAX Point-to-Point Protocol

3.3.1 INTRODUCTION

SY/MAX Point-to-Point communications involve two devices communicating over a link as illustrated in Figure 3.2. The two communicating devices can be referred to as "stations" and will be referred to as "station A" and "station B". In this case Station A is a Processor and Station B is a Network Interface Module.

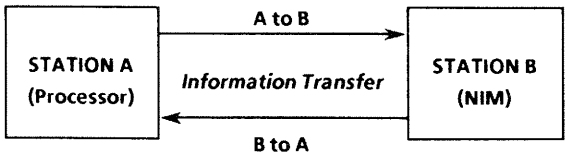


Figure 3.2 - SY/MAX Point-to-Point Communication Configuration

Square D's SY/MAX Point-to-Point protocol was being developed at roughly the same time as the OSI model was being defined. Due to the time constraints, it was not practical to delay SY/MAX protocol development until the OSI model was completed. This factor, combined with the fact that many of the features of the complete OSI model were not necessary for SY/MAX communications, resulted in the protocol not following the OSI model precisely, primarily in the difference in number of layers.

As illustrated in Figure 3.3, there are four layers in the SY/MAX Point-to-Point protocol definition: Physical, Data Link, Network, and Device Command. Just as in the OSI model, the application itself is positioned on top of the highest layer. The first three layers follow three layers of the OSI Model in terms of both structure and services. The Device Command Layer, however, is unique to SY/MAX point-to-point protocol. This layer is a functional replacement for the remaining layers in the OSI model, supplying the necessary subset of services provided by the Transport, Session, Presentation and Application Layers.

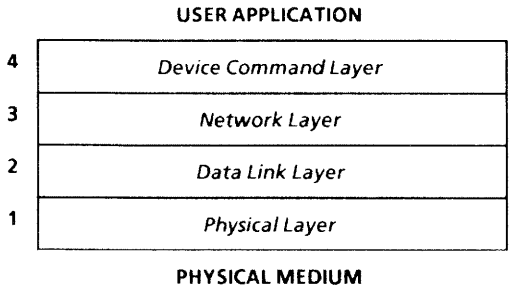
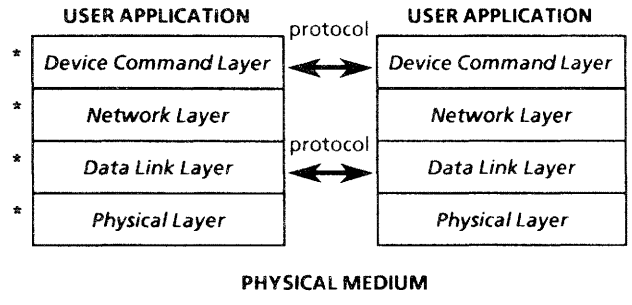


Figure 3.3 - SY/MAX Point-to-Point Protocol Layers

The detailed definition of each of the four layers follows. Note that, depending on the nature and the complexity of the layer, the description of each layer may address the following issues: discussion of the interfaces with upper and lower neighboring protocol layers (wherever applicable), definition of the responsibilities, services of the layer, and/or the protocol of the layer. The protocol of the layer can be viewed as the description of how two communicating devices exchange information following the syntax and the semantic rules defined in this specific layer.

Figure 3.4 outlines how the SY/MAX Point-to-Point protocol is defined in this document. Each layer's characteristics and responsibilities are defined as well as the services it provides for the upper layer. In addition to that, Device Command and Data Link Layer have protocol definition associated with them.



* - means layer's services, responsibilities and characteristics are present in the definition of SY/MAX protocol.

Figure 3.4 - SY/MAX Point-to-Point Protocol Definition

3.3.2 IMPLEMENTATION NOTICE

Due to the diversity of Square D's product line with unique functionality, application and performance requirements for each type of product, the implementations of some of the higher layers of the protocol are product-specific to such a degree that some protocol rules are transgressed. The protocol definition presented in this document is not meant to be product specific, but rather as formal and implementation independent as possible. Necessary product/application dependencies and performance related issues will be examined in Section 8.

3.3.3 SY/MAX PROTOCOL TRANSACTIONS

A communication transaction using SY/MAX protocol involves two devices, the initiating device and the replying device. As shown in Figure 3.5, each device is responsible for two transmissions: a message (command or reply) and a response. A communication transaction involves the following steps:

> The initiating device sends a command message, and the replying device returns a response. The replying device then sends a reply message, and the initiating device returns a response.

These are the required steps for a communication transaction. Additional steps may be required if an error is detected at any step; the extra steps perform error correction and insure error-free communications.

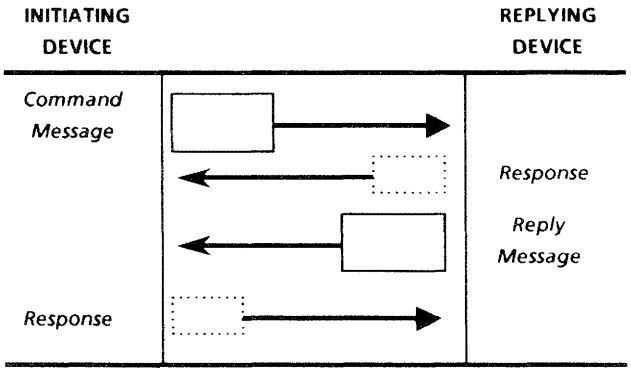
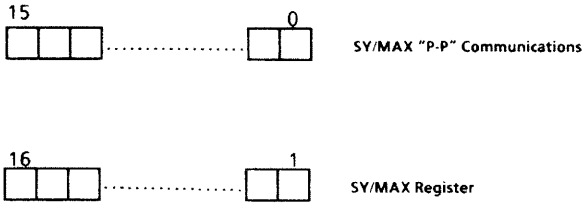


Figure 3.5 - SY/MAX Protocol Transaction

3.3.4 DOCUMENT CONVENTIONS

Throughout the rest of this document the word "Point-to-Point" will be abbreviated as "P-P" and the SY/MAX Point-to-Point protocol will be referred to as the "SY/MAX P-P protocol" or simply the "protocol".

Also note that any reference to bit number in this document is 0 based. In other words, the least significant bit is bit 0. This is different from the convention used throughout Square D's SY/MAX programmable logic controller documentation where in register context the least significant bit is bit 1.



3.3.5 DATA TRANSFORMATIONS / FRAMING THROUGH THE LAYERS

Figure 3.6 summarizes how the data is transferred and rearranged when moving through the layers of the protocol.

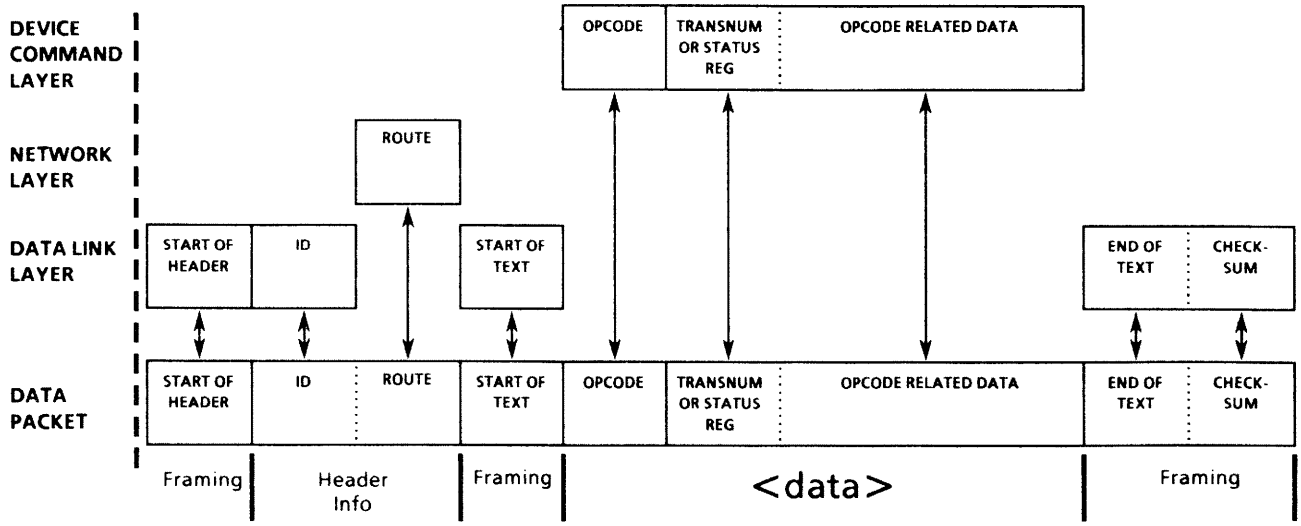


Figure 3.6 - Data Transformation Through Protocol Layers

4 DEVICE COMMAND LAYER

4.1 Introduction

This layer provides a variety of functions. Because it interfaces directly with the user application, it can be very application dependent. A complete definition is presented here, with the application/ implementation specific considerations addressed later on in the document. The Device Command Layer also interfaces with the Network Layer.

4.2 Primary Purpose/Responsibilities

The primary purpose of this layer is to perform the device command execution. This involves either acting on the commands submitted by the user application or fulfilling the requests coming in from the other station through the lower protocol layers. To accomplish this task the layer has to fulfill the following major responsibilities:

1. It must perform the transformations on the data passed from and to the user application as defined by the protocol. Since the definition of data transformations is heavily dependent on the user application, the only definition that can be provided is the final data representation used in the protocol of this layer. This aspect is presented as the syntax of the Device Command Layer protocol. Figure 4.1 depicts the data handling responsibilities of this layer.
2. This layer is also responsible for information flow from the initiating device to the replying device and back. The complete communication transaction is typically a multi-step process. This aspect of the protocol is presented in section 4.6.

When an operation is requested by the user application, the Device Command Layer will typically perform the data transformations described in number 1 above. Then, to complete the operation, it will perform all the information flow related tasks by passing the appropriate data to the next lower layer, requesting appropriate services, and verifying that the requests placed to the lower layer have been successfully fulfilled.

Note that the successfully fulfilled request does not necessarily mean that operation of the lower layer was a success; just that the request was serviced by the lower layer and success or failure for a particular reason has been indicated. This can be equally applied to a relationship between any two neighboring layers of the SY/MAX protocol.

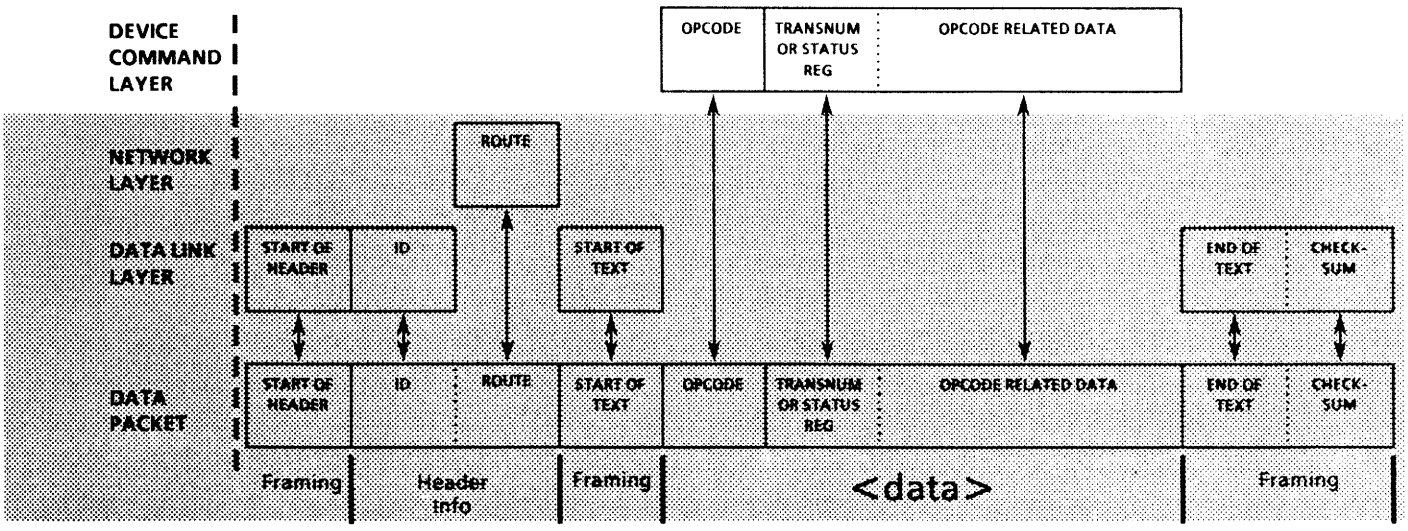


Figure 4.1 - Device Command Layer Data Handling

4.3 Device Command Layer Syntax

4.3.1 INTRODUCTION

Each communication transaction unit at this level is called a message. There are two different classes of messages in the Device Command Layer protocol: 1) command and 2) reply. A typical communication transaction involves one station sending a command and the other station answering with a reply. The messages have variable format and length, both of which depend on the opcode indicating the message's operation type.

Messages of different types have a unique purpose and structure. Some components of a message are unique to that type of message while others are shared by many types of messages. All messages can be separated into five functional categories according to the general purpose of the message. Figure 4.2 summarizes all messages of the Device Command Layer protocol; the subsequent sections define the format and purpose of each message in greater detail.

When generating a command, the opcode and the related data is passed down from the user application and transformed to fit the format of the corresponding command message. Each message contains the opcode of the operation itself and related components. The related components are referred to as "fields". Also, in some operations a transmission number (TRANSNUM) field is added to the command message for information flow control purposes.

The generation of the reply messages often involves sending back some of the fields present in the initial command message.

The format of each command and the corresponding successful operation execution reply are presented in sequence by the command categories. Note that the corresponding reply listed with each command is referred to as "the successful operation execution reply" and assumes no errors. If an error reply has to be generated as an answer to a command, it is handled differently for Priority commands (category F) than for other commands. Generation of error replies is detailed later in the document.

4.3.1.1 Alphabetical Key to the Fields

Note that all multi-byte fields must be transmitted starting with the most significant byte first.

ASCII_DATA
multiple bytes of 8-bits/character ASCII data

BIT_MASK
16-bit bit mask to selectively set or reset certain bits within a register when performing a Write Register (opcode 02). When the write is performed, the only alterable bits in the destination register(s) are the bits set in the mask. The rest of the bits in the destination register(s) are not affected by the operation. The following formula describes the transaction:

new_destination_register_value =
(REGISTER_DATA * BIT_MASK) +
(old_destination_register_value *
BIT_MASK)

A single BIT_MASK value is applied to all REGISTER_DATA values in the command.

Functional Category	Operation Type	Command Opcode (in hex)	Reply Opcode (in hex)
A. Data Commands	1. Read Register	00	86
	2. Write Register	02	80
	3. Multiple Register Read	04	8A
B. Data Replies	1. Read Register Reply	86	
	2. Multiple Read Register Reply	8A	
C. Priority Data Commands	1. Priority Read Register	20	90
	2. Priority Write Register	1E	92
	3. Print ASCII	22	92
D. Priority Data Replies	1. Priority Read Register Reply	90	
	2. Priority Write Register Reply	92	
	3. Print ASCII Reply	92	
E. General Replies	1. Operation Complete	80	
	2. Error reply	A2	

Figure 4.2 Messages for Device Command Layer Protocol

BYTE_COUNT
16-bit value; zero based (decremented); maximum value of 255 (i.e., 256 bytes) ; For example, value of 3 implies count of 4.

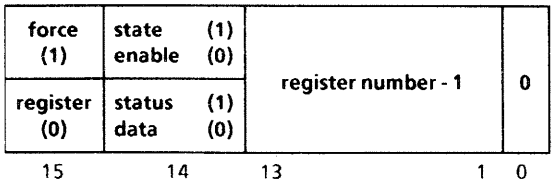
DESTINATION_REGISTER_ADDRESS
16-bit destination register address; calculations similar to REGISTER_ADDRESS.

ERROR_CODE
8-bit field; complete list in the appendix.

OPCODE
8 bit binary number as defined previously.

REGISTER_ADDRESS
16-bit value; protocol representation of an address of a general register from user application. Calculation is performed as shown:

REGISTER_ADDRESS =
*(register number - 1) * 2*
+ force + state + status;



REGISTER_COUNT
16-bit value; zero based (decremented); maximum value is 127 (i.e., 128 registers). For example, value of 3 implies count of 4.

REGISTER_DATA
multiple 16-bit binary values.

RETURN_PRINT_DATA
not defined.

SOURCE_REGISTER_ADDRESS
16-bit source register address. Calculation similar to REGISTER_ADDRESS.

START_REGISTER_ADDRESS
16-bit register address serving as a starting point in the specified operation. Calculation similar to REGISTER_ADDRESS.

STATUS_REGISTER_DATA
16-bit contents of STATUS_REG_ADDRESS register indicating the success/failure status of the command operation.

bit 15 Success/failure indicator.
- set – Operation completed successfully.
- clear – Bits 7 - 0 contain the error code. See appendix for list of all error codes.

bit 14 Run/Halt indicator.
- set – Replying device is running.
- clear – Replying device is halted.

bit 0 - 7 Error code
If bit 15 is clear, bits 0-7 hold error code. Notice that all replies indicating an error in priority data operation have bit 0 set.

STATUS_REGISTER_ADDRESS
16-bit status register address. Calculation similar to REGISTER_ADDRESS.

Note that when used in Priority Write Register, bit 15 has the following meaning:

If set, then the operation that generated the command is located in the security protected area of the originating device.

