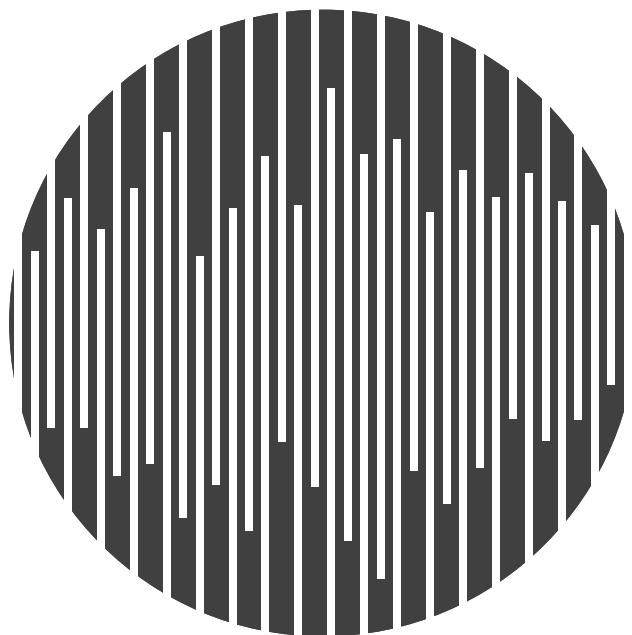


# **Modicon**

## **Modbus Plus Network**

### **I/O Servicing Guide**

840 USE 104 00 Version 2.0



**MODICON**



# **Modicon Modbus Plus Network I/O Servicing Guide**

840 USE 104 00    Version 2.0

**March 1996**

**AEG SCHNEIDER  
AUTOMATION**  
Modicon • Square D • Telemecanique

**AEG Schneider Automation, Inc.  
One High Street  
North Andover, MA 01845**



# Preface

---

This book describes the Modicon Modbus Plus Network Input/Output (I/O) system. It presents guidelines for developing network architecture, and provides examples to assist network developers in estimating I/O performance in typical network configurations.

The data and illustrations found in this book are not binding. We reserve the right to modify our products in line with our policy of continuous product development. The information in this document is subject to change without notice and should not be construed as a commitment by AEG Schneider Automation, Inc.

AEG Schneider Automation assumes no responsibility for any errors that may appear in this document. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

No part of this document may be reproduced in any form or by any means, electronic or mechanical, including photocopying, without express written permission of the Publisher, AEG Schneider Automation, Inc.



**Caution: All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to assure compliance with documented system data, repairs to components should be performed only by the manufacturer.**

MODSOFT® is a registered trademark of AEG Schneider Automation, Inc.

The following are trademarks of AEG Schneider Automation, Inc.:

Modbus	Modbus Plus	Quantum Automation Series
Modicon	984	

© Copyright 1996, AEG Schneider Automation, Inc.  
Printed in U.S.A.

## Related Publications

---

Refer to the following publications for further information about the Modbus Plus network and other Modicon products.

840 USE 100	Modicon Quantum Automation Series Hardware Reference Guide
890 USE 100	Modicon Modbus Plus Network Planning and Installation Guide
890 USE 101	Modicon Remote I/O Cable System Planning and Installation Guide
890 USE 102	Modicon IBM Host Based Devices User's Guide
890 USE 104	Modicon Terminal Block I/O Modules Hardware Reference Guide
890 USE 115	Modicon Modsoft Programmer User Manual

# Contents

---

<b>Chapter 1</b>	
<b>Introducing Modbus Plus I/O .....</b>	<b>1</b>
1.1 Introducing the Modbus Plus I/O System .....	2
1.1.1 Components of the I/O System .....	2
1.2 How the I/O System is Organized .....	4
1.3 Network Terminology .....	6
1.4 Network Media .....	7
1.4.1 Trunk Cable .....	7
1.4.2 Taps and Drop Cables .....	7
1.5 Network Cable Layouts .....	8
1.5.1 Basic Cable Layouts .....	8
1.5.2 Repeaters for Network Expansion .....	9
1.5.3 Bridges For Standard Modbus Plus Messages .....	11
1.6 How the I/O Network Operates .....	12
1.6.1 Overview of the Logical Network .....	12
1.6.2 The Token Rotation Sequence .....	12
1.7 Processes During Network Startup .....	14
1.7.1 How the I/O Nodes are Known to the Application .....	14
1.7.2 How the DIO Drop Adapters Become Configured .....	14
1.7.3 How the TIO Modules Become Configured .....	15
1.8 Ownership of the Write Privilege to Drops .....	16
1.8.1 Write Privilege to DIO Drop Adapters .....	16
1.8.2 Write Privilege to TIO Modules .....	17
1.9 How I/O Messages are Transacted .....	18
1.9.1 Handling I/O Messages .....	18
1.9.2 Message Sizes - I/O Master Nodes .....	18
1.9.3 Message Sizes - DIO Drop Adapter Nodes .....	18
1.9.4 Message Sizes - TIO Module Nodes .....	18
1.9.5 Handling Standard Modbus Plus Messages .....	19

<b>Chapter 2</b>	
<b>I/O Network Configuration Options</b>	<b>21</b>
2.1	Setting Your Network Performance Goal . . . . . 22
2.1.1	Factors to Consider . . . . . 22
2.1.2	Planning Your Application . . . . . 22
2.1.3	How Your Network Layout Affects Node Access . . . . . 22
2.1.4	Deterministic I/O Servicing or Multi-Function Servicing . 23
2.2	An Overview of Configuration Options . . . . . 24
2.2.1	Option 1: Deterministic I/O Device Servicing . . . . . 24
2.2.2	Option 2: Multi-Function Device Servicing . . . . . 24
2.2.3	Separating Networks by Function . . . . . 25
2.2.4	Minimizing Node Counts for Speed . . . . . 26
2.2.5	Assigning Node Addresses . . . . . 27
2.2.6	Connecting or Disconnecting Active Nodes . . . . . 27
2.2.7	Considerations for Using Bridge Mode . . . . . 28
2.2.8	Defining Output Module Default States . . . . . 29
2.3	Networks for Deterministic I/O Servicing . . . . . 30
2.3.1	CPU/NOM Servicing Time . . . . . 30
2.3.2	Calculating Token Rotation Time . . . . . 30
2.3.3	Example: Token Rotation Time . . . . . 31
2.4	The Effects of Paths on Network Timing . . . . . 32
2.4.1	How Paths are Used . . . . . 32
2.4.2	How Paths Affect Timing . . . . . 32
2.4.3	Normal I/O Transactions: Paths are Not Used . . . . . 33
2.4.4	Path Types . . . . . 34
2.4.5	Path Quantities . . . . . 35
2.5	Multifunction Networks With a Programmer . . . . . 36
2.5.1	Impact on I/O Servicing . . . . . 36
2.5.2	Impact on Token Rotation Time . . . . . 37
2.5.3	Connecting a Node Temporarily to the I/O Network . . . . 37
2.5.4	Calculating Token Rotation Time . . . . . 37
2.5.5	Example: Token Rotation Time . . . . . 38
2.6	Multifunction Networks With Mixed Nodes . . . . . 40
2.6.1	Impact on I/O Servicing . . . . . 41
2.6.2	Impact on Token Rotation Time . . . . . 41
2.6.3	Calculating Token Rotation Time . . . . . 41
2.6.4	Example: Token Rotation Time . . . . . 42