TRANSNUM
8-bit value; transmission number. Used for information flow control purposes. For non-priority messages, the TRANSNUM from the command message is used in the reply message.

MESSAGE DEFINITION
The definition of each message includes the OPCODE value in hex and Category Id which can be used to locate the message in Figure 4.2. Note that each field has a length specification associated with it in bytes. It is listed as a single number or as a range of numbers under the corresponding field of the message.

4.3.2 NON-PRIORITY DATA COMMANDS AND REPLIES

4.3.2.1 *Non-Priority Read Register*

Read REGISTER_COUNT registers starting at START_REGISTER_ADDRESS.

Command Message
Opcode: 00
Category id: A1
Format:

OPCODE	TRANSNUM	START_REGISTER_ADDRESS	REGISTER_COUNT
1	1	2	2

Reply Message
Name: Read Register Reply
Opcode: 86
Category id: B1
Format:

OPCODE	TRANSNUM	START_REGISTER_ADDRESS	REGISTER_DATA
1	1	2	2-256

4.3.2.2 Non-Priority Write Register

Write REGISTER_DATA with applied BIT_MASK starting at START_REGISTER_ADDRESS.

Command Message
Opcode: 02
Category id: A2
Format:

OPCODE	TRANSNUM	START_REGISTER_ADDRESS	REGISTER_DATA	BIT_MASK
1	1	2	2-256	2

Reply Message
Name: Operation Complete
Opcode: 80
Category id: E1
Format:

OPCODE	TRANSNUM
1	1

4.3.2.3 Multiple Register Read

Read REGISTER_DATA from all REGISTER_ADDRESS locations specified. The list of REGISTER_ADDRESSes can be up to 128 registers long.

Command Message
Opcode: 04
Category id: A3
Format:

OPCODE	TRANSNUM	REGISTER_ADDRESS	REGISTER_ADDRESS	...	REGISTER_ADDRESS
1	1	2	2		2

Reply Message
Name: Multiple Read Register Reply
Opcode: 8A
Category id: B2
Format:

OPCODE	TRANSNUM	REGISTER_ADDRESS
1	1	2-256

4.3.3 PRIORITY DATA COMMANDS AND REPLIES

This group of operations is also known as processor initiated operations. Error reply generation for the commands in this category is different than the other categories.

4.3.3.1 Priority Read Register

Read REGISTER_COUNT registers as REGISTER_ DATA starting at SOURCE_REG_ADDR register address of the remote station and place the data starting at DEST_REG_ADDR register address of the local station. STATUS_REG_DATA indicates the successful completion or failure of the operation. STATUS_REG_ADDR is the local station's register address uniquely associated with this operation.

Command Message
Opcode: 20
Category id: C1
Format:

OPCODE	STATUS_REGISTER_ADDRESS	DESTINATION_REGISTER_ADDRESS	SOURCE_REGISTER_ADDRESS	REGISTER_COUNT
1	2	2	2	2

Reply Message
Name: Priority Read Register Reply
Opcode: 90
Category id: D1
Format:

OPCODE	STATUS_REGISTER_ADDRESS	START_REGISTER_DATA	DESTINATION_REGISTER_ADDRESS	REGISTER_DATA
1	2	2	2	2-256

4.3.3.2 Priority Write Register

Write REGISTER_DATA starting at DEST_REG_ ADDR register address of the remote station. STATUS_REG_DATA indicates the successful completion or failure of the operation. STATUS_ REG_ADDR local station's register address uniquely associated with this operation.

Command Message
Opcode: 1E
Category id: C2
Format:

OPCODE	STATUS_REGISTER_ADDRESS	DESTINATION_REGISTER_ADDRESS	REGISTER_DATA
1	2	2	2-256

Reply Message
Name: Priority Write Register Reply
Opcode: 92
Category id: D2
Format:

OPCODE	STATUS_REGISTER_ADDRESS	STATUS_REGISTER_DATA
1	2	2

4.3.3.3 Print ASCII

Print ASCII_DATA. STATUS_REGISTER_DATA indicates the success or the failure of the print operation. STATUS_REGISTER_ADDRESS is the register address in the local device unique to this operation. Note that the reply is equivalent to Priority Write Register Reply, except the RETURN_PRINT_DATA field may optionally be present. The definition and purpose of RETURN_ PRINT_DATA field is not yet defined, and should not be used.

Command Message
Opcode: 22
Category id: C3
Format:

OPCODE	STATUS_REGISTER_ADDRESS	ASCII_DATA
1	2	2-256

Reply Message
Name: Priority Write Register Reply
Opcode: 92
Category id: D3
Format:

OPCODE	STATUS_REGISTER_ADDRESS	STATUS_REGISTER_DATA	RETURN_PRINT_DATA
1	2	2	2-256

4.4 Error Detection

The Device Command Layer has a responsibility to carry out two distinct types of error detection.

Type 1 –

The first type of error detection deals with verification of the format of the incoming command/reply messages. If a format of a command message is determined to be incorrect, an error reply with an appropriate error code must be generated and the device operation defined by the command message **SHOULD NOT** be executed.

If the format of a reply message is determined to be invalid, the data supplied by the reply SHOULD NOT be used as a valid data. In this case the Device Command Layer must simply drop the reply and continue as if no reply was received at all.

Type 2 –

The second type of error checking is the validation service provided by the Device Command Layer for the Data Link Layer. It is described in detail in section 6.8.3. To summarize it, the Data Link Layer may submit some data it has received and request that Device Command Layer perform limited checking on it to ensure that it is valid.

If the data presented by the Data Link Layer represents a command, the Device Command Layer must check whether or not it is possible to construct a reply to this command and indicate the result. The criteria for this evaluation are listed in section 6.8.3.

If the data submitted by the Data Link Layer represents a reply, it must be flagged to the Data Link Layer as valid.

4.5 Error Reply Generation

Whenever the operation requested by the command message is impossible to complete successfully for any reason, an error reply must be generated to indicate such a situation. The error reply should indicate which command message it corresponds to (via STATUS_REGISTER_ADDRESS or TRANSNUM) and specify the reason for operation failure by providing an appropriate error code.

An appropriate error reply must be sent when an illegal or unsupported opcode command is received. For example, if a priority read command is received by a device that does not support priority commands, that device must generate a priority error reply.

As stated before, two different methods are used when sending an error reply. The non-priority commands require error reply from category E, while the priority commands utilize an error reply format similar to the "successful operation" replies from category D.

Priority Error Reply Format

A part of the successful operation execution reply format is used to build a error reply. The error reply format is as follows:

Here OPCODE and STATUS_REGISTER_ADDRESS are the same as in the successful operation execution reply. The STATUS_REGISTER_DATA field for an error reply is built as follows:

Bit 15 should be clear to indicate that this is an error reply.

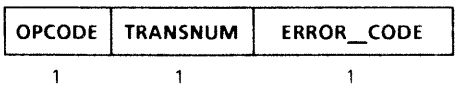
Bit 14 should either be set to indicate that replying device is running, or clear to indicate the replying device is in halt.

Bits 0 - 7 should contain an appropriate error code. See Appendix D for the complete list of error codes. NOTE THAT IN PRIORITY ERROR REPLIES ONLY ODD ERROR CODES CAN BE USED, AS BIT 0 MUST BE SET.

Non-Priority Error Reply Format

This type of error reply is built as follows:

Name: Error reply
Opcode: A2
Category id: E2
Format:



A complete list of error codes is presented in Appendix D.

4.6 Device Command Layer Semantics

Throughout this section of the document the device initiating the operation by sending out the command message is referred to as the 'initiating device'. The device the command is directed at is the 'replying device'.

As stated earlier, there are two steps in operation execution. For each operation execution at the Device Command Layer, successful or not, there is a command message and a reply message. The initiating device sends a command and receives an appropriate reply from the replying device.

The responses shown in Figures 4.3.1 and 4.3.2 are not actually part of the Device Command Layer semantics, rather the responses are part of the Data Link Layer. The responses are shown for the description of section 4.6.1.

As indicated in section 3.3.2, there are at least four transmissions involved in any one communication transaction. Because SY/MAX protocol is implemented as a Full-Duplex protocol, both messages and responses can occur in opposite directions at the same time. This full-duplex "overlapping" is shown by the arrows in Figure 4.2.2. For the purposes of clarity, all diagrams and figures will not show the full-duplex overlap and will be simplified as shown in Figure 4.3.1. Note Figure 4.3.1 does not show the Command and Reply messages with TRANSNUM 255.

The Device Command Layer protocol allows for multiple outstanding command messages. This means that, if desired, the initiating device can send a command message even though the previously sent command message has not received a reply. Thus, the initiating device can send many command messages in sequence and collect the corresponding replies at a later time.

4.6.1 BUSY RESPONSE

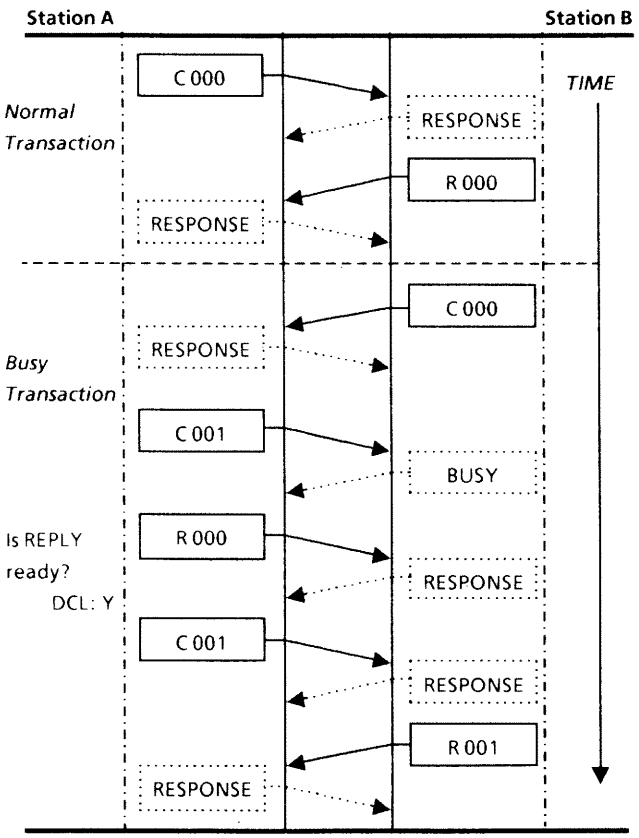
A VERY IMPORTANT responsibility of the Device Command Layer is to handle the situation where the other station is responding that it is too busy to process the incoming data. Although this may seem necessary to be prescribed to be a part of Data Link Layer, it DOES belong in Device Command Layer. The decision tree designated by the SY/MAX P-P protocol on reception of the busy signal requires differentiation between handling a command and a reply message. This information is only available in the Device Command Layer. This means that whenever the Data Link Layer receives a response from the other station indicating the Busy condition, it will return that information to the Device Command Layer (through the Network Layer) indicating that the Data Link Layer was not able to perform the service requested by the Device Command Layer for that specific message operation.

The Device Command Layer determines the proper way to handle the condition described above by applying the following rules:

If a reply message was being transmitted, the Device Command Layer should submit it for retransmission.

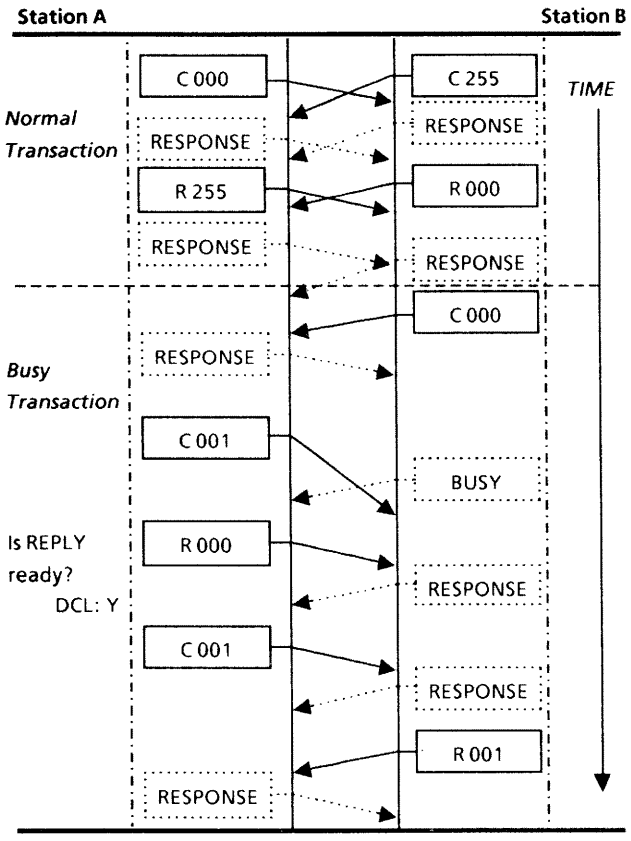
If a command message was being transmitted, the Device Command Layer should check whether there is a reply message ready to be sent. If there is one available, the device should switch back and transmit that reply message. After successful completion of that transmission, the Device Command Layer should return to retransmitting the original command message. If there is no reply message ready to be transmitted, the Device Command Layer should retransmit the original command message.

The protocol of the Device Command Layer provides a maximum value of 256 message retransmissions in response to a BUSY, which when exceeded would lead to an error condition and an error could be flagged to the user application.



C 000 is a Command Message with a TRANSNUM of 000.
R 000 is a Reply Message with a TRANSNUM of 000.
RESPONSE is a positive acknowledgement.
BUSY is the busy response.

Figure 4.3.1 Device Command Layer Semantics (Simplified)



C 000 is a Command Message with a TRANSNUM of 000.
R 000 is a Reply Message with a TRANSNUM of 000.
RESPONSE is a positive acknowledgement.
BUSY is the busy response.

Figure 4.3.2 Device Command Layer Semantics (Full-Duplex Overlapping)

To describe the remaining rules of the Device Command Layer protocol, a distinction must be made between the priority messages and non-priority (all other) messages.

4.6.2 NON-PRIORITY OPERATIONS SEMANTICS

In non-priority operations the TRANSNUM field is used in the command and reply messages to control the flow of information between the two devices and to match the replies with commands. This field contains a value 0 through 255.

Initiating Device Responsibilities

If the initiating device needs to send multiple outstanding commands, it first has to insure that the lower protocol layers have successfully completed all the transactions required of them for the current command message transaction. Then the Device Command Layer protocol handler can start sending out another command message marked with a new unique TRANSNUM. Thus, the initiating device can have up to 256 outstanding command messages. Of course, there should not be two outstanding command messages with the same TRANSNUM. The typical method of producing a unique TRANSNUM is to increment it with each command message which has been successfully processed by the lower layers, so all consecutive command messages are sequentially numbered.

In the one-command-message-at-a-time mode the initiating device sends one command message, and waits to send the next command message until after the reply to the first command message has been received. Even in this mode it is necessary for the initiating device to use a unique TRANSNUM for each command.

When the initiating device receives a non-priority reply, it is responsible for matching the reply with the corresponding command using the TRANSNUM.

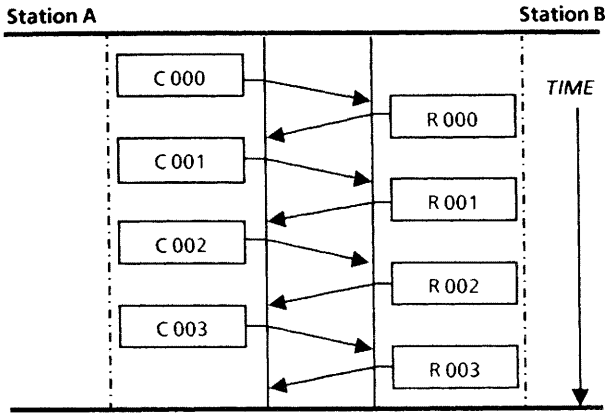
Replying Device Responsibilities

When constructing the reply to a non-priority command, the replying device is responsible for using the same TRANSNUM as the one used in command message. It is not responsible for verifying that the TRANSNUM is unique from message to message.

The sequence of Figures 4.4, 4.5 and 4.6 illustrate possible transaction sequences between two stations A and B. Each message transfer (command or reply) is assumed to be performed successfully by the lower protocol layers and is marked with a unique TRANSNUM. Note that a device CAN NOT start sending a command before previous command's transmission has been completed (as indicated by the lower protocol layers).

Figure 4.4 illustrates a typical command/reply exchange between A and B where A initiates all commands and B replies.

Scenario: Station A sends the following sequence of messages, one at a time, and receives the corresponding replies.

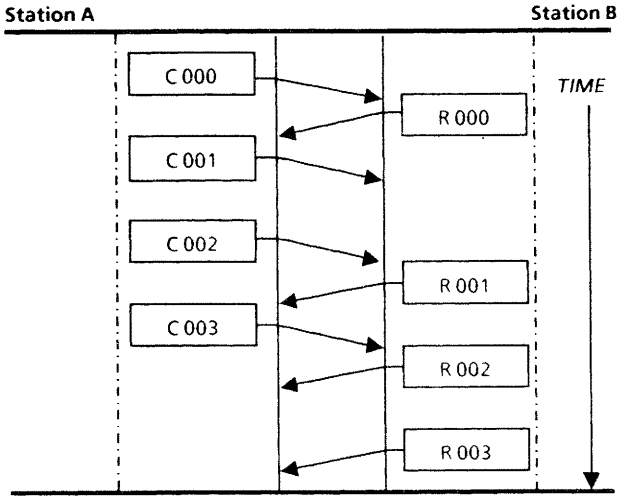


C 000 is a Command Message with a TRANSNUM of 000.
R 000 is a Reply Message with a TRANSNUM of 000.

Figure 4.4 Single Outstanding Message. Station A initiates all communications.

Figure 4.5 illustrates a command/reply exchange between A and B where A initiates all commands and B replies.

Scenario: Station A sends the following messages and collects the corresponding replies. Note that A station is not waiting for the reply for the previous command before proceeding with the next one.

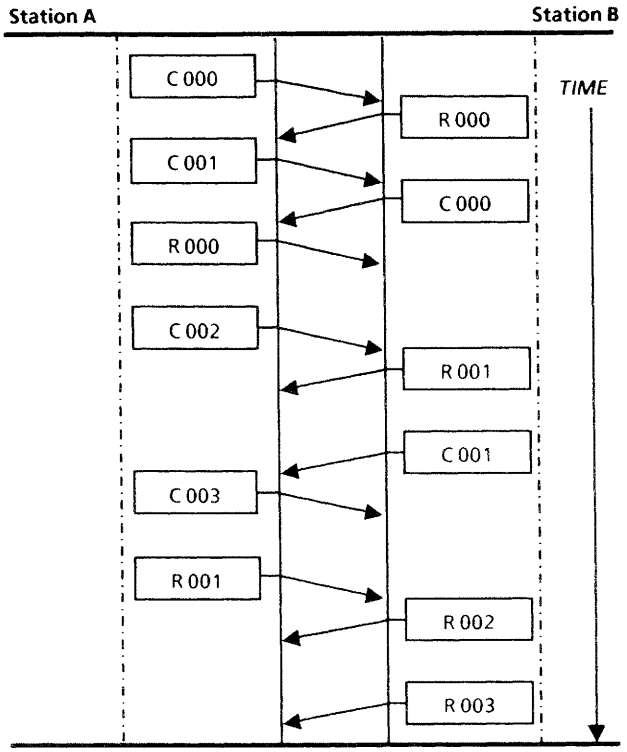


C 000 is a Command Message with a TRANSNUM of 000.
R 000 is a Reply Message with a TRANSNUM of 000.

Figure 4.5 Multiple Outstanding Message. Station A initiates all communications.

Figure 4.6 illustrates a command/reply exchange between A and B where both A and B initiate commands.

Scenario: Both stations send command messages and collect the corresponding replies. Note that both devices are capable of transmitting at the same time due to the full duplex nature of the lower layers of the protocol.



C 000 is a Command Message with a TRANSNUM of 000
R 000 is a Reply Message with a TRANSNUM of 000.

Figure 4.6 Multiple Outstanding Message. Both stations initiate communications.

4.6.3 PRIORITY OPERATIONS SEMANTICS

Priority messages use the STATUS_REGISTER_ADDRESS field in place of the TRANSNUM. As each outstanding priority command message **should** have a unique STATUS_REGISTER_ADDRESS, this field serves the same function. If two outstanding priority commands do have the same STATUS_REGISTER_ADDRESS, the first reply is assumed to correspond to the first command.

4.6.4 PRIORITY VS. NON-PRIORITY OPERATIONS

In general, when a device has to service both priority and non-priority messages, priority messages are serviced first. However, some mechanism must be provided to insure that eventually the non-priority message is serviced, even if there are some priority messages queued. Otherwise, it is possible that a non-priority message could potentially never be operated on.

4.7 Device Command / Network Layer Interface

All of the data assembled by the Device Command Layer, including OPCODE, OPCODE related data, and TRANSNUM, are passed down to Data Link Layer (through the Network Layer). The representation of all these items in the Data Link Layer is the symbol <data> in Figure 4.1.

This page intentionally left blank.

5 NETWORK LAYER

5.1 Introduction

The Network Layer interfaces with Device Command and Data Link Layers. Typically, its responsibility is to handle routing information by providing the complete path from source device to destination device through all the intermediate points involved in the communication. Although SY/MAX P-P protocol involves just two stations communicating with each other, in the context of SY/MAX P-P as the access point to SY/NET Network communications, complete routing through the network has to be considered. Thus, in the definition of SY/MAX P-P protocol, the Network Layer provides the complete path through all necessary intermediate communication drops to perform the communications between end devices. Figure 5.1 depicts the data handling responsibilities of this layer.

The complete route consists of a list of drops. The list can be empty or can consist of up to 8 drops.

5.2 Responsibilities

The responsibilities of the Network Layer will vary depending on whether it is participating in origination or reception of the message, and on whether it is a command or a reply message.

5.2.1 COMMAND MESSAGE RESPONSIBILITIES

5.2.1.1 Sending A Command

Depending on the user application, the complete route information may be provided in one of the two ways:

1. The upper layers may provide the complete route information, or

2. The upper layers identify only the intended destination of the message. The corresponding complete route may then be determined by the Network Layer. This may be accomplished by referencing a predefined table.

The complete route information is transformed into a data representation needed by the Data Link Layer. The data transformation of the routing information is performed in the Network Layer and is passed down to the Data Link Layer. To present routing information in the proper form to the Data Link Layer, each drop is converted to a two digit hex value which is ASCII coded. All the drops are listed in sequence in that form.

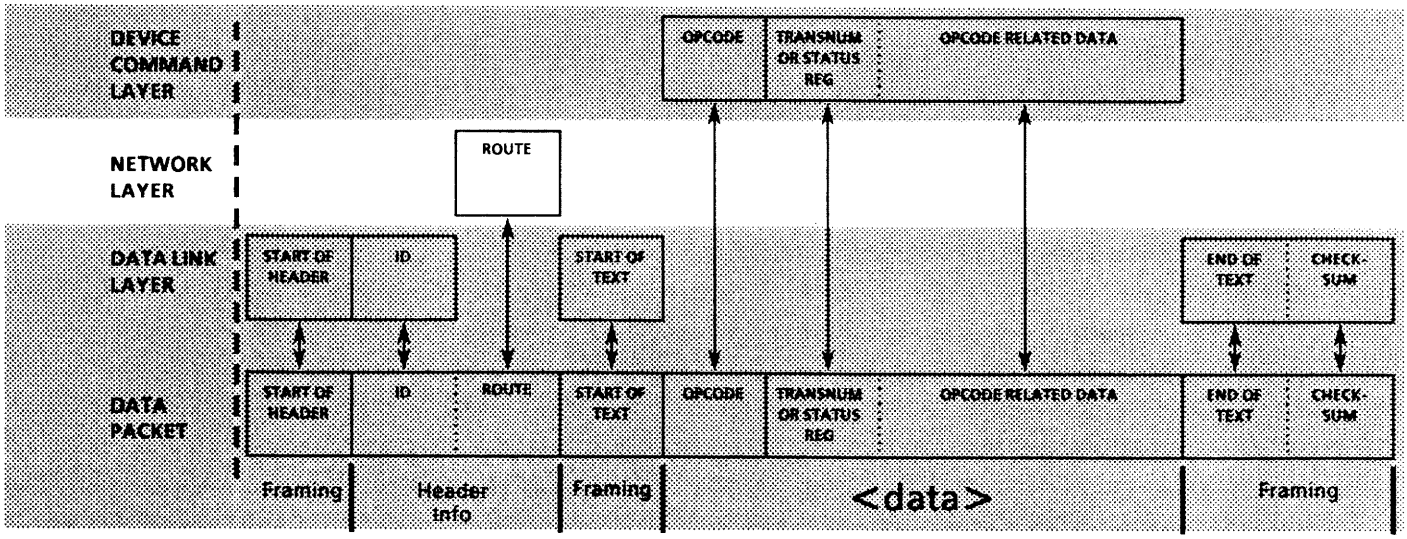


Figure 5.1 - Network Layer Data Handling

Example: *Route of 3 drops : 100, 14, 81*

Drop	#1	#2	#3
Decimal	100	14	81
Hex	64	0E	51
ASCII Hex	36 34	30 45	35 31

The format of the data representing the route information is reviewed in the data link protocol syntax definition. The name of the routing element in the Data Link Layer definition is <route>. See section 6.7 for more information.

It is important to remember that the format of the routing data is repeated in section 6.7 for the purposes of discussion only and to illustrate the possibility of an embedding response in the route field. It is the recommended that the Network Layer perform all necessary data transformations of the routing information before passing it down to the Data Link Layer.

5.2.1.2 Receiving A Command

The Network Layer of the device receiving the command message performs two types of error checking on the route field:

1. Check route reversibility as a service to the Data Link Layer

2. Validate route data as a service to the Device Command Layer.

Route reversibility check.

The route is said to be reversible if it is between 0 and 16 bytes long and it consists of an even number of bytes. If the route is not reversible, the Network Layer must notify the Data Link Layer, which will then negatively acknowledge the command message that the route came with.

Route data validation.

The route data is considered to be valid if:

1. The Route field does not consist of only one drop, and

2. Each byte in the route field is either between 30 and 39 hex, or between 41 and 46 hex.

If the route is not valid, the command message **SHOULD NOT** be acted upon. The Device Command Layer must generate an error reply with the route reversed and the error code decimal 29, signaling an "Attempt to send message with illegal route".

5.2.2 REPLY MESSAGE RESPONSIBILITIES

5.2.2.1 Sending A Reply

The Network Layer is responsible for reversing the order of the drops in the route field from the corresponding command message before passing the route down to the Data Link Layer. As a result of the route reversal, the last drop of the command's route becomes the first in the reply's route, the second to last becomes the second and so on.

Example: *Command's route of 3 drops: 100, 14, 81*

Resulting Reply's route: 81, 14, 100

After appropriate data transformations:

Drop	#1	#2	#3
Decimal	81	14	100
Hex	51	0E	64
ASCII Hex	35 31	30 45	36 34

5.2.2.2 Receiving A Reply

The Network Layer of the device receiving a reply message will perform the route reversibility checking of the route field, but it will not validate the route data. The reversibility check is performed as a regular service to the Data Link Layer, because the Data Link Layer does not distinguish between reply and command messages. The Device Command Layer determines whether a message is a command or a reply, and will not request the route data validation service from the Network Layer when a reply is received.

6 DATA LINK LAYER

6.1 Introduction

This section presents a complete definition of the Data Link Layer of the SY/MAX P-P protocol. This layer of the protocol is a variation of the ANSI X3.28 protocol standard, and Appendix B explores the similarities and differences of both.

The Data Link Layer (DLL) is responsible for the complete and error-free transfer of the binary messages across the communication link using the physical layer as the medium. This is accomplished by framing the messages, handling the procedural rules, and performing the error detection and correction activities. The actual contents and the meaning of the messages are of no concern to this layer. The message type also has no effect on the operation of the layer. This layer interfaces with Network and Physical Layers. Figure 6.1 depicts the data handling responsibilities of this layer.

6.2 Data Link Protocol Characteristics

The primary data transfer characteristics of the Data Link Layer of the SY/MAX Point-to-Point protocol include: FULL DUPLEX OPERATION, EMBEDDED RESPONSE CAPABILITY, ALTERNATE ACKNOWLEDGMENTS, TRANSPARENT HEADER AND DATA AREAS, AND MESSAGE-ASSOCIATED BLOCKING WITH BLOCK CHECK CHARACTER FOR ERROR DETECTION.

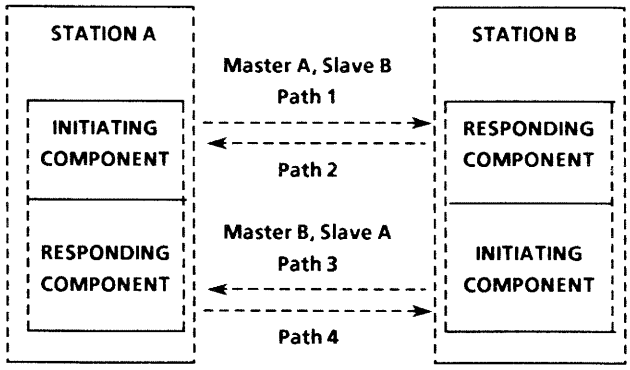


Figure 6.2 - Logical Data Paths in Point-to-Point Connection

The logical data paths shown in Figure 6.2 allow for two-way simultaneous operation, one of the most important features of the protocol. Each station must have the capability of maintaining concurrent master and slave status, that is, master status on the transmit side and slave status on the receive side. The receiver is always ready to receive data, while the transmitter is only active when necessary. The initiating component is the conceptual part of the device that originates and controls the message transfer. The responding component works with the initiating component of the other device to complete a successful message transfer. The data paths demonstrated in Figure 6.2 are only LOGICAL, not physical entities and are actually implemented in the Physical Layer with only two physical circuits. However, it is important to understand that there are 4 logical paths needed to accommodate the two-way simultaneous operation.

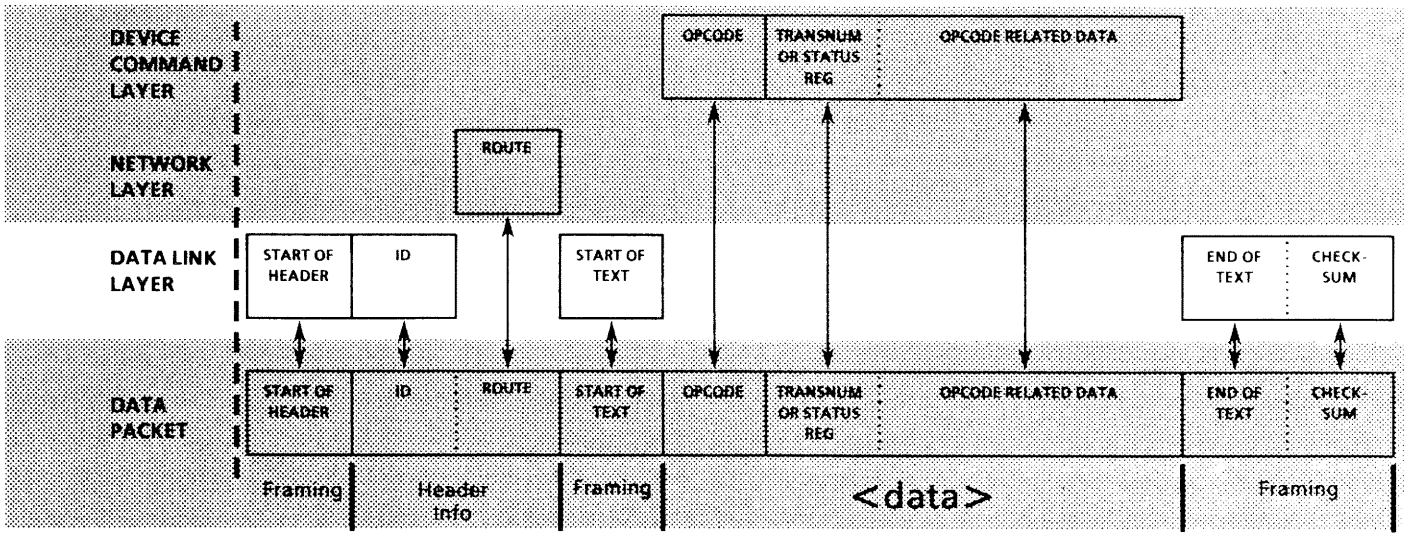


Figure 6.1 - Data Link Layer Data Handling

Path 1 carries message information from A to B; path 2 carries response type of information back from B to A. Path 3 carries messages from B to A; path 4 carries responses from A back to B. To implement these four paths with two physical wires, as defined in the Physical Layer, a software multiplexing mechanism is employed on the one end of the link to embed or interleave responses with message information flowing in the same direction. The demultiplexing software at the other station separates the responses from the messages. Thus on each physical circuit, response type of information is mixed with messages. Multiplexing and demultiplexing concepts are illustrated in Figure 6.3. Since this is a Full Duplex protocol, everything stated about the A to B message transfer using paths 1 and 2 can be applied equally to the B to A message transfer over paths 3 and 4. The two initiating/responding component pairs can be looked upon as two independent subsystems operating under the same protocol. One subsystem is responsible for A to B message transfer, the other is concerned with B to A message communication. There is some interaction between the two because they have to share the same physical circuit. This is accomplished via the multiplexing/demultiplexing of message and response information.

While the syntax part of the protocol definition does accommodate message transfer for both A to B and B to A, the procedural rules are demonstrated for the "master A - slave B" relationship. Note that everything stated about the "master A - slave B" relationship can be unconditionally applied to, and be valid for, the "master B - slave A" relationship.

6.3 Network – Data Link Interface

When assembling a message, the following information is passed down to the Data Link Layer from the Network Layer: route description, opcode, TRANSNUM (or STATUS_REG), and opcode related data when receiving a message, the same information is delivered to the Network Layer by the Data Link Layer. The route description is used to build the ROUTE field of the Data Link's syntax called ROUTE <data_frame>. The opcode, TRANSNUM (or STATUS_REG) and opcode related data are used to produce the <data> field of the <data_frame>.

6.4 Data Link – Physical Interface

The data exchange between the Data Link and Physical Layers is a packet of bytes. The format of the packet is shown in Figure 6.1.