2.7	Estimating Token Reconstitution Time .....	44
2.8	Verifying the I/O Configuration .....	46
2.8.1	Verifying Node Addresses Before Startup .....	46
2.8.2	Verifying Controller Tables Before Startup .....	46
2.8.3	Verifying DIO Drop Adapter Configurations .....	46
2.8.4	Verifying TIO Module Configurations .....	46

### **Chapter 3**

## **Estimating I/O Network Performance ..... 47**

3.1	Estimating Data Throughput Time .....	48
3.1.1	Examples in This Chapter .....	48
3.1.2	Network Model for Measuring Throughput Time .....	49
3.1.3	Example: Time Versus Program Size .....	50
3.1.4	Example: Time Versus Program Size and I/O Drops .....	51
3.1.5	Comparison Between Quantum 113/213 and 424 CPU ...	52

### **Chapter 4**

## **Using Multiple Controllers ..... 53**

4.1	Using More Than One Controller .....	54
4.1.1	Multiple Controllers With DIO Drop Adapters .....	54
4.1.2	Multiple Controllers With TIO Modules .....	55
4.2	Example: Drops With Multiple Controllers .....	56
4.2.1	Processes During Startup .....	57
4.3	Configuration Requirements for Multiple Controllers .....	58
4.4	Verifying the I/O Configuration .....	59
4.4.1	Verifying Node Addresses Before Startup .....	59
4.4.2	Verifying Controller Tables Before Startup .....	59
4.4.3	Verifying DIO Drop Adapter Configurations .....	59
4.4.4	Verifying TIO Module Configurations .....	59

<b>Appendix A</b>	
<b>Checking DIO Drop Adapter Status</b>	<b>61</b>
A .1 DIO Drop Adapter Status Information	62
A .1.1 Health Reporting by the Drop Adapter	62
A .1.2 How the Drop Adapter RAM Contents are Stored	63
A .2 Checking Drop Health With DIOH	64
A .3 Checking Drop RAM Contents With MSTR	66
A .3.1 The Drop Adapter RAM Table	67
A .4 Monitoring I/O Module Status	68
 <b>Appendix B</b>	
<b>Checking TIO Module Status</b>	<b>69</b>
B .1 TIO Module Registers	70
B .1.1 Data Registers	71
B .1.2 Configuration Register	72
B .1.3 Status Registers	73
B .2 Checking TIO Registers With MSTR	74
B .2.1 TIO Module Registers	75
B .3 Defining Write Privilege to TIO Modules	76
 <b>Appendix C</b>	
<b>Comparing Modbus Plus I/O to Remote I/O</b>	<b>77</b>
C .1 Comparing Modbus Plus I/O and RIO Networks	78
C .2 Comparing Modbus Plus I/O and RIO Throughput	80
 <b>Glossary</b>	<b>83</b>
 <b>Index</b>	<b>91</b>

## Illustrations

Figure 1	Modbus Plus I/O System Overview .....	3
Figure 2	Network Option Module .....	5
Figure 3	DIO Drop Adapter and TIO Module .....	5
Figure 4	Basic Network Configuration (Single Cable) .....	8
Figure 5	Basic Network Configuration (Dual Cables) .....	8
Figure 6	Network Expansion Example: RR85 .....	9
Figure 7	Network Expansion Example: FR85 .....	10
Figure 8	Example: Token Rotation Sequence .....	13
Figure 9	Example: Separating Networks by Function .....	25
Figure 10	Example: Multiple Networks for Optimizing Speed .....	26
Figure 11	Example: Network With ModConnect Devices .....	28
Figure 12	Example: Network for Deterministic I/O Timing .....	30
Figure 13	Token Rotation Time: Network for I/O Servicing .....	31
Figure 14	Example: Network With Two Paths in Use .....	32
Figure 15	Example: Network With Two Paths in Use .....	34
Figure 16	Example: Multifunction Network With Programmer Node .....	36
Figure 17	Token Rotation Time: Network With Programmer .....	38
Figure 18	Example: Multifunction Network With Mixed Nodes .....	40
Figure 19	Token Rotation Time: Network With Mixed Nodes .....	42
Figure 20	Estimated Token Reconstitution Time (One Node Dropped) .....	44
Figure 21	Estimated Token Reconstitution Time (Two Nodes Dropped) ...	45
Figure 22	Network Model for Measuring Throughput Time .....	49
Figure 23	Estimated Throughput Time: Program Size and Modules/Drop .	50
Figure 24	Estimated Throughput Time: Program Size and Modules/Drop .	51
Figure 25	Estimated Throughput Time: Program Size and Modules/Drop .	52
Figure 26	Multiple Controllers With I/O Drops .....	54
Figure 27	Multiple Controllers With TIO Modules .....	55
Figure 28	Startup Example With Multiple Controllers .....	56
Figure 29	DIOH Block Overview .....	64
Figure 30	MSTR Block Overview .....	66
Figure 31	Module Status Example .....	68

Figure 32 Modbus Plus TIO Module Registers Layout ..... 70

Figure 33 Input/Output Registers Bit Layout ..... 71

Figure 34 MSTR Block Overview ..... 74

Figure 35 How the Controller Solves for Modbus Plus I/O and RIO Networks 78

Figure 36 Comparison of Modbus Plus I/O and RIO ..... 80

# Chapter 1

## Introducing Modbus Plus I/O

---

- ☐ Introducing the Modbus Plus I/O System
- ☐ How the I/O System is Organized
- ☐ Network Terminology
- ☐ Network Media
- ☐ Network Cable Layouts
- ☐ How the I/O Network Operates
- ☐ Processes During Network Startup
- ☐ Ownership of the Write Privilege to Drops
- ☐ How I/O Messages are Transacted

## 1.1 Introducing the Modbus Plus I/O System

---

The Modicon Modbus Plus Input/Output system is a local area network that provides distributed control of input/output devices in industrial processes. The network enables Modicon TSX Quantum Automation Series controllers to communicate with field devices in a wide range of industrial applications.

Modbus Plus Input/Output (I/O) networks provide a cost effective and flexible method for servicing remote input/output device sites in the industrial control application. This book provides a comparison of the performance of Modbus Plus I/O and Modicon's Remote I/O network.

### 1.1.1 Components of the I/O System

The Modbus Plus I/O system consists of five kinds of components:

- A Modicon Quantum Automation Series controller, running the user application. The controller has a built-in Modbus Plus port which enables it to operate as a master node on the I/O network.
- Optionally, up to two Modbus Plus Network Option Modules (NOMs). These can be installed in the controller's backplane, and operate as master nodes on up to two additional I/O networks.
- Distributed I/O (DIO) Drop Adapters, installed in backplanes at remote field device locations. Each adapter can control up to 15 I/O modules installed in its backplane, and can furnish operating power to the modules through the backplane connection.
- Terminal Block I/O (TIO) modules, installed on panels at remote field device locations. TIO modules can be directly connected to the I/O field devices.
- Modicon ModConnect modules, supplied by a variety of third-party manufacturers and installed at remote locations.

Figure 1 shows a Modbus Plus I/O system with three I/O networks. One network connects to the Modbus Plus port on the controller (CPU). Two additional networks connect to Network Option Modules (NOMs) in the controller's backplane.

Figure 1 shows typical network and field device connections for a Distributed I/O (DIO) Drop Adapter, Terminal Block I/O (TIO) module, and ModConnect third-party module.

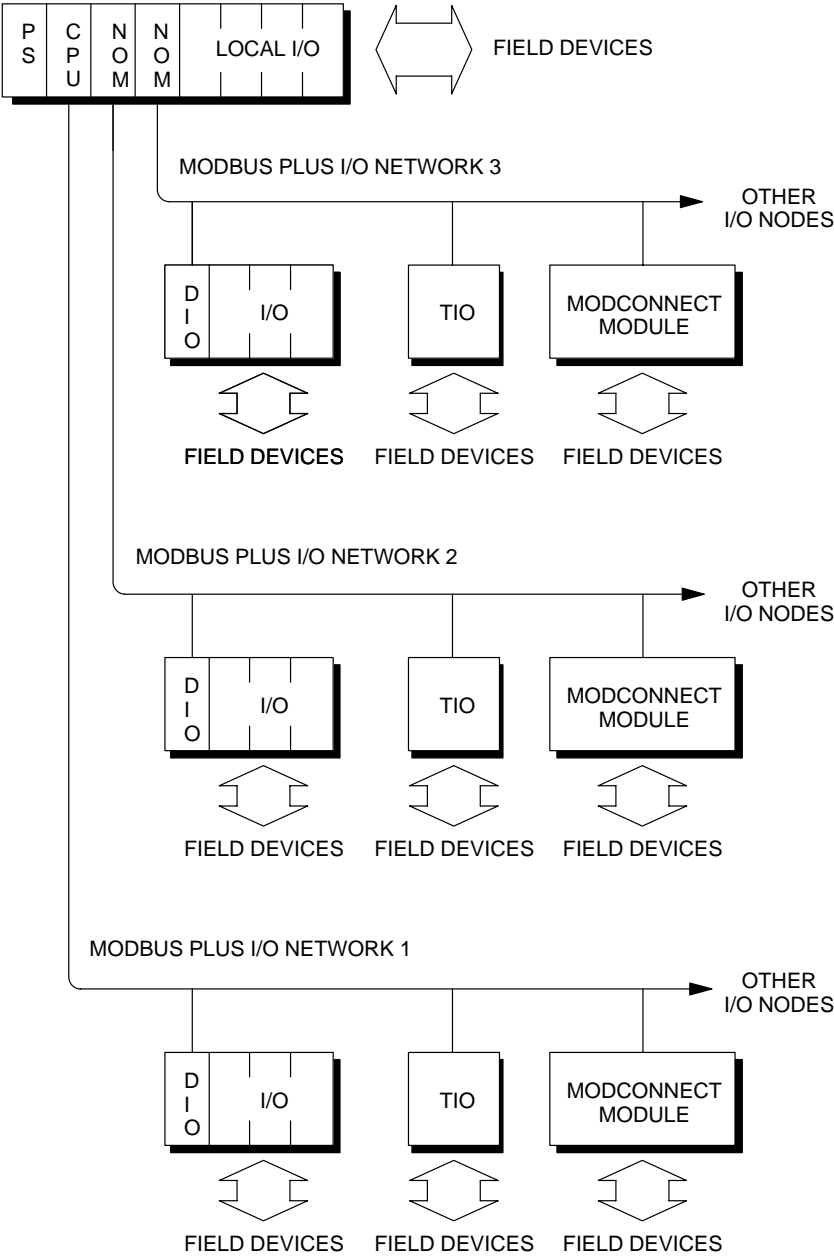


Figure 1 Modbus Plus I/O System Overview

## 1.2 How the I/O System is Organized

---

### Head Ends and the Overall I/O System

The Modbus Plus I/O system can be organized as up to three networks. Each network supports up to 64 addressable node devices.

One network can connect directly to the controller's Modbus Plus port. The controller uses one address. Therefore the controller's network can accommodate up to 63 I/O drops.

Up to two Network Option Modules (NOMs) can be installed in the controller's backplane. Each NOM uses one address on its network, allowing up to 63 I/O drops on the network. Therefore, when fully configured using a controller and two NOMs, the Modbus Plus I/O system can accommodate up to 189 I/O drops.

Modicon controllers and NOMs are available for either single-cable or dual-cable network layouts. Figure 2 shows the front view of a typical Network Option Module. Specifications for controllers and NOMs are provided in the *Quantum Automation Series Hardware Reference Guide*.

Up to 32 nodes can be connected physically to the network cable over a total length of 1500 ft (450 m). Repeater devices can extend the network to 64 addressable nodes, with 6000 ft (1800 m) of cable. Fiber optic repeaters are available for applications requiring longer distances.

### Drop Ends With Distributed I/O (DIO) Drop Adapters

Remote sites can be serviced using Distributed I/O (DIO) Drop Adapters. Drop Adapters are available for either single-cable or dual-cable layouts. At each site one Drop Adapter is installed in a backplane, together with up to 15 I/O modules. The Drop Adapter has a built-in 3.0 A power supply that provides operating power to the I/O modules.