6.5 Data Link Responsibilities and Functions

The Data Link Layer is not concerned with the types, contents or meaning of a message, only the successful transfer of the message over the physical medium. The Data Link Layer performs two major functions to insure error-free transactions:

1. Assembling/Disassembling message packets called <data_frame>s.

2. Generation/Transmission and Reception/ Interpretation of link control packets called <control_frame>s.

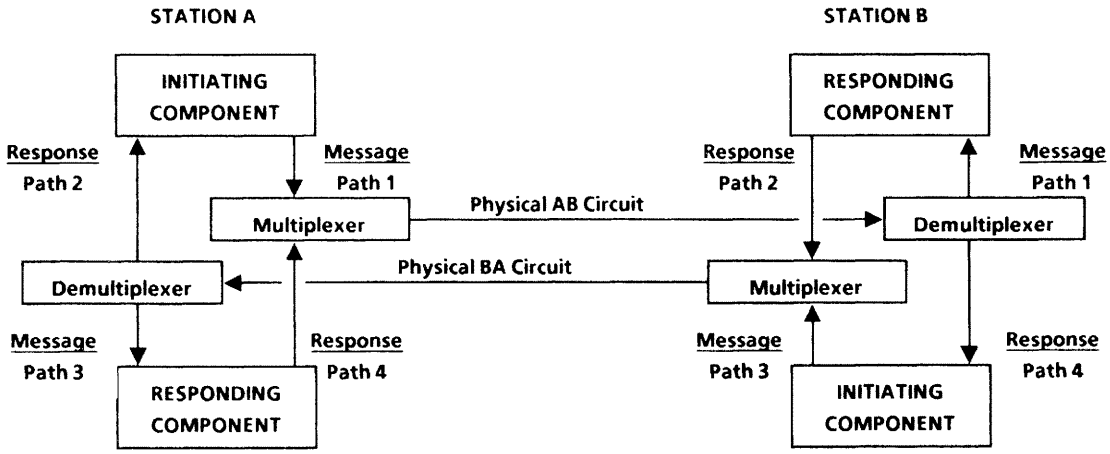


Figure 6.3 Message/Response Multiplexing/Demultiplexing for Full Duplex Communications

6.5.1 Message Framing

The DLL is responsible for producing a data frame by surrounding the route description and the opcode, TRANSNUM (or STATUS_REG) and opcode related data with framing elements. This activity is referred to as framing and can be facilitated either by constructing an equivalent representation of the frame using data structures, or sequentially by following the transitions on the DLL Transmitter state diagram. The binary information contained in the data frame form is sent out to the Physical Layer byte-by-byte. While the transmission of the byte stream is taking place, the sequence of the bytes in the data frame is preserved with the following exception: additional <control_frame>s may be inserted in the middle of the <data _frame>. The nature of the embedded control frames are discussed in Section 6.7.11.

6.5.2 Control Frames

The DLL is responsible for the error-free transmission of the data frame; thus its duties include generation/transmission as well as reception/interpretation of the <control_frame>s. Control frames are the means for error detection and correction and can be one of two types: interleaved control frames, occurring between data frames or embedded control frames, inserted into a data frame. Embedded control frames are equivalent to interleaved control frames in every respect; the only difference being their location.

6.6 Data Link Protocol Definition

The following definition of this layer has two elements: syntax and semantics (procedural rules). To enhance the comprehension of the vast amount of information presented in this section, the syntax is defined first, followed by the semantics of the protocol.

6.7 Data Link Protocol Syntax

Figures 6.4 through 6.6 define the syntax of the Data Link protocol in a formal grammar fashion similar to Backus Normal Form (BNF), which is typically used in computer science to represent language syntax. This definition corresponds to the binary form of data when the framing of the message has already occurred. Since the control frames are demonstrated here as well, this syntax definition can be thought of as a structured and organized depiction of binary information exchange between two stations AFTER the multiplexing process described above has taken place. Thus, it DOES include definition of the embedded responses. Following is the key for Figures 6.4 through 6.6:

:=	means	"is composed of"
< >	marks	a variable element
<a>	means	"a is a required element"
<a><b>	denotes	"b follows a"
<a>|<b>	means	"either a or b"
...	means	"and so on up to"
Ë	denotes	"empty element" or "an element does not exist"

symbols not enclosed by < > are hex constants.

Element	Definition	Section
<frame> : =	<data_frame> | <control_frame>	6.7.2
<data_frame> : =	<start_header><header><start_text><data><end_text><checksum><pad_seq>	6.7.3
<control_frame> : =	<interleaved_response> | <inquiry>	6.7.10
<start_header> : =	<dle><soh> <embedded_response>	6.7.4
<header> : =	<id><route>	6.7.5
<start_text> : =	<dle><stx> <embedded_response>	6.7.6
<data> : =	<data> | <data_unit> <data_unit>	6.7.7
<end_text> : =	<dle><etx>	6.7.8
<checksum> : =	00H | 01H | 02H | ... |FFH	6.7.9
<pad_seq> : =	<pad> | <pad_seq><pad>	6.7.17
<interleaved_response> : =	<response>	6.7.12
<embedded_response> : =	<response> | Ë	6.7.11
<inquiry> : =	<dle><enq>	6.7.16
<id> : =	<odd><embedded_response>< | <even><embedded_response>	6.7.5
<route> : =	Ë | <drop> | <drop><drop> | ...	6.7.18
	<drop><drop><drop><drop><drop><drop><drop><drop>	
<data_unit> : =	00H <embedded_response> |01H	6.7.7
	<embedded_response> |	
	02H <embedded_response> | ... |0FH	
	<embedded_response> |	
	<dle> 10H <embedded_response> |11H	
	<embedded_response> |	
	12H <embedded_response> | ... |FFH	
	<embedded_response>	
<response> : =	<positive_ack> | <negative_ack> | <busy>	6.7.12
<drop> : =	<hex_digit><embedded_response><hex_digit><embedded_response>	6.7.18
<positive_ack> : =	<dle><odd> | <dle><even>	6.7.13
<negative_ack> : =	<dle><nak>	6.7.14
<busy> : =	<dle><syn>	6.7.15
<dle> : =	10H	
<soh> : =	01H	
<stx> : =	02H	
<etx> : =	03H	
<pad> : =	FEH	
<enq> : =	05H	
<odd> : =	11H	
<even> : =	12H	
<hex_digit> : =	30H | 31H | 32H | ... | 39H | 41H | 42H | 43H | 44H | 45H | 46H	
<nak> : =	15H	
<syn> : =	16H	

Figure 6.4 Data Link Layer Syntax Table

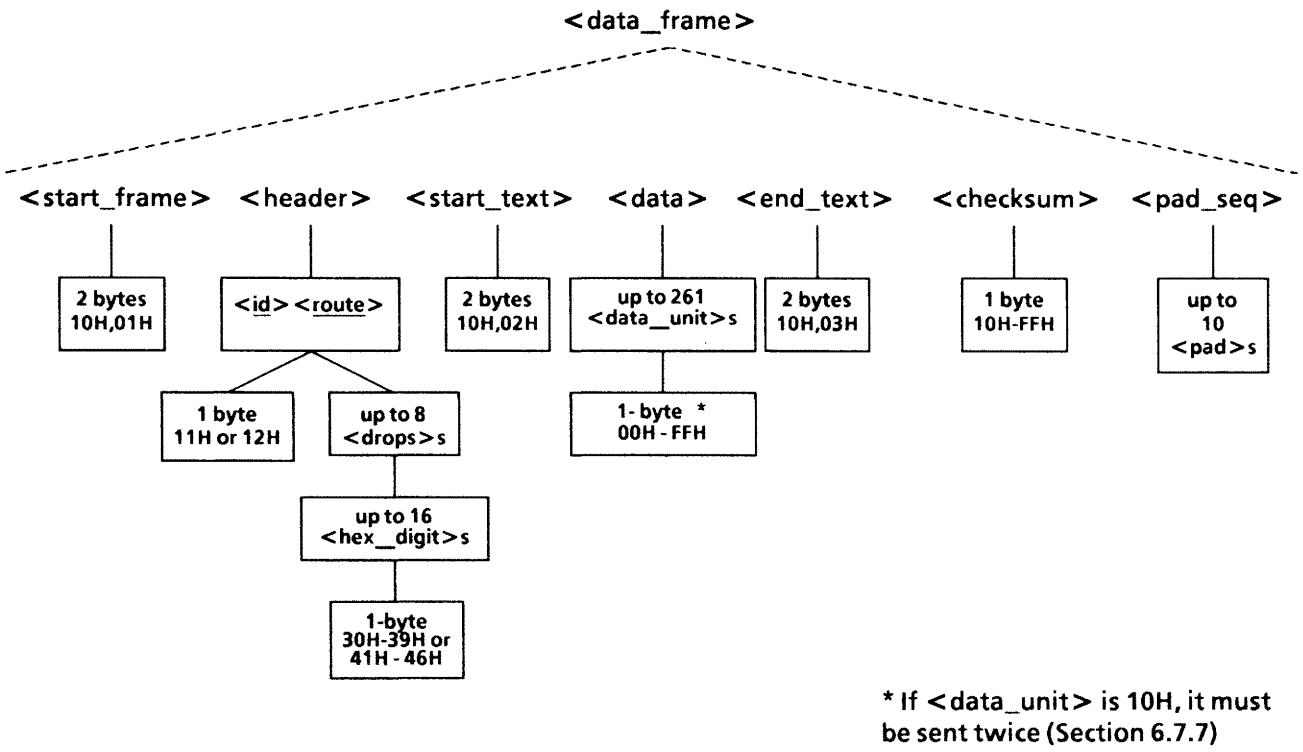


Figure 6.5 Data Frame Syntax (locations for embedded responses are not shown)

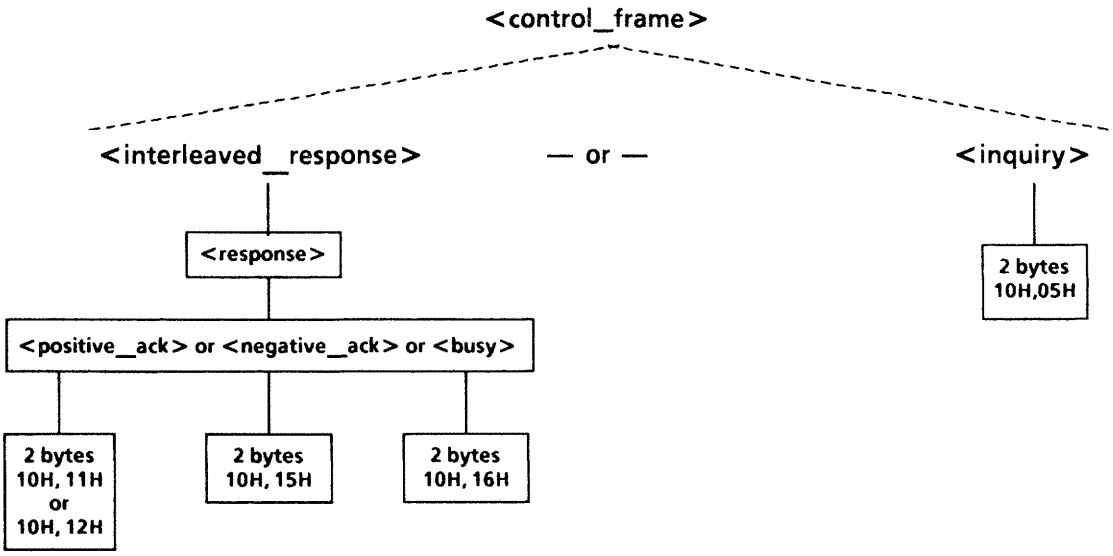


Figure 6.6 Control Frame Syntax

The following is a discussion of the syntax of the Data Link protocol; it clarifies and explores in more detail the information presented above in Figures 6.4 through 6.6. It also builds the terminology used throughout the remainder of the document.

6.7.1 FRAME EXCHANGE

An error-free transaction between two communicating stations (devices) involves an exchange of four or more <frame>s. Procedural rules for frame exchange are described in Section 6.8.

6.7.2 FRAMES

A frame can be of two types: <data_frame> or <control_frame>. The transaction sequence starts with a station sending out a <data_frame>, which includes the information in the form of <data> that ultimately needs to be transferred from one station to another. <control_frames> are used for link establishment error detection and correction.

6.7.3 DATA FRAME

The <data_frame> contains the actual data that needs to be transferred; all other components of the <data_frame> (except for <embedded_response>s) are referred to as "framing components" or "framing". Framing components serve the sole purpose of preserving the integrity of the data which is accomplished through error detection, correction and recovery methods. Note that although an <embedded_response> could be a part of the <data_frame> physically, logically it has nothing to do with facilitating the successful transmission of the <data> field of the <data_frame> it is embedded in. Insertion of the <embedded_response>s into the <data_frame>, although physically performed at the transmitter, is logically triggered by the events happening at the receiver.

6.7.4 START OF HEADER

A <start_header> is used to indicate the start of a <data_frame> and the header area. Note that an <embedded_response> may occur immediately after this component.

6.7.5 HEADER

Information in the <header> includes the <id> (alternating between <odd> and <even> with every successful data transmission of the <data_frame>) and a <route>.

6.7.6 START OF TEXT

A <start_text> is used to delimit the header and data areas.

6.7.7 DATA

Successful transfer of the <data> field is the primary purpose of the Data Link protocol. The <data> is passed to the Data Link Layer from the Network Layer during transmission and from the Data Link Layer to the Network Layer during reception. As stated previously, the Data Link protocol has no concern for contents of the <data> field, its function in the operation of higher levels of the protocol, or the ultimate purpose of the data.

Note that a <data_unit> can have a value of 10 hex which is a <dle>. In order to prevent that <data_unit> from being interpreted as a part of control frame, an extra <dle> must be inserted prior to the data_unit. The extra <dle> is a part of the <data> and must be taken into account when calculating the checksum value.

6.7.8 END OF TEXT

An <end_text> indicates that end of data was reached and the next field is the <checksum>.

6.7.9 CHECKSUM

The value of the <checksum> is computed as the two's complement of the eight bit sum of all components of the data_frame from <start_header> (not including <start_header>) up to and including <end_text>. Note: Any <embedded_response>s ARE NOT counted in the calculation. Note that any extra <dle> that are required in the <data> field ARE added to the checksum (Section 6.7.7).

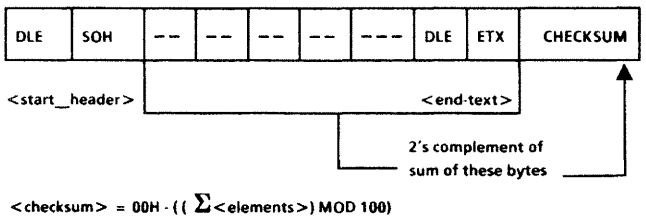


Figure 6.7 Bytes Used for Calculating Checksum

6.7.10 CONTROL FRAME

A <control_frame> can be either an <interleaved_response> or an <inquiry>.

6.7.11 EMBEDDED RESPONSE

To accommodate for the <embedded_response> definition, the Ë symbol is used to indicate the possibility of the <embedded_response> being empty (non-existant). The syntax rules allow for 0 or more <embedded_response>s in a <data_frame>. According to the definition presented above, an <embedded_response> can occur anywhere in a <data_frame> except for the following two locations:

(a) between <dle> and the following byte

(b) between <dle><etx> and <checksum>

Embedded responses cannot occur in a <control_frame>.

6.7.12 USE OF RESPONSES

Both <embedded_response>s and <interleaved_response>s are used by the receiving station to accomplish one of the following:

(a) Positively acknowledge (<positive_ack>) the <data_frame> received thus indicating successful reception of the <data_frame>, or

(b) During or immediately after the reception, negatively acknowledge (<negative_ack>) the <data_frame> indicating that an error occurrence was detected, or

(c) While receiving the data_frame, indicate to the sending device that the receiving station is busy (<busy>) and thus TEMPORARILY cannot proceed with reception.

The procedural rules governing the use of responses are addressed in Section 6.8.

6.7.13 POSITIVE ACKNOWLEDGMENT

To positively acknowledge a <data_frame> reception, either the <dle><odd> or <dle><even> sequence can be used. Which response to use is determined by the <id> of the <data_frame> whose reception is being acknowledged. Thus if last received <data_frame>'s <id> is <odd>, <dle><odd> is used; if it is <even>, <dle><even> is employed.

6.7.14 NEGATIVE ACKNOWLEDGMENT

For negative acknowledgment the <dle><nak> sequence is used. It is typically sent when an error is detected by the station while receiving an incoming <data_frame>.

6.7.15 BUSY

To indicate that the receiver is currently unable to process incoming data, the <busy> sequence of <dle><syn> is used. This sequence may be employed to indicate situations such as buffer space deficiency or processing time shortage.

6.7.16 INQUIRY

Inquiries are used by a station to invoke the other station to repeat its last <response>. The <dle><enq> sequence is used when a transmitting station did not receive an appropriate <response> to a <data_frame> within a certain period of time (Section 6.8.5).This sequence can also be used when the communication line is idle to determine whether the other device is operational. For more details on procedural rules governing the use of inquiries, refer to sections of 6.8.5 through 6.8.7.

6.7.17 PAD SEQUENCE

At least one <pad> character must be sent out after the <checksum> for resynchronizing the asynchronous data stream in the event that a framing error should occur.

6.7.18 ROUTE

A non-empty <route> field in the header area can list up to a maximum of 8 drops on. Each drop is represented as a sequence of two <hex_digit>s. Note that an <embedded_response> may occur in the route field. Route field information is exchanged back and forth through the Data Link Layer/Network Layer interface. Although the format of the route information is presented in Data Link Layer, it is a function of the Network Layer to perform those transformations. Refer to Section 5 for more information.