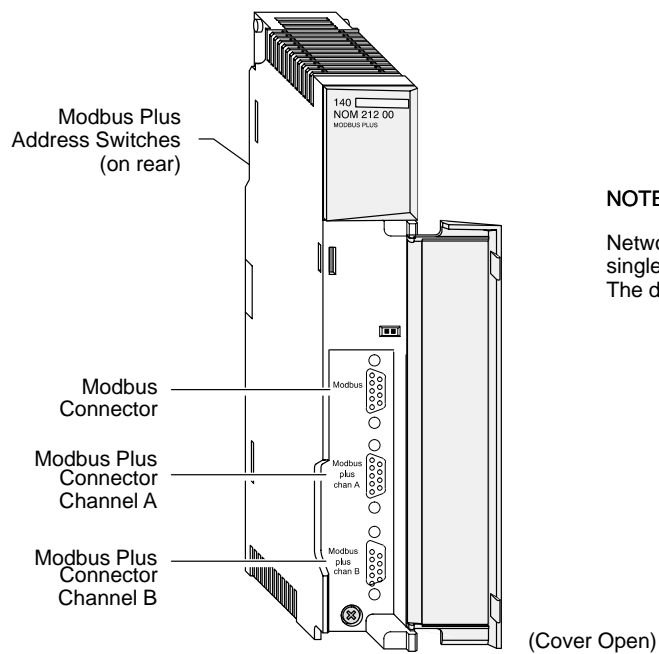
Figure 3 shows the front view of a typical DIO Drop Adapter. Specifications are provided in the *Quantum Automation Series Hardware Reference Guide*.

### Drop Ends With Terminal Block I/O (TIO) Modules

Remote sites can be serviced using Terminal Block (TIO) modules. These compact modules mount directly to a panel or DIN rail, and provide direct wiring connections to field devices at the site. TIO modules are available for single-cable layouts only, and are not applicable for use in dual-cable layouts.

Figure 3 shows the front view of a typical TIO module. Specifications are in the *Terminal Block I/O Modules Hardware Reference Guide*.

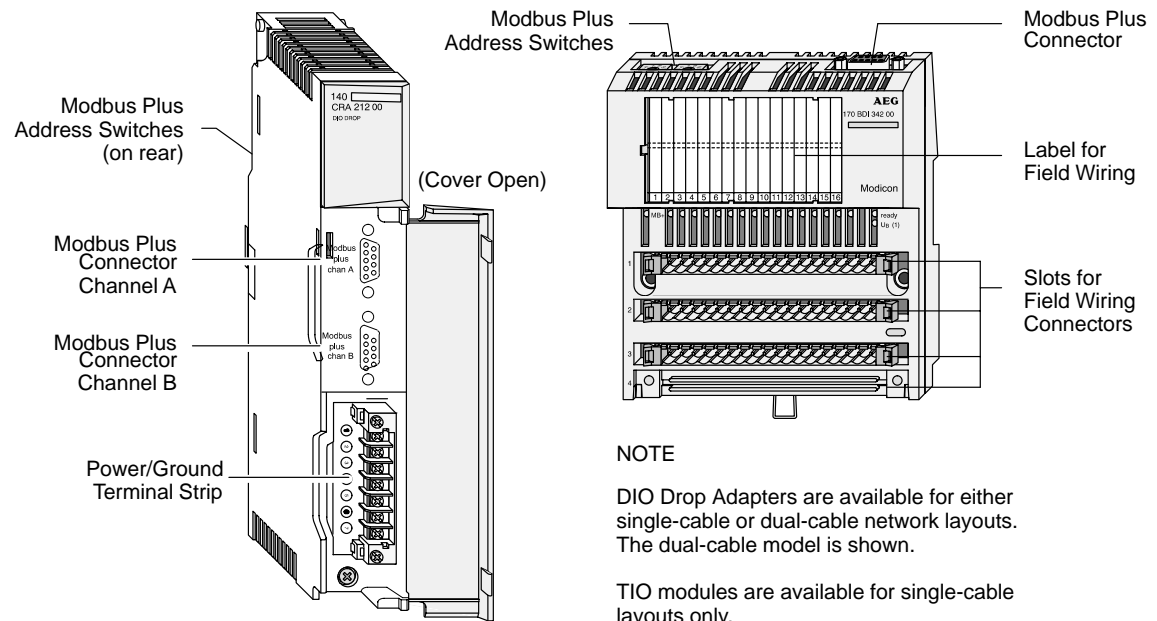




#### NOTE

Network Option Modules are available for either single-cable or dual-cable network layouts. The dual-cable model is shown.

**Figure 2 Network Option Module**



#### NOTE

DIO Drop Adapters are available for either single-cable or dual-cable network layouts. The dual-cable model is shown.

TIO modules are available for single-cable layouts only.

**Figure 3 DIO Drop Adapter and TIO Module**

## 1.3 Network Terminology

---

The following terms are used in this guide to describe the network elements:

### **Node**

This term applies to any device that is physically connected to the Modbus Plus cable. Some *nodes*, like programmable controllers, NOMs, DIO Drop Adapters, and TIO modules have network addresses and can serve as sources or destinations for messages. The Repeater is a *node* with no address, serving to extend the network.

### **I/O Master Node**

A Master node will always appear on each Modbus Plus I/O network, and may be either the programmable controller's Modbus Plus port or a NOM. Master nodes can originate data to multiple drop nodes, and can receive data from multiple drop nodes.

### **I/O Drop Node**

This term applies to the DIO Drop Adapters and TIO modules. I/O drop nodes can receive data from one master node, and can originate data to that same master node only.

### **Cable Segment**

A *cable segment* is a single length of trunk cable between two taps. Taps are passive devices that provide connections for cable segments, and for a drop cable to the I/O drop node. On dual-cable networks, two *cable segments* run in parallel between two taps at each I/O drop location.

### **Section**

A *section* of the network is a series of nodes joined only by cable segments, with their associated taps and drop cables. The *section's* signal path does not pass through any active device (such as a Repeater). *Sections* are all part of one network, sharing the same token and address sequence. Each *section* can be up to 1500 ft (450 m) long, and can accommodate up to 32 nodes. Two *sections* may be joined through a Repeater to extend the network's cable length and node count.

### **Network**

A *network* is the total group of nodes on a common signal path that is accessed by the passing of a token. It consists of one or more sections. In its minimum configuration a Modbus Plus I/O network consists of one controller (the master node) and one or more I/O drop nodes. Up to two NOMs can be installed in the backplane with the controller, allowing the user application to access a total of three networks.

## 1.4 Network Media

---

Modbus Plus I/O networks use standard Modbus Plus cable media. Complete details for the design and installation of the cable system are provided in the *Modbus Plus Planning and Installation Guide*. Refer to that guide for a listing of components and part numbers. Here is an overview to assist in your preliminary planning of the I/O network.

### 1.4.1 Trunk Cable

The network trunk cable is a twisted-pair shielded cable that is run in a direct path between successive nodes. The network consists of one or more cable sections, with any section supporting up to 32 nodes at a maximum cable distance of 1500 ft (450 m). Sections can be joined by Repeaters to extend the network's length and to support up to 64 nodes.

The minimum cable length between any pair of nodes must be at least 10 ft (3 m). The maximum cable length between two nodes is the same as the maximum section length of 1500 ft (450 m).

On dual-cable networks, the cables are known as cable A and cable B. Each cable can be up to 1500 ft (450 m) long, measured between the two extreme end devices on a cable section. The difference in length between cables A and B must not exceed 500 ft (150 m), measured between any pair of nodes on the cable section.

### 1.4.2 Taps and Drop Cables

Nodes are connected to the cable by means of a drop cable and tap. The tap provides 'through' connections for the network trunk cable, 'drop' connections for the cable to the node device, and a grounding terminal.

At each inline drop location on a cable section, two lengths of trunk cable are routed to the tap, providing the 'through' connection for the trunk cable. At the two end drop locations on a cable section, only one length of trunk cable is routed to the tap.

The tap contains a resistive termination that is connected by internal jumpers. The tap at each end of a network section must have the termination in place to prevent signal reflections. All of the taps that are inline on the cable section must have their jumpers removed (open).

Drop cables are available in 8 ft (2.4 m) and 20 ft (6 m) lengths.

## 1.5 Network Cable Layouts

Complete details for the design and installation of the cable system are provided in the *Modbus Plus Planning and Installation Guide*. Here is an overview to assist in your preliminary planning of the I/O network.

### 1.5.1 Basic Cable Layouts

Networks can be designed using either a single-cable or dual-cable layout. Figure 4 shows one section of a network with four nodes connected in a basic single-cable configuration.

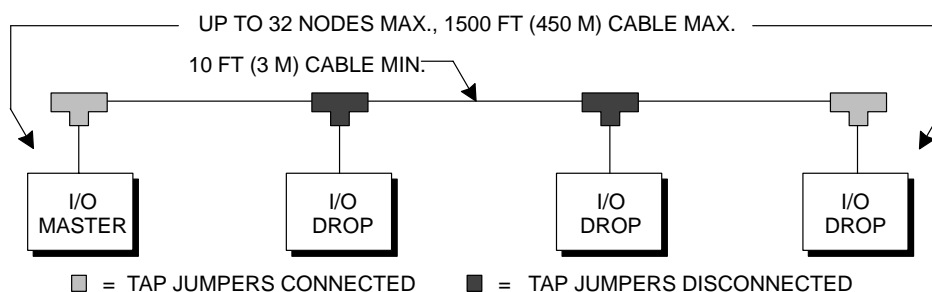


Figure 4 Basic Network Configuration (Single Cable)

Dual-cable layouts offer increased protection against cable faults or excessive noise bursts on either cable path, by continuing to process error-free messages on the alternate path. Figure 5 shows one section of a network with four nodes connected in a basic dual-cable configuration.

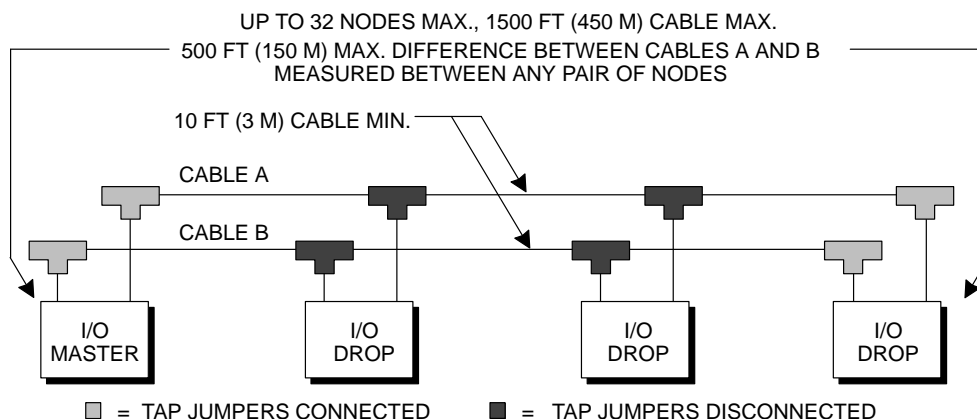


Figure 5 Basic Network Configuration (Dual Cables)

## 1.5.2 Repeaters for Network Expansion

If your network requires more than 32 nodes, or more than 1500 ft (450 m) of cable, you can install Repeaters to expand the network.

In addition to their use in expanding the network, Repeaters can be applied in plant environments that have high levels of electrical interference. Repeaters at key points in the cable system can help to maintain an excellent signal to noise ratio on the network.

### RR85 Repeaters for Trunk Cable Connections

Modicon RR85 Repeaters are available for expanding your network through extension of the network trunk cable. Up to three RR85 Repeaters can be present in the cable path between any pair of nodes that will communicate with each other. For more details about the application of RR85 Repeaters, see the *Modbus Plus Network Planning and Installation Guide*.

Figure 6 is an example of a network with two cable sections joined by one RR85 repeater.

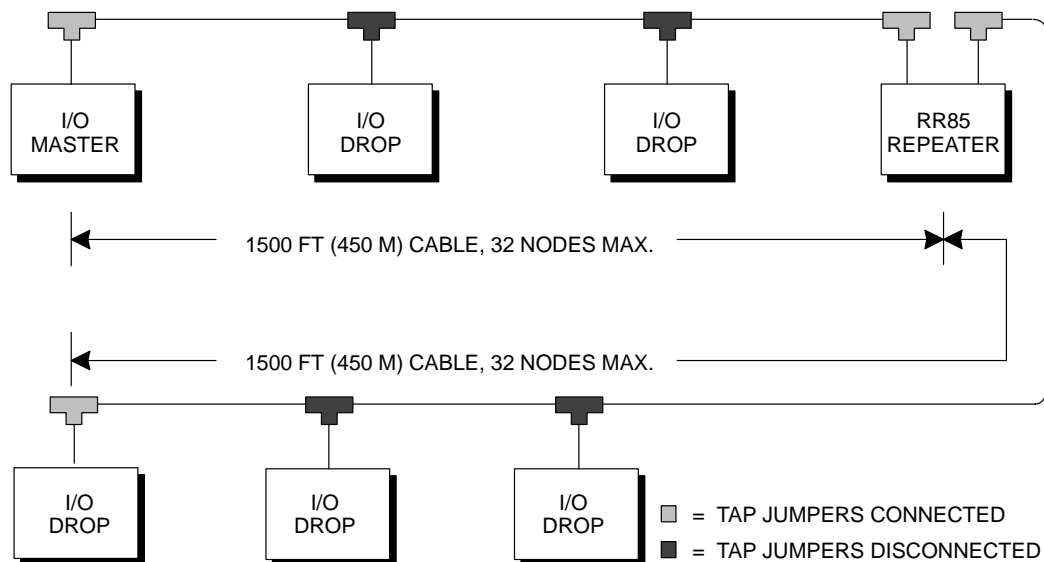


Figure 6 Network Expansion Example: RR85

### FR85 Repeaters for Fiber Optic Connections

Modicon FR85 Repeaters are available for extending your network through fiber optic links. These devices provide point-to-point links of up to 6700 ft (2 km) using 50  $\mu\text{m}$  fiber cable, and up to 10,000 ft (3 km) with 62.5  $\mu\text{m}$  or 100  $\mu\text{m}$  cable. For details about the application of FR85 Repeaters, see the *490 NRP Series Fiber Optic Repeaters User Guide*.