6.8 Data Link Layer Semantics

6.8.1 INTRODUCTION

The semantics (procedural rules) of the Data Link Layer of the SY/MAX Point-to-Point protocol are presented in this section. First, each type of response and inquiry is described in terms of its meaning, use, and expected outcome. It is important to remember that, although reception of responses is handled by the receiver of the master station, it is logically triggered by the operational needs of the master station's transmitter. The multiplexing/demultiplexing mechanisms provide the logical links between the transmitter and receiver.

Secondly, in order to keep the definition of the Data Link protocol as structured as possible, two distinct phases of the data communication at this level are defined: link establishment and data transfer. There is no link termination phase defined in SY/MAX protocol.

Note that for the purposes of the following discussion an outstanding <data_frame> is a complete <data_frame> which has been sent by the master station and the reception of which has not yet been positively acknowledged by the slave station.

Data Link Layer semantics of the master station are summarized in Figure 6.8.

6.8.2 POSITIVE ACKNOWLEDGMENT RESPONSE OPERATION

There are two positive acknowledgment responses in SY/MAX P-P protocol: <dle><odd> and <dle><even>. The correct positive acknowledgment for a <data_frame> depends on the <id> field within the <data_frame>. If the <id> is <even>, then the correct positive acknowledgment is <dle><even>. If the <id> is <odd>, then the <positive_ack> is <odd>.

Transmission

These response sequences should only be sent in two situations:

(a) to positively acknowledge completion of the error-free reception of a <data_frame>. The <odd >or <even> positive acknowledgment is determined by the <id> field of the <data_frame> received.

(b) as a response to an incoming inquiry (see section 6.8.5)

Reception

Interpretation of the received positive acknowledgment responses is accomplished as follows:

(a) If there is an outstanding <data_frame> which has not yet been positively acknowledged and the type of the positive acknowledgment DOES MATCH the <id> of that <data_frame>, then the received positive acknowledgment confirms an error-free reception of the <data_frame> by the other station. This signals that a complete and successful transmission of that <data_frame> has occurred. When the <positive_ack> is received, the Data Link Layer function and the transfer of the <data_frame> are complete.

(b) If there is an outstanding <data_frame> which has not yet been positively acknowledged and the type of the positive acknowledgment DOES NOT MATCH the <id> of that <data_frame>, then this positive acknowledgment should be treated as if it were a negative acknowledgment. (See section 6.8.3).

(c) If there are no outstanding <data_frames> that the received positive acknowledgment could be corresponding to, then it should be ignored completely.

6.8.3 NEGATIVE ACKNOWLEDGMENT RESPONSE OPERATION

Negative acknowledgment sequence <dle><nak> is used by the slave station to signal an error.

All errors signaled in the Data Link layer by negative acknowledgment can be divided into 2 distinct categories: <data_frame> errors and byte errors.

A byte error is defined as a failure to receive correctly a byte of the <data_frame>. The byte errors are usually detected by the hardware (UART) in the Physical Layer:

(a) parity error
(b) framing error
(c) overrun error

The <data_frame> errors indicate a fault while receiving a data_frame as a integral unit. These errors include:

(a) Checksum error. This type of an error occurs when the checksum calculated for the <data_frame> by the receiver does not match the <checksum> value sent with the <data_frame>. (Note that additional or missing bytes of 00H will not be detected by this method.)

(b) Illegal dle sequence. This error occurs when an illegal code is found after the <dle> character inside the <data_frame>. The valid codes are: <odd>, <even>, <stx>, <soh>, <etx>.

(c) Incorrect framing components sequence. This type of an error must be flagged when the framing components of the <data_frame> do not follow the correct syntax or sequence.

(d) Data_frame too large. This error is generated when the incoming <data_frame> exceeds the maximum length allowed by the protocol. This maximum value is dictated by the longest message of the Device Command Layer of the protocol. Maximum length without <embedded_responses> is 295 bytes.

(e) Unrepliable command data_frame. During the reception, the Data Link Layer can not positively acknowledge the <data_frame> it has collected until the Device Command Layer ensures that, the command message is repliable. It is considered to be repliable if one of the following is true:

1. It is a priority command with <data> field at least 3 bytes long (including the opcode).

OPCODE	STATUS_REGISTER_ADDRESS

2. It is a non-priority command with <data> field at least 2 bytes long (including the opcode).

OPCODE	TRANSNUM

Note that this error can be generated only for the <data_frames> carrying command messages.

(f) Irreversible route field. During the reception, the Data Link Layer can not positively acknowledge the <data_frame> it has collected until Network Layer verifies that the <route> field (if not empty) is reversible. The <route> is considered to be reversible if it consists of even number of bytes and the number of bytes is between 0 and 16. If the route is not reversible, the Network Layer must indicate to the Data Link Layer to send the negative acknowledgment in response to the <data_frame>.

The receiving station must indicate that occurrence of any one of these errors has been detected by sending out the <dle><nak> sequence.

Transmission

A negative acknowledgment sequence should be transmitted only under the following conditions:

(a) if an error was detected in the reception of the <data_frame>. It can be sent during or after the reception of the incoming <data_frame>. For example, if an error was detected in the middle of the incoming <data_frame>, it is not necessary to wait for the end of the <data_frame>: it SHOULD be sent out as soon as the error is detected.

(b) in response to an incoming inquiry. (Section 6.8.5)

Reception
The master station's interpretation of the received negative acknowledgment should proceed as follows:

(a) If there is a corresponding outstanding <data_frame>, its transmission should be repeated, unless transmission of this specific <data_frame> has already been retried 8 times, in which case a channel error is posted to the upper layer. The number of times the transmission of any specific <data_frame> has been retried is referred to as the retry count. Similarly, if the negative acknowledgment was received prior to the end of the transmission of the <data_frame> being currently sent out, the current transmission SHOULD be terminated immediately and the retransmission procedure, as described above, should be invoked. The retransmission of the <data_frame> should not included the retransmission of any <embedded_response>s .

(b) If there are no outstanding <data_frames> that the received negative acknowledgment could be corresponding to, then it should be ignored completely.

6.8.4 BUSY RESPONSE FUNCTION

The busy response <dle><syn> is used by the slave station to indicate that it is currently busy or does not have necessary resources to process the transmissions sent to it by the master station.

Transmission
A slave device sends out the busy response whenever it needs to inform the master device that it can not handle the incoming <data_frame>. For example, if the slave station is receiving a command <data_frame> and the buffer used during the reception is the last buffer the device has available, this <data_frame> should not be accepted. Instead, the slave station must generate the <busy> response.

Note that the <busy> response SHOULD NOT be sent in response to a master station's inquiry sequence.

Reception
The master station interprets an incoming busy response as follows:

(a) Suppose the response was sent as a result of the master's inquiry. This should never happen because of rule stated about <busy> being an illegal response to an <inquiry>. However, it is still possible for this situation to develop with some of the older devices implementing SY/MAX protocol. When a master station receives a <busy> in response to an inquiry, it should ignore the response.

(b) If the master was in the process of transmitting a <data_frame>, a response of busy from the slave device should be flagged to the Device Command Layer (through the Network Layer). The Device Command Layer will determine the appropriate course of action to be taken. As far as the Data Link Layer is concerned, it has not been able to complete the operation requested by the upper layer as related to this specific <data_frame>.

6.8.5 INQUIRY

The inquiry sequence <dle><enq> is used by the master device to trigger the slave to repeat its last response.

Transmission
An inquiry is sent under following conditions:

(a) No response has been received to an outstanding <data_frame> after waiting for a minimum of 10 (ten) character times. When the response has not been received, the inquiry sequence is repeated every 10 character times, up to the maximum of 32 repeats. If the response still has not been received within 32 inquires, error 17, "Remote Device Inactive" is posted. Note that the <pad> character be sent 10 times to act as a timer.

(b) If the communication line is idle, an inquiry is sent to determine whether the other device is operational. This must be done upon power-up of the master when link establishment phase has to be completed. See sections 6.8.6 and 6.8.7 for more information.

Reception

Upon reception of an inquiry, the slave station should repeat its last response. The valid responses to the inquiry are as follows:

(a) positive acknowledgment
<dle> <odd> or <dle><even>

(b) negative acknowledgment
<dle> <nak>

Note that sending a <busy> in response to an <inquiry> is a violation of the protocol.

6.8.6 LINK VERIFICATION DUTIES

(The Link Verification is an optional procedure in SY/MAX P-P protocol. It's use is recommended to insure that critical communication links are active, and all receivers must be able to respond to the periodic inquiry sequences.) Link verification must be performed every 2 seconds to ensure that the link is still operational and to detect potential disconnection of the other station.

Link Verification is accomplished by the master station sending either a <data_frame> or an <inquiry>, and receiving the appropriate <response>:

(a) positive acknowledgment
<dle> <odd> or <dle><even>

(b) negative acknowledgment
<dle> <nak>

Note that the rules on reception of the above sequences by the master station, stated in the sections 6.8.2 and 6.8.3, apply here as well.

The response to an inquire must be saved by the master device. The next <data_frame> sent by the master must have the <id> field set to the OPPOSITE of the last received response. The <id> should be set to <odd> if the last response to an inquiry was a < negative_ack>.

If no valid response was received from the slave station, the link between two stations is considered to be lost and non-operational. Even so, the link verification activities must continue. Now, however, the device should only send out a maximum of 4 enquire sequences instead of 32. Note that in this case, the device should retain its own last response to be whatever it was prior to the detection of the loss of the link; IT SHOULD NOT BE CHANGED.

If a valid response has been received, the link is considered operational.

Note that Link Verification activities should be performed every 2 seconds whether the link is operational (and idle) or non-operational. The difference between the two situations is that in case of the operational link the maximum number of enquire sequences is 32; in case of non-operational link that number should be 4.

6.8.7 LINK ESTABLISHMENT

The purpose of this phase is for the master to identify the state of the slave station that will be participating in data communications. In term of the procedure, it is very similar to Link Verification. The response of the slave device, when received by the master station, will be used to determine the odd/even state of the slave.

Link Establishment activities should only be terminated after the acceptable response is received from the slave station. The first <data_frame> transmission can not be initiated until the Link Establishment has been successfully completed.

Note the following distinction between the Link Establishment and Link Verification activities:

The Link Establishment activities should be performed upon power-up and should continue until the link is considered to be operational. <Data_frame> transmissions may not be initiated until the link has been established. After the Link has been established, the Link Verification activities are maintained on the link. <Data_frame> transmissions can be initiated (instead of Link Verification) at any time.

6.8.8 POWER-UP INITIALIZATION

The link establishment must be performed upon power-up. When the device powers-up, it must initialize as follows:

1. It should set its own last response sent to be a <negative_ack>.

2. It should perform a link establishment procedure as described in 6.8.7. The logic to establish the initial state of <id> to use in the first <data_frame> is:

If the response from the slave station was <dle><even> or <dle><nak>, the value of <id> should be set to <odd>.

If it was <dle><odd>, the value of <id> should be set to <even>.

6.8.9 DATA TRANSFER RULES

After the link establishment has been completed the data transfer functions can be performed whenever requested by the upper or the lower layers of the protocol. The responsibility of the master is to send the <data_frame> and receive a positive acknowledgment from the slave, which signals the completion of the transfer of the current <data_frame>. At this point the master can proceed with the next <data_frame> when it is submitted by the upper layer. Data transfer rules define what the master device should do when the negative acknowledgment is received, and what number of retries are required by the protocol before the master considers it appropriate to indicate a communications error.

Figure 6.8 illustrates the data transfer procedure rule. Refer to the appropriate sections in the Data Link Layer definition to better understand the information presented in the figure.

Note that the Error Recovery terminator indicated on the figure is basically an action taken to indicate to the upper layer that the operations needed to fulfill a specific upper layer's request cannot be completed.

Note also that the timing considerations are not presented in the figure. For example, the response could potentially be collected while the <data_frame> is being transmitted, as can be the case with an error response from the slave station. All timing considerations have been stated previously in this section and should be considered when reviewing this figure.

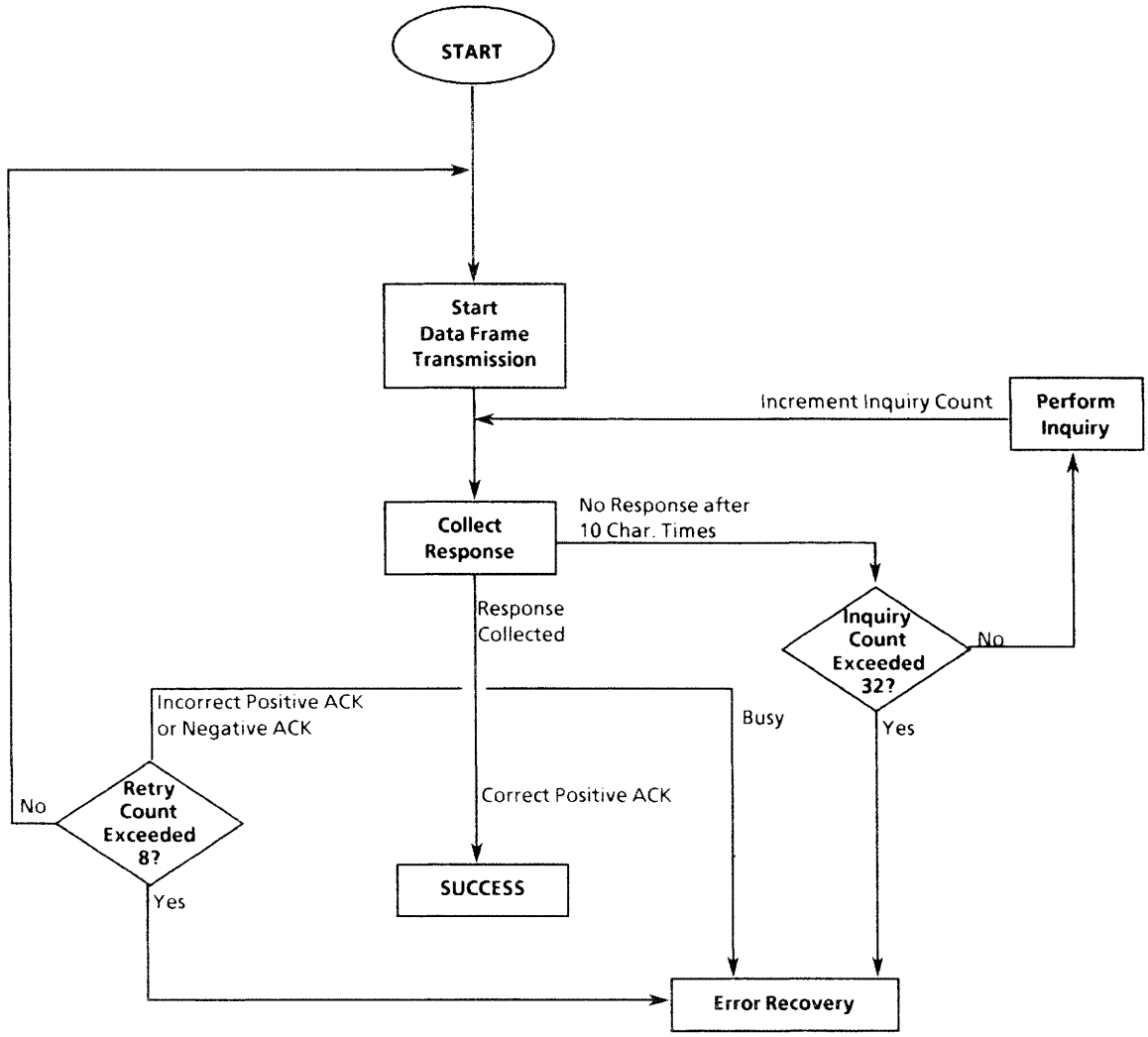


Figure 6.8 - Data Link Transfer Rules

7 PHYSICAL LAYER

7.1 Introduction

The Physical Layer provides the physical media for communications and its definition consists of the description of the electrical and mechanical characteristics of the media as well as the control signals and pin configuration used on it.

7.2 Media

The Point-to-Point communication system is designed to communicate over the twisted pair wiring, but it is not necessarily restricted to the wire. Fiber optic links, modems, or radios may be used in the Point-to-Point communication circuits.

7.3 Electrical Characteristics

EIA standard RS-422 for electrical characteristics of balanced voltage digital interface circuits is used for differential line drivers and receivers in SY/MAX Point-to-Point circuits. Since RS-232-C is not directly compatible with RS-422, a converter device is necessary in order to establish communication with a RS-232-C compatible device.

7.4 Control Signals

In addition to Transmitted Data and Received Data signals, Request To Send (RTS) and Clear To Send (CTS) are provided on some SY/MAX devices. These differential control signals can be utilized when interfacing with modems or printers. All of these signals comprise a subset and are compatible with RS-449 data communications interface standard.

SY/MAX Point-to-Point protocol does not require the use of hardware handshaking when the two devices are connected directly over wire media. Hardware handshaking is only employed for use with modems, radio's, printers, etc. that require additional flow control capabilities. If a SY/MAX device that implements RTS, CTS handshaking is connected to a device that does not use handshaking, the RTS and CTS pins should be jumpered as 5 to 6 and 7 to 8.

7.5 Asynchronous Format

SY/MAX P-P protocol is an asynchronous, serial, full-duplex format. The 11-bit word structure is shown in figure 7.1.