Figure 7 is an example of a network with two cable sections joined by a pair of FR85 repeaters.

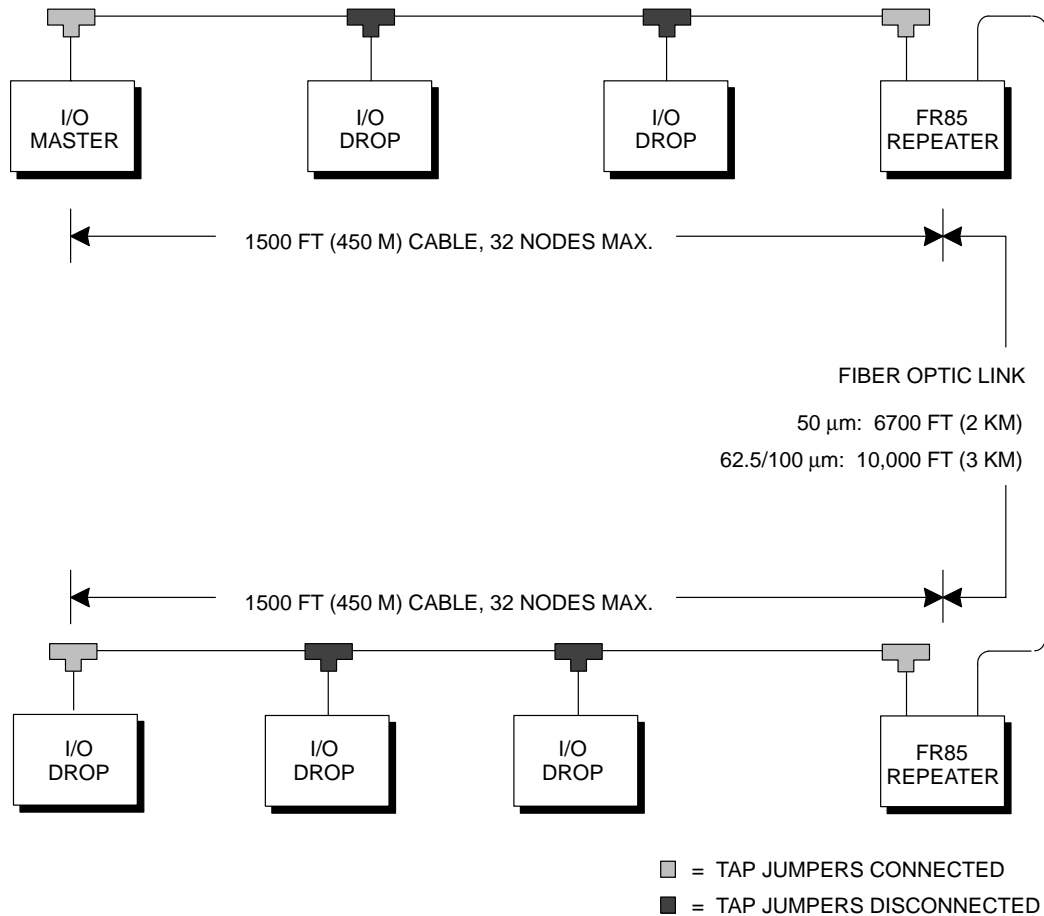


Figure 7 Network Expansion Example: FR85

### 1.5.3 Bridges For Standard Modbus Plus Messages

The BP85 Bridge Plus device joins two Modbus Plus networks together. BP85 devices can be present on I/O networks to allow the programmable controller to process standard Modbus Plus messages with nodes on another network. For example, nodes on other networks can be accessed through the BP85 device using MSTR function blocks in the controller's application program. For more details about the use of BP85 devices for bridging networks, see the *Modbus Plus Network Planning and Installation Guide*.

#### **Bridges Not Applicable for I/O Network Transactions**

On Modbus Plus I/O networks, communications between the master node (the controller or NOM) are transacted during the network's token pass. Tokens do not pass through BP85 bridges. Therefore BP85 devices are not applicable for direct transactions between a master node on one network and I/O nodes on another network.

## 1.6 How the I/O Network Operates

---

### 1.6.1 Overview of the Logical Network

Network nodes are identified by addresses that are defined by the user. Addresses are assigned by setting hardware switches within each node device. Each node's address is independent of its physical site location. Addresses are within the range of 1 to 64 decimal, and do not have to be sequential. Duplicate addresses are not allowed.

Network nodes function as peer members of a logical ring, gaining access to the network upon receipt of a token frame. The token is a grouping of bits that is passed in a rotating address sequence from one node to another. In applications using multiple networks, each network maintains its own token rotation sequence independently of the other networks. Tokens are not passed between networks.

In addition to the node addresses set in hardware switches, the various nodes must also be known to the user application program. When the controller is initially configured through panel software, a DIO Map and Peer Cop table are setup to identify each node's network address and the logic references for its data transfers.

### 1.6.2 The Token Rotation Sequence

The token sequence is determined by the node addresses. Token rotation begins at the network's lowest-addressed active node, proceeding consecutively through each higher-addressed node, until the highest-addressed active node receives the token. That node then passes the token to the lowest one to begin a new rotation.

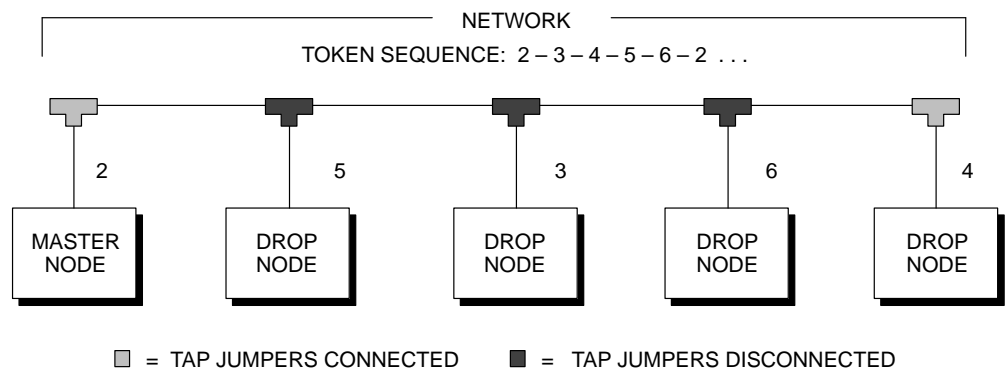
If a node leaves the network, a new token-passing sequence will be established to bypass it, typically within 100 milliseconds. If a new node joins, it will be included in the correct address sequence for the passing of the token, typically within five seconds.