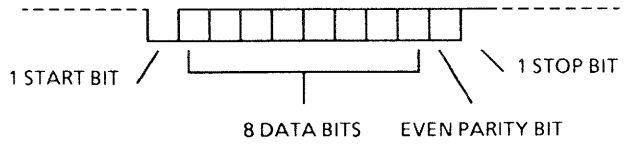


Figure 7.1 - SY/MAX P-P protocol format

7.6 Pin Configuration / Mechanical Characteristics

There are two types of connector interface pin configurations used throughout the SY/MAX product line for Point-to-Point communications. The 9 pin subminiature "D" connector is used extensively in both types of interfaces. Female connectors are used on SY/MAX devices and male plugs are used on all cables.

The Point-to-Point port pin configuration of the Network Interface Modules and Data Logging devices is as follows:

PIN	SIGNAL	DIRECTION
1	Transmitted Data -	Out
2	Transmitted Data +	Out
3	Received Data -	In
4	Received Data +	In
5	Clear To Send +	In
6	Request To Send +	Out
7	Clear To Send -	In
8	Request To Send -	Out
9	Shield (capacitive tie to the chassis)	

The pin configuration used in SY/MAX programmable logic controllers for the Point-to-Point communications does not provide the Request to Send and Clear To Send signals. Instead, these pins provide power to handheld programming equipment. The following is the corresponding pin assignment:

PIN	SIGNAL	DIRECTION
1	Transmitted Data -	Out
2	Transmitted Data +	Out
3	Received Data -	In
4	Received Data +	In
5	+ 5 Volts DC	Out
6	+ 5 Volts DC	Out
7	DC Power Common	Out
8	DC Power Common	Out
9	Shield (capacitive tie to the chassis)	

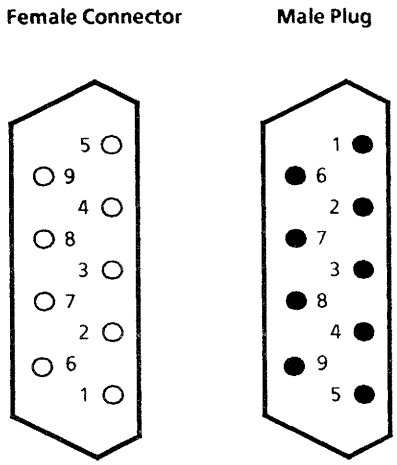


Figure 7.2 - Connector Pin Assignments

8 SOFTWARE FOR SY/MAX COMMUNICATIONS

8.1 Introduction

This section intends to explore the implementation issues of the SY/MAX protocol and should be of interest to the reader aiming to develop software capable of handling the SY/MAX Point-to-Point communications. Topics presented in this section include application-specific considerations as well as critical protocol issues which need to be addressed and implementation hazards to avoid.

8.2 Protocol Addendum

This sub-section lists some features of or corrections to SY/MAX P-P communications that are not part of the formal definition of the protocol, but are important details for implementation.

Baud Rate

Even though the protocol definition does not define baud rates used in the SY/MAX P-P communications, at the time of this document's writing they range between 75 and 19,200 bps.

Character time
(Data Link Layer)

The Data Link Layer definition includes reference to character time as a unit of time measurement. It is the amount of time it takes for the transmitter to send out a character including start, stop and parity bits. Obviously, this time is baud rate dependent. This means that a time period of 10 character times at 2400 bps is four times that at 9600 bps.

Negative Acknowledgment Reception
(Data Link Layer)

Negative Acknowledgment indicates that the slave station could possibly have encountered a framing error. According to the protocol, the corresponding <data_frame> must be retransmitted. It is suggested that prior to retransmission of the <data_frame>, the master station sends a sequence of 10 <pad> characters (at least one) and waits for a total of 10 character times before retransmission. This is necessary to facilitate the resynchronization of the two stations.

SYINET Broadcast Messages

If the last node in the route of the command message is above 233, the message is considered to be a broadcast message. When this type of message is transmitted by the device through the NIM and onto the SY/NET network, it will be accepted by all appropriate NIM modules and forwarded to the attached end devices (see references 3 and 4 for more information). In this case, no replies to the command message will be generated. This constitutes the only exception to the rule of the Device Command Layer stating that a reply is required for each command message.

Note that when two end devices are directly connected Point-to-Point with no intervening network, the command initiating device will receive a reply, even with a broadcast route in the route field.

<id> Toggling
(Data Link Layer)

It is REQUIRED of the master station to toggle <id> between <even> and <odd> AFTER EVERY SUCCESSFUL <data_frame> TRANSMISSION. When, under certain error conditions, retransmission of the <data_frame> is necessary, the retransmitted <data_frame> should NOT toggle the <id>.

It is NOT a responsibility of the slave station to verify that the master fulfills the above stated requirement.

Note that the state of <id> field of the <data_frame> is independent between two stations. The <id> in the <data_frame> of the reply message MAY OR MAY NOT match the <id> of the corresponding command message. Only responses must match the <id> of <data_frame>.

Illegal <id> Value Reception
(Data Link Layer)

Note that, if during the <data_frame> reception the slave device detects an <id> field with a value other than <odd> or <even>, a negative acknowledgment must be sent to indicate an error.

Reception of Data Inside and Outside of the <data_frame>

As stated previously, start of the <data_frame> is signaled with an occurrence of the <start_header>. From that point on, up to and including the <checksum>, all binary data, including an <embedded_response> type of information, is considered to be inside of a <data_frame>. All other data is considered to be received outside of the <data_frame>. The definition of Data Link protocol presents a precise definition of the legal <data_frame>. Rules on how to handle some illegal situations are:

Inside the <data_frame>

If an illegal value is found following <dle>, the <data_frame> should be negatively acknowledged. Whether or not a value following <dle> is legal might depend on which part of the <data_frame> is currently being received.

For example, the <dle><etx> sequence is legal if at some point in time during the reception of the current <data_frame>, <dle><stx> was received. Otherwise, it is illegal and must be negatively acknowledged.

However, some other <dle> sequences are illegal independent of the place of their occurrence within the <data_frame>. For example, <dle> followed by <enq> is illegal when embedded anywhere within a <data_frame>.

Outside of the <data_frame>

Rules on reception of data outside of the <data_frame> are as follows:

Reception of <dle><soh> indicate start of the <data_frame>. At this point rules on reception inside the <data_frame> start taking effect.

Reception of <positive_ack>, <negative_ack>, <busy> or <inquiry> is legal and should trigger an appropriate sequence of actions prescribed by the Data Link Layer definition.

Reception of anything else should cause no action to be taken, as if nothing was received.

Reception of <busy>

As stated before, reception of a <busy> in response to a <data_frame> triggers the retransmission of a <data_frame>, either the same or, potentially, a different one. It is recommended that retransmission of the <data_frame> is started 10 character times after the reception of the <busy> sequence, to give the busy device time to free the necessary resources.

AS INDICATED BEFORE, TRANSMISSION OF THE <BUSY> AS A RESPONSE TO THE <INQUIRY> CONSTITUTES A MAJOR PROTOCOL VIOLATION AND WILL LEAD TO A LOCK-UP CONDITION. Although some SY/MAX devices may conditionally send <busy> in response to an inquiry, no future implementations should because it will create lock-up situation.

Sequencing of Replies

Earlier in this document, a distinction was drawn between priority and non-priority commands. When generating appropriate replies to commands within each of two groups, the order of the replies should follow the order of the commands these replies correspond to. In other words, all priority replies should be generated in the same order as the priority commands were received in. The same applies to non-priority operations.

Effect of <busy> count on <nak> retry count

Note that since the <busy> count is handled in the Device Command Layer and retry count is dealt with in the Data Link Layer, a change in the <busy> counter implies resetting the <nak> retry counter to 0.

Multiple Outstanding Messages (Device Command Layer)

Note that some older SY/MAX devices will accept only one command message at a time, and will send a <busy> sequence in response to additional command messages.

8.3 Implementation Warnings

The following is a brief list of items to watch out for and avoid.

Busy Condition (Device Command Layer)

When the device receives an indication that the other station is unable to process the incoming <data_frame>, IT MUST ALWAYS FOLLOW THE COURSE OF ACTION SPECIFIED BY THE DEVICE COMMAND LAYER SEMANTICS IN SECTION 4.6.

Remember that a device should have the resources available to accept a reply message even when indicating a busy condition to an incoming command message.

Receive Buffer Overflow (Data Link Layer)

When collecting an incoming <data_frame>, the data link software must be able to detect a situation where the length of the incoming <data_frame> exceeds the maximum dictated by the format of the longest message of the Device Command Layer. When the software detects that the incoming <data_frame> exceeds the maximum allowed length, it should respond with a <busy>.

Note that, although <embedded_responses> are part of the <data_frame>, they are reacted to immediately upon reception and do not need to be buffered. This makes it easy to establish the maximum size needed for the <data_frame> buffer, which is 295 bytes without <embedded_response>s.

8.4 Application-Specific Considerations

No-reply detection and time-out

SY/MAX P-P protocol does not state any rules about the necessity of the reply coming back within a certain time period after the command transmission. Depending on the application, a "no reply within certain time period" situation may need to be detected. The length of that time period is referred to as the "no reply time-out" and can be implemented by the user application as dictated by the application considerations.

8.5 Software Design Considerations

Response transmission (Data Link Layer)

Whenever a device detects a situation requiring transmission of a response, for link efficiency and maximum throughput considerations it is best to send it as soon as possible. However, that does not mean that the design of the Data Link Layer software should place priority on detection of these situations over structured, layer-like implementation.

8.6 Read Register Example

This example presents in detail some of the steps involved in a typical implementation of station A reading registers from station B using a Non-Priority READ command.

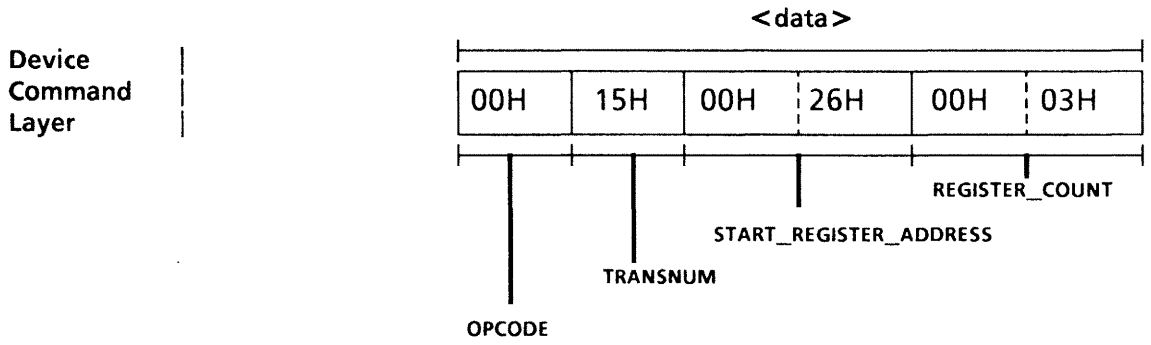
It is assumed that:

- No errors of any kind occur during this transaction.
- Stations A and B are connected directly P-P.
- The User Application has already provided the Network Layer with complete routing info, indicating that station B on the P-P connection has the route of 5, 123.
- Some communications have occurred prior to this point in time and the link is operational.
- Station B does not initiate commands throughout this read register transaction.

1. The operation is started by station A's User Application placing a request to Device Command Layer. The request contains the following information:

   Type of operation: Non-Priority Read Register
   Start reading at register number: 20
   Number of registers to be read: 4

2. The Device Command Layer performs needed data translations and assembles the buffer as shown in Figure 8.1.



OPCODE: 00H, Non-Priority Read Register Command
TRANSNUM: 15H, previous messages have been sent
START_REGISTER_ADDRESS: 0026H, calculated from 20 decimal = 14H
(14H-1) *2 = 26H
REGISTER_COUNT: 0003H, calculated from 4-1 = 03H

Figure 8.1 Device Command Layer Fields (Command Message)

The Device Command Layer passes this information down to Network Layer.

3. The Network Layer calculates the <route> field by performing data transformations on the route information (See Figure 8.2).

Example: *Route of 2 drops : 05, 123*

Drop	#1	#2
Decimal	05	123
Hex	05	7B
ASCII Hex	30 35	37 42

This information is then passed to the Data Link Layer together with the info contained in Device Command Layer buffer.

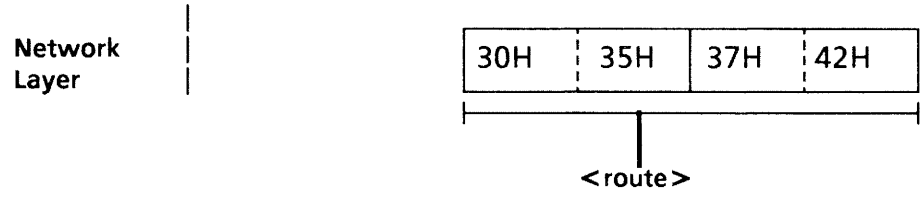
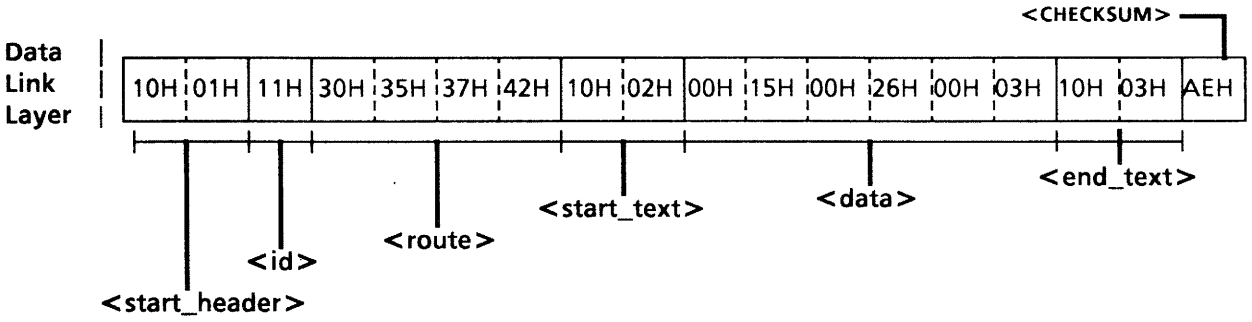


Figure 8.2 Network Layer Fields (Command Message)



<CHECKSUM> is calculated as:
10 + 01 + 11 + 30 + 35 + 37 + 42 + 10 + 02 + 00 + 15 + 00 + 26 + 00 + 03 + 10 + 03 = 152H
the <checksum> is a single byte so only the least significant byte is used = 52H
Two's complement of 52H = AEH

Figure 8.3 Data Link Layer Fields (with Checksum Calculation)

4. The Data Link Layer then calculates the checksum value as shown in Figure 8.3.

5. This information is passed on to the Physical Layer and is transmitted over the physical medium to station B. At the end of the <data_frame> transmission, station A sends out at least one <pad> character.

6. Station B's Data Link Layer collects the <data_frame> and transmits an odd positive acknowledgment (<dle><odd>) back to station A.

7. Station A's Data Link layer collects the <dle><odd> response and reports (through the Network Layer to the Device Command Layer) successful completion of the transmission. For the purposes of the Data Link Layer, the transaction has been completed.

8. Station B's Data Link Layer strips off the framing and other Data Link Layer components and passes the <route> and the <data> up to the Device Command Layer (through the Network Layer).

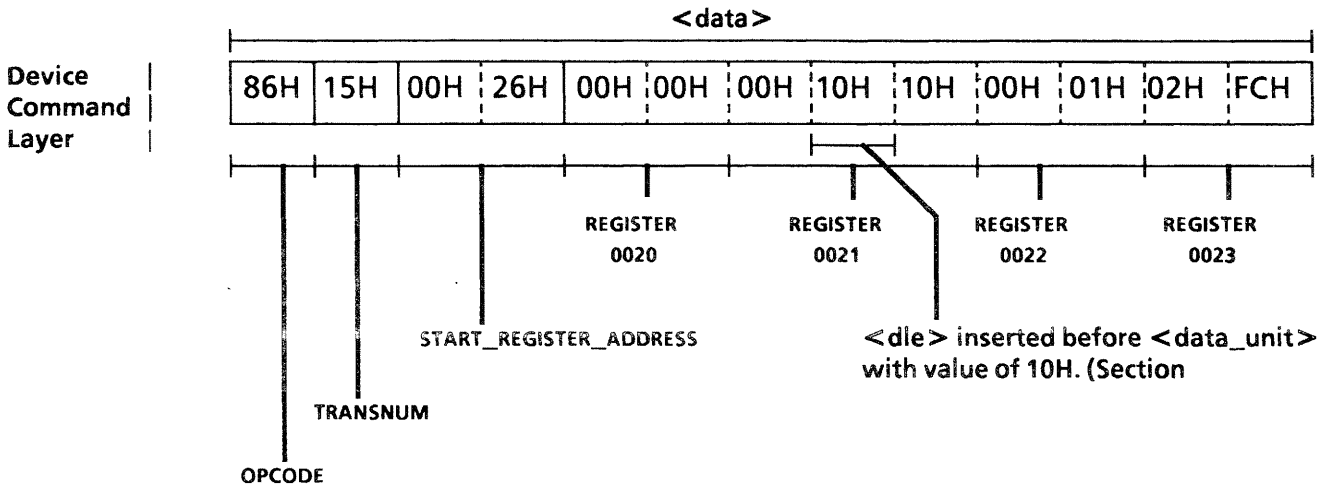
9. Station B's Command Layer performs the appropriate data transformations on START_REGISTER_ADDRESS and REGISTER_COUNT and interprets the command as:

   Type of operation: Non-Priority Read Register
   Start reading at register number: 20
   Number of registers to be read: 4

It then performs the read operation by collecting data from registers 20, 21, 22 and 23. Suppose the data was as follows (decimal):

REG. 20: 0
REG. 21: 16
REG. 22: 1
REG. 23: 764

The constructed reply of the Device Command Layer will be as shown in Figure 8.4.