The process of deleting and adding nodes is transparent to the user application, however you should be aware of the impact these times may have on the servicing of your I/O devices.

Chapter 2 provides guidelines for planning networks in which I/O device timing must be controlled and maintained within repeatable intervals. Chapter 3 provides examples for estimating network performance.



Figure 8 shows the token rotation sequence in a network with five nodes. Note that the master node (the programmable controller or NOM) is always assigned the lowest address on the network. Token rotation is determined by the assigned node addresses, and is not related to the physical site locations of the nodes.



**Figure 8 Example: Token Rotation Sequence**

## 1.7 Processes During Network Startup

---

### 1.7.1 How the I/O Nodes are Known to the Application

Prior to running the controller application, I/O networks are configured logically within the controller using an appropriate programming device and software. In the configuration process, 'DIO Map' identifications are assigned to the controller, NOMs, and all drops with DIO Drop Adapters. Modbus Plus 'Peer Cop' identifications are assigned to the controller, NOMs, and drops with TIO modules. Modicon Modsoft or ConCept panel software can be used for this purpose.

DIO mapping techniques are the same as those for Remote I/O (RIO) mapping, in which head end (master) device and drop end (remote) device configurations are specified. During the controller configuration, the controller and NOMs are identified by their backplane slot positions. The controller can also be identified as slot zero.

DIO Adapters and TIO modules are identified by the combination of their controller or NOM slot position and their network node addresses. The controller's DIO Map table identifies the DIO Adapters present at the Modbus Plus port of the controller and NOMs, and specifies the data references of messages to be transacted with each node. The controller's Peer Cop table identifies the TIO modules and their data references.

### 1.7.2 How the DIO Drop Adapters Become Configured

During startup the programmable controller downloads configuration information for storage into the DIO Drop Adapters.

The controller downloads a DIO Map to each DIO Drop adapter, either directly or through its local NOMs. The Drop adapter's DIO Map is a section of the controller's DIO Map that pertains to the individual drop. Each Drop Adapter stores its own DIO Map.

This process sets the privilege within the drop adapter to allow only that one master node to perform future DIO Map downloads to the drop. Subsequent attempts by other nodes to write that drop adapter's DIO Map will fail. If, however, the master node is removed from the network, the drop adapter will accept a new DIO Map from a new master. Section 1.8.1 describes the management of write privileges in the Drop Adapters.

When the drop adapter's DIO Map has been received and stored, the drop will start processing outgoing messages at its next token. Even if a

drop has not been configured to transmit data, it will still transmit a minimum of two words containing its DIO Map checksum and current health status.

### **1.7.3 How the TIO Modules Become Configured**

As the network becomes active the controller begins transacting data immediately with TIO modules as specified in its Peer Cop table. Transactions are handled in the network's node address sequence, with the controller or NOM assigned the lowest node address and initiating write transactions to the TIO nodes. Each TIO node transmits its data in turn as it acquires its first token.

Each output module has a register that contains a timing parameter for holding its current output states, in the event that the module is not updated by a new write command. If the holdup time expires, the module's outputs will go to a logical 0 (zero).

The holdup time value can be modified in the range of 300 ms ... 60 s, in 10 ms increments, using MSTR blocks in the application program. The default holdup time is 1 second.

## 1.8 Ownership of the Write Privilege to Drops

---

### 1.8.1 Write Privilege to DIO Drop Adapters

After a Drop Adapter has been downloaded with its DIO Map, and a data message of the correct length has been received by the Drop Adapter, two kinds of write privilege exist in the drop: privilege to write a new DIO Map to the drop; and privilege to write new data to the drop.

Only one master node can own the write privilege to a given drop. This includes the privilege for writing both the DIO Map and data messages to the drop. The master node which writes the DIO Map to the drop must be the sole node which writes data messages to the drop.

Other master nodes can be configured for read-only privilege, and can read messages from the drop. There is no restriction on the quantity of masters that may have read-only privilege from any drop.

Thus a single I/O network can contain multiple master devices, each of which receives read (incoming) data from any combination of drops. However only one master device may have write (outgoing) data privilege to any given DIO Drop Adapter.

## 1.8.2 Write Privilege to TIO Modules

Only one master node can own the write privilege to a given drop. When a master node writes to a TIO module, the module will reserve further write privilege to that node for a duration of 60 seconds, before timing out. If the master node continues to write to the module within 60-second intervals, that node retains the write privilege to the module. If a timeout occurs, the module will accept a write from another node.

The 60-second timeout value is fixed, and is not accessible to the application. Note that the timeout process applies only to the write privilege to a module. Reading of input data or status from the module is allowed to any node.

## 1.9 How I/O Messages are Transacted

---

### 1.9.1 Handling I/O Messages

Data sources and destinations for DIO Drop Adapters are specified in the controller's DIO Map and in the DIO Maps that are downloaded to the drops during startup of the network. Data sources and destinations for TIO modules are specified in the controller's Peer Cop table.

While the controller or NOM holds the token, it transmits data to Drop Adapters for delivery to their output modules, and to TIO modules for delivery to their output points. When each Drop Adapter has the token, it transmits data from its input modules to the controller or NOM. When each TIO module has the token, it transmits data from its input points to the controller or NOM.

Messages consist of a defined quantity of 16-bit words. The maximum amount of words that can be transacted differs between master nodes (the controller and NOMs), DIO Drop Adapters, and TIO modules. Word counts for each type of node are further explained below.

### 1.9.2 Message Sizes – I/O Master Nodes

Master nodes can transmit up to 500 words of data, and can receive up to 500 words of data per scan. This is the maximum amount of data which can be specified in the controller's configuration. The amount of words that can be transacted with each type of drop node depends upon the node type as outlined in Sections 1.9.3 and 1.9.4.

### 1.9.3 Message Sizes – DIO Drop Adapter Nodes

Drop Adapter nodes can each receive up to 32 words, all of which may contain data to the node's output modules.

Drop Adapter nodes can transmit up to 32 words in their total message. The first two words in the message are reserved for reporting the drop node's DIO Map checksum and health status. The remaining words (up to 30 total) contain the drop node's data from its input modules.

### 1.9.4 Message Sizes – TIO Module Nodes

TIO module nodes can receive or transmit either 2 or 3 words in their data message, as determined by the specific module type. Appendix B summarizes the data handling requirements for each TIO module.