OPCODE: 86H, Non-Priority Read Register Reply
TRANSNUM: 15H, same as Command Message
START_REGISTER_ADDRESS: 0026H, same as Command Message
REGISTER_DATA: Registers 0020 - 0023

Figure 8.4 Device Command Layer Fields (Reply Message)

This information and associated route is passed on to the Station's B Network Layer.

10. Network Layer performs route field reversal. (See Figure 8.5).

Original Route:	30 35	37 42
Reversed Route:	37 42	30 35

11. Now a request is placed by the Network Layer to the Data Link Layer to transmit the reply data and associated route. (See Figure 8.6).

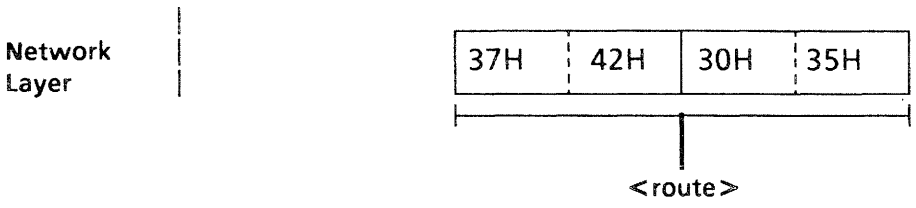
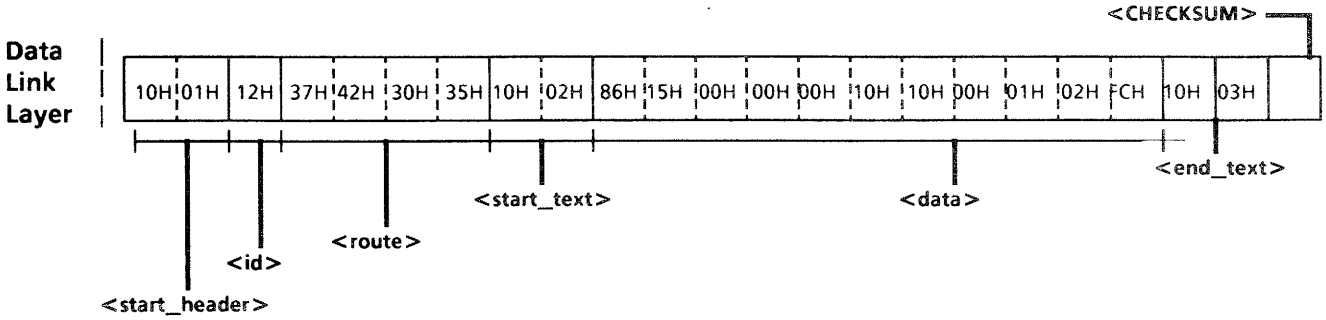


Figure 8.5 Network Layer Fields (Reply Message)



<CHECKSUM> is calculated as:
10 + 01 + 12 + 37 + 42 + 30 + 35 + 10 + 02 + 86 + 15 + 00 + 00 + 00 + 10 + 10 + 00 + 01 + 02 + FC + 10 + 03 = 2E0H
Least Significant byte = E0H
Two's Complement of E0H = 20H

Figure 8.6 Data Link Layer Fields (with Checksum Calculation)

12. This information is passed on to the Physical Layer and is transmitted over the physical medium to station A. After the <data_frame> transmission, station B sends out at least one <pad> character.

13. Station A's Data Link Layer collects the <data_frame> and sends an even positive acknowledgment in the form of <dle><even> back to station B.

14. Station B's Data Link Layer collects <dle><even> and reports (through the Network Layer to the Device Command Layer) successful completion of the transmission. For the purposes of the Station B's Data Link Layer, the transaction has been completed.

15. Station A's Data Link Layer strips off the framing and other Data Link Layer components and passes the <route> and the <data> up to the Device Command Layer (through the Network Layer).

16. Station A's Command Layer performs the data transformations interprets the command as:

    Type of operation: Non-Priority Read Register Reply
    Register data starting at register number: 20
    Register data is as follows:
    REG.20: 0
    REG.21: 16
    REG.22: 1
    REG.23: 764

This data can now be used by Station A's User Application.

8.7 Data Link Layer State Diagrams

The dynamic of events occurring at the transmitter and receiver are fairly complex. Although somewhat independent, software components participating in these events communicate with each other using a number of signals.

To better illustrate the dynamics of the transmitter/receiver system of the Data Link Layer, the concept of a State Transition Diagram (STD) is employed. STD is a model describing the following:

1. What causes the system to change its operation, and

2. What actions are taken when the need to change operation is recognized

An STD shows the dynamics of the system as a network of states connected by transitions. Figure 8.7 defines the STD notation.

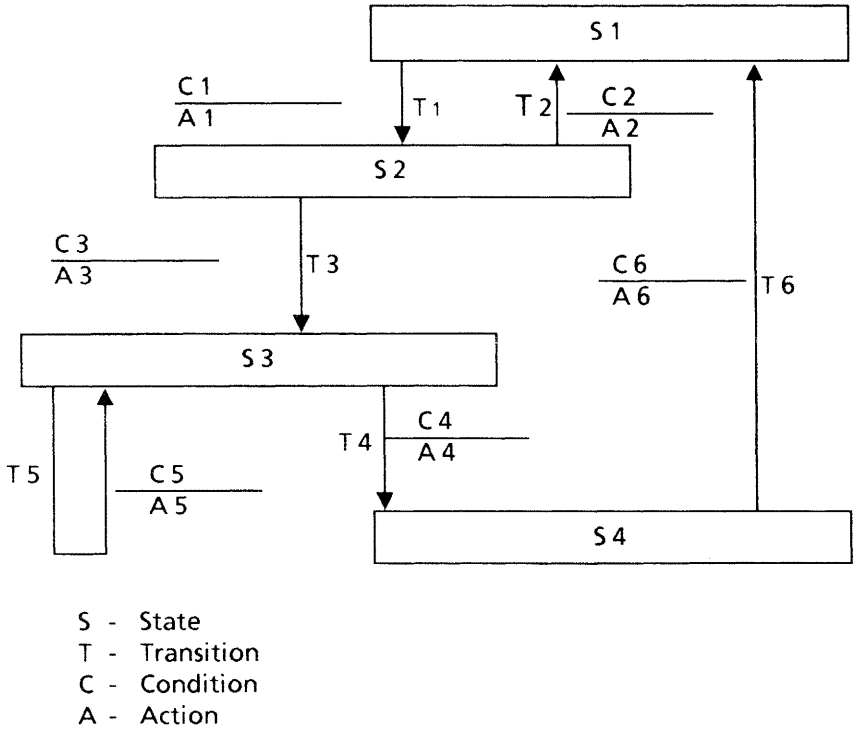


Figure 8.7 State Transition Diagram Notation

States

A State is a mode of the system, characterized by a particular set of operations, and identified by a unique combination of conditions and destination states. A State is a static aspect of the system awaiting for conditions to occur.

The dynamics of the transmitter/receiver system of the Data Link Layer are defined using two STDs.

Transitions

Transitions represent a movement of the system from one state to another. Transitions typically have two parts: the Condition and the Action. Transition Conditions cause the system to move from one state to another, and are located above the transition line (Figure 8.7). Transition Actions are carried out by the system as it moves from one state to another, and are located below the transition line.

Transition Actions

Transition Action may include triggering a "one-shot" data process (transformation) or enabling/disabling continuous data transformation.

A continuous data transformation is a transformation that persists over time. Thus it must be Enabled (shown as E:) and Disabled (D:).

A "one-shot" can be completed in short (relatively instantaneous) time. It must be activated by a Trigger (shown as T:)

Transitory States

Transitory States are states that persist for a short time. They are not named, but are used for decisions. Transitory States typically have Trigger actions associated with transitions into the states.

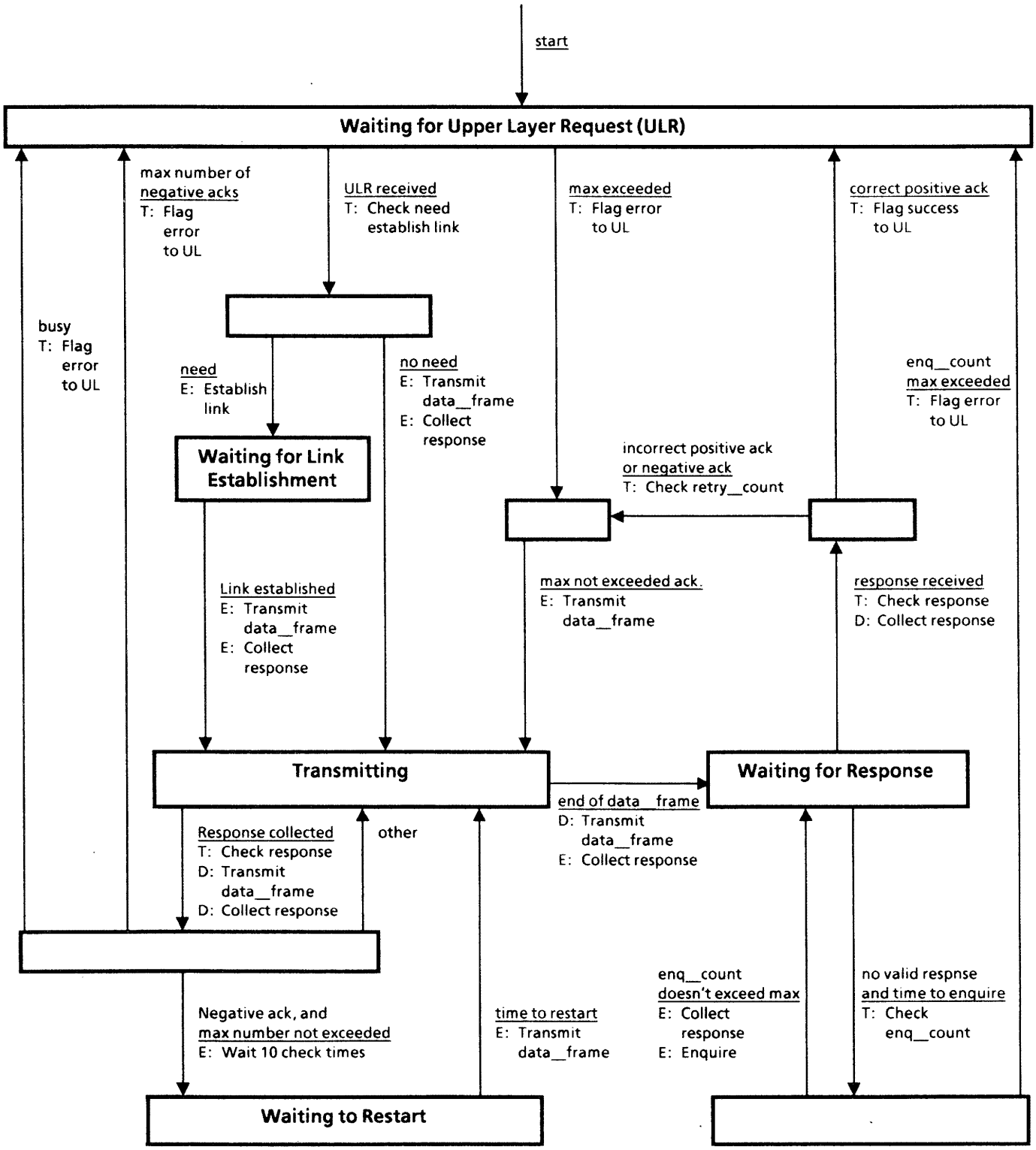
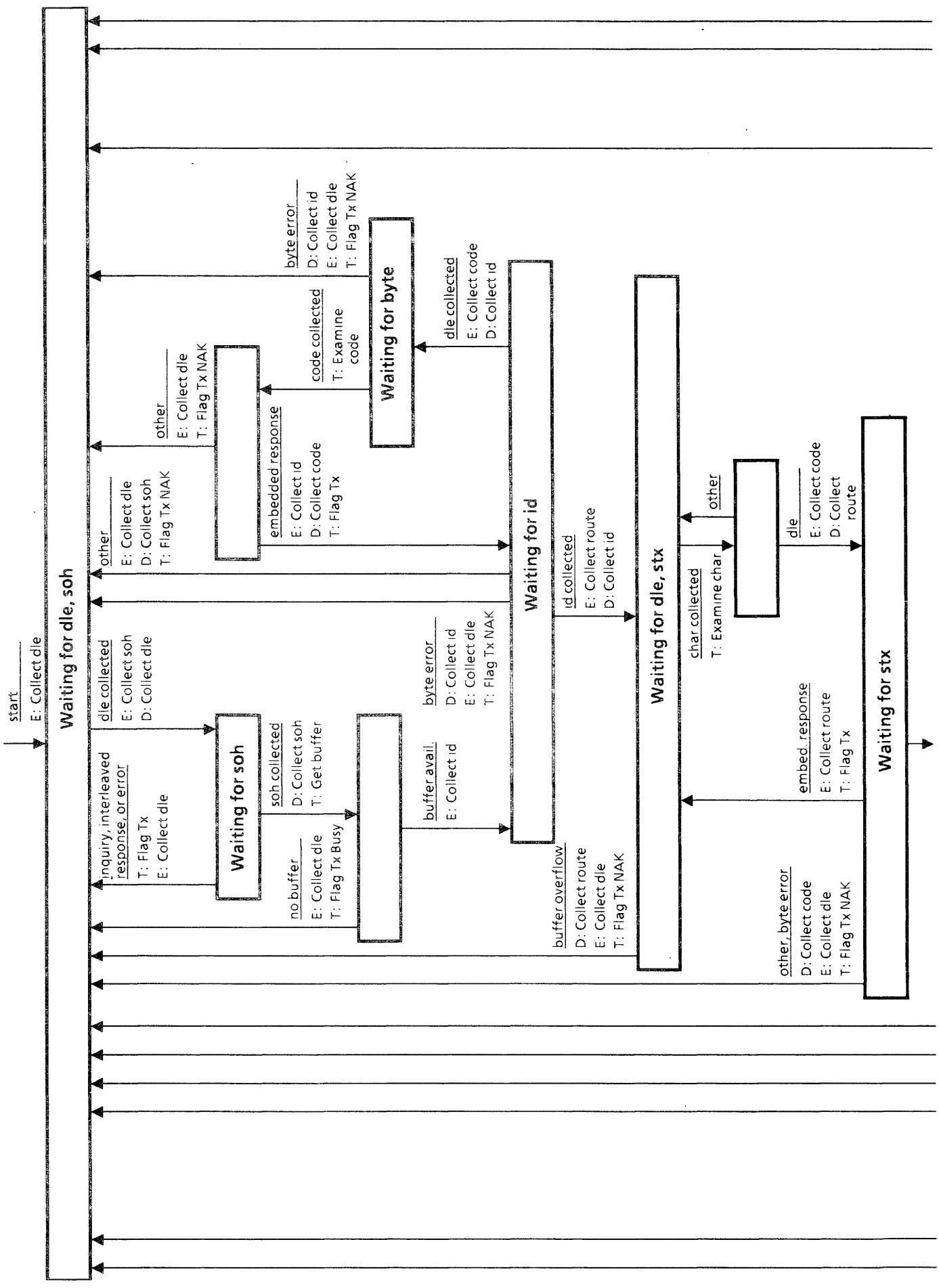


Figure 8.8 DLL Transmitter State Transition Diagram


start
E: Collect dle
Waiting for dle, soh
dle collected
E: Collect soh
D: Collect dle
inquiry, interleaved response, or error
T: Flag Tx
E: Collect dle
Waiting for soh
soh collected
D: Collect soh
T: Get buffer
no buffer
E: Collect dle
T: Flag Tx Busy
buffer avail.
E: Collect id
byte error
D: Collect id
E: Collect dle
T: Flag Tx NAK
Waiting for id
id collected
E: Collect route
D: Collect id
buffer overflow
D: Collect route
E: Collect dle
T: Flag Tx NAK
Waiting for dle, stx
other
E: Collect dle
D: Collect soh
T: Flag Tx NAK
other
E: Collect dle
T: Flag Tx NAK
embedded response
E: Collect id
D: Collect code
T: Flag Tx
code collected
T: Examine code
Waiting for byte
dle collected
E: Collect code
D: Collect id
byte error
D: Collect id
E: Collect dle
T: Flag Tx NAK
char collected
T: Examine char
other
dle
E: Collect code
D: Collect route
embed. response
E: Collect route
T: Flag Tx
other, byte error
D: Collect code
E: Collect dle
T: Flag Tx NAK
Waiting for stx

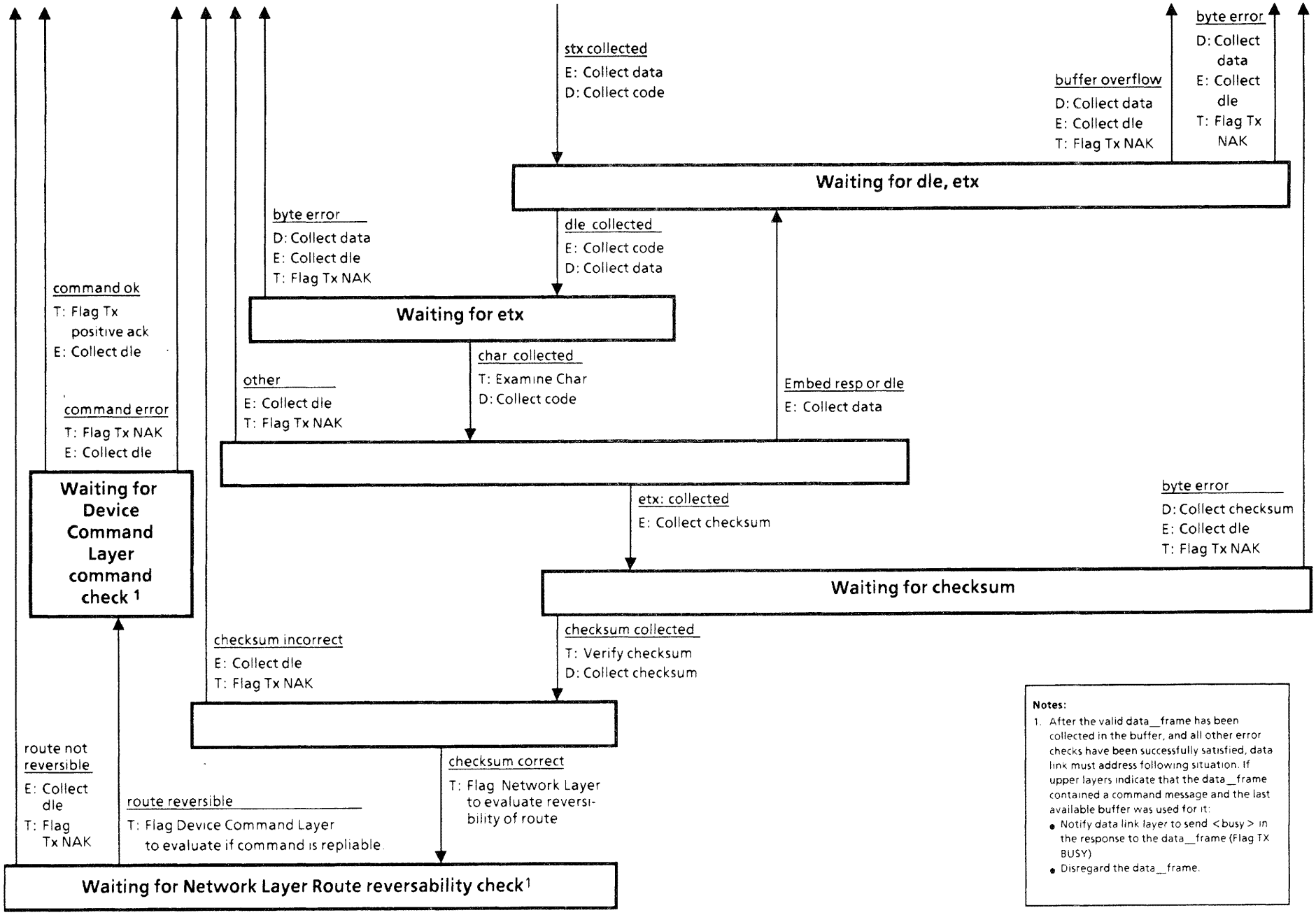


Figure 8.9 DLL Receiver State Transition Diagram

This page intentionally left blank.

APPENDIX A SOURCES OF INFORMATION

1. Lane M.G., Data Communications Software Design, p. 92 Computer Science Series, Boyd and Fraser, 1985

2. X3.28 American National Standard Procedures for the Use of the Communication Control Characters of American National Standard Code for Information Interchange in Specified Data Communications Links, ANSI, 1976

3. SY/MAX Class 8030 Type CRM-510 SY/NET Network Interface Module, Instruction Bulletin 30598-257-01, Square D, November 1984

4. SY/MAX Class 8030 Type SFI-510 SY/LINK Communication Interface Board, Instruction Bulletin 30598-277-01, Square D, July 1987

5. EIA Standard RS-232-C

6. EIA Standard RS-449-A

7. EIA Standard RS-422-A

This page intentionally left blank.

APPENDIX B ANSI X3.28 ANCESTRY

B.1 Introduction

As mentioned earlier, SY/MAX Point-to-Point protocol was based on the ANSI X3.28 standard for data communications. This standard does not truly cover the multilayer structure of the protocol but rather "presents control procedures for specified data communication links that employ the communication control characters of American National Standards Code for Information Interchange, X3.4 -1968 (ASCII)". In other words, it deals primarily with Data Link Layer aspects of the protocol. Note that elements referred to as "subcategories" in this section should not be confused with "categories" described in the Device Command Layer Protocol.

The standard defines communication control characters and construction of the control sequences utilizing a technique of code extension using DLE (data link escape). It then demonstrates the subcategories of establishment and termination control procedures. Finally, the subcategories of the message transfer control procedures are presented. An appropriate subcategory should be selected for the link protocol implementation depending on such parameters as link configuration, desired throughput and so on.

B.2 Communication Control Characters

The communication control character set defined by the standard is essentially equivalent to the one used in the Data Link Layer of SY/MAX Point-to-Point protocol and, although the names of control sequences do differ in some instances, their intended use is the same.

B.3 Establishment/Termination and Message Transfer Procedures

There isn't a single establishment and termination or message transfer control procedure subcategory of the ANSI X3.28 standard that can be identified as uniquely employed in the SY/MAX Point-to-Point protocol. Although based on the standard, the SY/MAX protocol has certain variations and departures from it. However, it is certainly possible to identify the primary characteristics of the protocol, indicate their origins in the standard, point out the differences and recognize unique features of the Data Link Layer of the SY/MAX Point-to-Point protocol.

B.4 Message Transfer Issues

Message transfer subcategory F1 meets most of the Data Link protocol requirements listed in section 6.2 providing "message-associated blocking with longitudinal checking and alternate acknowledgments for two-way simultaneous operation with embedded responses". Message transfer subcategory D1 supplies transparent header and data area attributes. SY/MAX Data Link Layer protocol is based on both of these subcategories. It fulfills the necessary characteristics requirements by combining the features offered by these two subcategories. The only feature of the SY/MAX P-P protocol that is not provided for by the F1 and D1 subcategories is the <busy> response.

B.5 Establishment and Termination Issues

The establishment and termination phases of the SY/MAX P-P Data Link protocol are not nearly as intricate as any of the subcategories prescribed by the ANSI X3.28 standard. As a matter of fact, the F1 subcategory submits that on a nonswitched, dedicated, full duplex link where both stations idle in ready-to-receive state, no establishment and termination phases are necessary. If any, the closest subcategory would be 3.1, described as "two-way simultaneous, switched Point-to-Point". The establishment aspect of SY/MAX P-P Data Link protocol is basically equivalent to the establishment phase of the 3.1 subcategory, but there is no termination phase in SY/MAX P-P Data Link protocol.

This page intentionally left blank.

APPENDIX C CONTROL CODES

C.1 Special Control Codes Usage and Interpretation

HEX CODE	SYMBOL	DESCRIPTION/FUNCTION
01	<soh>	Part of the start_header sequence
02	<stx>	Part of the start_text sequence
03	<etx>	Part of the end_text sequence
05	<enq>	Part of the inquiry sequence
10	<dle>	Used as the data link escape code to trigger special interpretation of the following code
11	<odd>	Used as data_frame's id as well as a part of the positive_ack response
12	<even>	Used as a part of negative_ack response
15	<nak>	Used as a part of busy sequence
16	<syn>	Used at the trailing end of the data_frame for synchronization purposes

Figure C.1

C.2 Control Sequences

SEQUENCE	TERMINOLOGY
<dle><soh>	Master generated framing component: start of header area
<dle><stx>	Master generated framing component: start of text area
<dle><etx>	Master generated framing component: end of text area
<dle><enq>	Master initiated inquiry: command the slave to repeat its last response
<dle><odd>	Slave initiated response: positive acknowledgement (odd)
<dle><even>	Slave initiated response: positive acknowledgement(even)
<dle><nak>	Slave initiated response: negative acknowledgement
<dle><syn>	Save initiated response: informs the master that the slave is temporarily unable to process the incoming data

Figure C.2

C.3 ASCII Codes

Decimal	Hex	Character
0	00	ctrl @ NUL
1	01	ctrl A SOH
2	02	ctrl B STX
3	03	ctrl C ETX
4	04	ctrl D EOT
5	05	ctrl E ENQ
6	06	ctrl F ACK
7	07	ctrl G BEL
8	08	ctrl H BS
9	09	ctrl I HT
10	0A	ctrl J LF
11	0B	ctrl K VT
12	0C	ctrl L FF
13	0D	ctrl M CR
14	0E	ctrl N SO
15	0F	ctrl O SI
16	10	ctrl P DLE
17	11	ctrl Q DC1
18	12	ctrl R DC2
19	13	ctrl S DC3
20	14	ctrl T DC4
21	15	ctrl U NAK
22	16	ctrl V SYN
23	17	ctrl W ETB
24	18	ctrlX CAN
25	19	ctrl Y EM
26	1A	ctrl Z SUB
27	1B	ctrl [ESC
28	1C	ctrl \ FS
29	1D	ctrl] GS
30	1E	ctrl ^ RS
31	1F	ctrl _ US
32	20	Space
33	21	!
34	22	"

Figure C.3

Decimal	Hex	Character
35	23	#
36	24	$
37	25	%
38	26	&
39	27	'
40	28	(
41	29	)
42	2A	*
43	2B	+
44	2C	,
45	2D	-
46	2E	.
47	2F	/
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
58	3A	:
59	3B	;
60	3C	<
61	3D	=
62	3E	>
63	3F	?
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E

Figure C.3 continued

Decimal	Hex	Character
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z
91	5B	[
92	5C	\
93	5D	]
94	5E	^
95	5E	_
96	60	'
97	61	a
98	62	b
99	63	c
100	64	d
101	65	e
102	66	f
103	67	g
104	68	h

Figure C.3 continued

Decimal	Hex	Character
105	69	i
106	6A	j
107	6B	k
108	6C	l
109	6D	m
110	6E	n
111	6F	o
112	70	p
113	71	q
114	72	r
115	73	s
116	74	t
117	75	u
118	76	v
119	77	w
120	78	x
121	79	y
122	7A	z
123	7B	{
124	7C	|
125	7D	}
126	7E	~
127	7F	DEL

Figure C.3 continued

This page intentionally left blank

APPENDIX D SY/MAX ERROR CODES

NOTE: Only ODD numbered error codes can be used in the priority replies.

CODE		
Dec.	Hex.	ERROR DESCRIPTION
000	00	No error
001	01	Illegal Protocol Opcode
002	02	Illegal Intermediate Code
003	03	Illegal Address Attempted
004	04	Illegal Rack Address Attempted
005	05	Instruction Attempted Which Is Not Allowed
006	06	Item Being Searched For Cannot Be Found
007	07	Attempt To Alter Data In A Protected Register
008	08	Attempt To Alter Data In Protected Memory
009	09	Attempt To Alter Data In A Read-Only Register
010	0A	Attempt To Exceed Memory Limitations
011	0B	Communications Overflow
012	0C	Illegal CPU Rack Addressing
013	0D	Communications Error
014	0E	The Operation Attempted Is Not Allowed In Run
015	0F	Communication Overflow
016	10	Register Count Too Large For The Processor
017	11	Remote Device Inactive
018	12	The Rung Number Used Is Not Allowed
019	13	Illegal Read Parameter
020	14	Illegal Channel Number
021	15	Trying To Change A Forced Bit
022	16	Forcing Function Is Inhibited
023	17	Attempt To Alter Data In A Fenced Register
024	18	Attempt To Force A Non-Forcible Register

Figure D.1

CODE		
Dec.	Hex.	ERROR DESCRIPTION
025	19	CPU Error, Check Register 8175
026	1A	Rack Addressing And User Memory Overlap
027	1B	Memory Error - Clear All Operation Required
028	1C	Illegal Baud Rate Selected
029	1D	Attempt To Send Message With Illegal Route
030	1E	Attempt To Alter Protected Memory - Inhibit Coil
031	1F	End Of File Encountered
032	20	Operation Not Allowed In Protected Memory
033	21	Data Error
034	22	UVPROM Memory Error
035	23	Write Protected
036	24	Replace Not Allowed - Use Delete And Insert
037	25	Attempt To Skip Or Read A File Past Last File
038	26	Data View Protected - Cannot Read
039	27	Alarm Already Set
040	28	Hardware Memory Security In Effect
*040	28	Conflict with above; illegal file type; used by welder products
041	29	Illegal Register Write
042	2A	Illegal File Number
043	2B	Illegal Operation Attempted
044	2C	File Not Found
045	2D	Operation Not Allowed - DLOG Tape Operation In Progress
046	2E	Protected File
047	2F	Operation Not Allowed Due To Keyswitch Position
048	30	Programming MCR In Subroutine Area Not Allowed
049	31	I/O, Register Or Channel Safeguarded

Figure D.1 continued

CODE		ERROR DESCRIPTION
Dec.	Hex.	
050	32	Rack Address Not Alterable While Forcing Active
051	33	Missing Module
052	34	Illegal File Type
053	35	Illegal Media Format - Erase Track/Clear Memory Required
054	36	Illegal Mark Number
055	37	File Operation Aborted
057	39	COMMS Queue Buffer Forced Emptied
059	3B	Battery Low
060	3C	Illegal Mark Number For GOTO or GOSUB
061	ED	Mark St. Sub Cannot Be Inserted - Append Only
062	3E	Return Rungs Not Allowed In Ladder Area
063	3F	Mark Number Previously Used
064	40	Operation Not Allowed In Run - Cannot Delete Return
065	41	Operation Not Allowed - Cannot Delete Mark St. Sub
066	42	Operation Not Allowed - Cannot Delete Mark Having Return
067	43	Operation Not Allowed - Return Must Have Mark
068	44	Operation Not Allowed - Only One Return Per Mark
069	45	Operation Not Allowed In Run - Gosub With No Return
070	46	Operation Not Allowed In Run - Goto To Mark With Return
071	47	Operation Not Allowed In Run - Cannot Delete Mark
072	48	Operation Not Allowed In Run - Unused Mark Number
079	4F	Operation Not Allowed In Run With Programmable Interrupt
080	50	Insufficient Data
081	51	Unrecognized Command
082	52	Illegal Parameter
083	53	Illegal Source Parameter
084	54	Illegal Destination Parameter

Figure D.1 continued

CODE		ERROR DESCRIPTION
Dec.	Hex.	
085	55	Invalid Filespec
086	56	Invalid Input Channel
087	57	Invalid Output Channel
088	58	Device Not Available
089	59	Serial Link Not Established
090	5A	Hardware Read Fault
091	5B	Hardware Write Fault
092	5C	Read Error
093	5D	Write Error
094	5E	Directory Read Error
095	5F	Directory Write Error
096	60	Directory Space Full
097	61	Device Space Full
098	62	File Not Found
099	63	File Access Denied
100	64	File Already Exists
101	65	File Already Open
102	66	File Format Error
103	67	File Wrong Type
104	68	Open Attempt To Open File
105	69	I/O Attempt To Open File
106	6A	Write Attempt To Read Only File
107	6B	Record Out Of Range
108	6C	Too Many Open Files
109	6D	Operation Aborted By User
110	6E	Source Filespec Required
111	6F	Destination Filespec Required
112	70	Maximum Open Count For File
113	71	Incompatible Source Device/File Type
114	72	Incompatible Destination Device/File Type

Figure D.1 continued

CODE		
Dec.	Hex.	ERROR DESCRIPTION
115	73	Device Timeout
116	74	Protected File
117	75	Illegal File Type
118	76	Illegal Device
119	77	File Type Required
120	78	Device Required
121	79	Channel Required
122	7A	Route Required
123	7B	Filename Required
124	7C	Illegal Offset
125	7D	Illegal Count
126	7E	Source/Destination Cannot Be Same Device
127	7F	Duplicate Symbol
128	80	File Mismatch
129	81	Invalid Register Address
130	82	Invalid File Name
131	83	Write Data Error
133	85	Insufficient Data For Write
135	87	Invalid Read Parameter
136	88	Communication Parity Error
137	89	Busy Data Transfer or Weld In Progress
138	8A	Communication Overrun Error
139	8B	Communication Framing Error
140	8C	Dropped Unexpected Reply
141	8D	Job Space Full
145	91	Write Sequence Not Initiated
147	93	Memory Full
149	95	No Data Found
150	96	Cannot Stop Attached Batch Job
151	97	Illegal Communication To COMM Port
153	99	Error Noted During A Transfer

Figure D.1 continued

This page intentionally left blank.

NEED HELP WITH SY/MAX® AUTOMATION PRODUCTS?

Class 8005, 8010, 8020, 8030, 8040, 8052, 8054

- **Troubleshooting Help**
- **Repair Parts Shipment**
- **Field Service**

If you have a breakdown requiring EMERGENCY SERVICE, at any time call the phone number shown below. Calling this number will alert a Technical Services Representative who will call you back to help with troubleshooting problems or make arrangements for repair parts shipment or Field Service.

Before calling, be ready to provide complete identification information including class and type number, series number and serial number (if any). When ordering repair parts or requesting field service assistance, be prepared to supply a purchase order number and other ordering information (such as shipping information, special airfreight routing, etc.).

FOR EMERGENCY SERVICE

CALL (800) 634-2003
or (708) 397-2600