### 1.9.5 Handling Standard Modbus Plus Messages

In addition to its use for servicing I/O drops, the network can be used as a standard Modbus Plus link. For example, the controller and NOMs can transact standard Modbus Plus Peer Cop and MSTR function block messages with other nodes that are not part of the I/O application. MSTR messages can be passed through Bridge Plus devices to nodes on other networks, or through Bridge Multiplexers to standard Modbus or serial devices.

If you add non-I/O nodes to the network, the increased node count will affect the network token rotation time and speed of I/O communications. Because of their potential for randomly timed access to the network, non-I/O nodes like programmers and User Interfaces (UIs) can prevent the network from achieving deterministic timing in the control of I/O processes.



**Recommendation:** If you are planning your application for fully deterministic timing of I/O processes, your I/O network should include only the kinds of drop devices which perform the I/O application. Those devices are the DIO Drop Adapters and TIO modules.

Chapter 2 provides examples of configurations for both deterministic and multifunction timing applications.





# Chapter 2

## I/O Network Configuration Options

---

- ☐ Setting Your Network Performance Goal
- ☐ An Overview of Configuration Options
- ☐ Networks for Deterministic I/O Servicing
- ☐ The Effects of Paths on Network Timing
- ☐ Networks With a PC/SA85 Node
- ☐ Networks With a Programmer Node and User Interface Node
- ☐ Networks With ModConnect Devices
- ☐ Estimating Token Reconstitution Time
- ☐ Verifying Your I/O Configuration

## 2.1 Setting Your Network Performance Goal

---

### 2.1.1 Factors to Consider

Several factors will affect the speed of transacting messages between the nodes on your Modbus Plus I/O network. These include:

- ☐ The kinds of node devices you plan to include on the network, and the timing of their access to the network.
- ☐ The network node count, average message size, and resulting token rotation time.
- ☐ The size of the controller application program, and the scan time of the controller.

### 2.1.2 Planning Your Application

You should consider your requirements for the speed of data transfers, and for the repeatability of timing of those transfers (deterministic or non-deterministic timing).

When you plan your network application, you should be aware of the impact of including various kinds of devices. For example, if your requirement is for fully deterministic timing of data transactions between your master node and I/O drops, then you should not include any types of devices on that network which may alter network timing. Place devices like host-based network adapters, User Interfaces (UIs), and programmers on a separate network.

### 2.1.3 How Your Network Layout Affects Node Access

When the network is initialized, each node becomes aware of the other active nodes. Each node builds a table identifying the other nodes. Initial ownership of the token is established, and the token rotation sequence begins.

The fast token rotation time of small networks allows rapid transfer of high-priority data. This facilitates time-critical messaging to nodes that are tightly linked in a portion of your application. Your choice between laying out your application as one large network, or as several smaller networks, affects the timing of token rotation, and therefore influences each node's access time for data.

## 2.1.4 Deterministic I/O Servicing or Multi-Function Servicing

The performance of your I/O network will be directly affected by the kinds of devices you are planning to use on the network.

You may choose to construct your Modbus Plus network application in one of two ways. You can provide for deterministic timing to service your I/O processes at predictable timing intervals, or you can provide for non-deterministic timing for the servicing of multi-function nodes.

If your requirement is for deterministic timing of network transactions that must occur within repeatable time intervals, you must plan to include only the kinds of node devices that will adhere to this kind of timing. These include the CPU or NOM, and the DIO Drop Adapters and TIO modules.

Devices that may not produce deterministic timing of transactions include User Interfaces (UIs), programming panels, and host-based network adapters. Typically these devices access the network upon demand of a person or an external process. Therefore the timing of their access, and their resulting effect on the network token rotation time, are not fully predictable or repeatable in your application.

If your application must handle both functions (I/O processes to be serviced at repeatable timing intervals, plus devices to be serviced on demand), then you should consider incorporating a separate network for each function.

This chapter provides guidelines for various network layouts. You should review these guidelines for the design of your application.



**Warning! Modbus Plus is a network which can offer either deterministic I/O servicing, or non-deterministic multi-function servicing of I/O, Programming, User Interface, and third party ModConnect devices. Unintended operation may result if proper network architecture issues are not considered.**

## 2.2 An Overview of Configuration Options

---

As the designer of your I/O servicing network you are responsible for selecting one of two configuration options. The option you select is determined by the servicing requirements of your application.

### 2.2.1 Option 1: Deterministic I/O Device Servicing

When using Modbus Plus as an I/O network, the node devices should include only those types that perform the I/O application -- the CPU or NOM, DIO Drop Adapters, and TIO modules. For truly deterministic I/O servicing do not connect any other kinds of devices to the network. Do not allow nodes to be added or deleted on the network while the I/O machine or process is running.

Section 2.3 shows an example of a network designed for deterministic timing in the servicing of I/O devices.

### 2.2.2 Option 2: Multi-Function Device Servicing

Modbus Plus networks can be used for communication among many kinds of node devices. You should be aware of each device's requirement for access to the network. You should also be aware of the impact each device will have on the timing of your network communication, including the servicing of I/O nodes.

Section 2.4 describes the kinds of access required for various devices. Sections 2.5 and 2.6 show examples of multi-function networks.

#### **Programmers**

A programmer may be connected temporarily to a spare connector on an active network, however you must use caution when adding or deleting these devices as this may cause an interruption in I/O servicing. The duration of the interruption depends greatly on the number of nodes on the network and their assigned addresses. In general, the lower the addresses on the network the shorter the interruption will be. If the use of a programmer is being considered, the device should be connected to another network, or to the I/O network on a permanent basis.

In addition, the use of programmers may add to the token rotation time of the network. For example, Modsoft can add up to 6 milliseconds.

### User Interfaces (UIs)

UI devices typically communicate with the CPUs on a periodic polled basis, which generates network traffic that may vary over succeeding token rotations. Therefore, to ensure proper I/O servicing, you should understand the worst case traffic these devices can generate. Typically a UI can add 6 milliseconds or more to the token rotation time.

### Host-Based Adapters and ModConnect Devices

Many adapters and third party devices exist which may be attached to a Modbus Plus network. To ensure proper network performance, it is important that you be aware of the impact these devices may have on token rotation time. Please consult the appropriate Vendor's literature.

## 2.2.3 Separating Networks by Function

Through the use of Network Option Modules (NOMs), separate networks can be dedicated to I/O and to other types of communications.

By placing the I/O and non-I/O portions of your application into separate networks, you can tailor each network to its data handling and speed requirements. This can also allow you to more easily develop, program, and test each portion of your application.

Figure 9 shows three networks separated according to their functions.

- ☐ The CPU controls a network designed for deterministic timing of I/O processes. Only the CPU and I/O drops are present on this network.
- ☐ NOM1 is on a network for handling communication with standard Modbus Plus nodes, such as a host-based adapter or another CPU.
- ☐ NOM2 is on a multi-function network with a User Interface and I/O. Through the NOM, the CPU's application services the I/O drops.

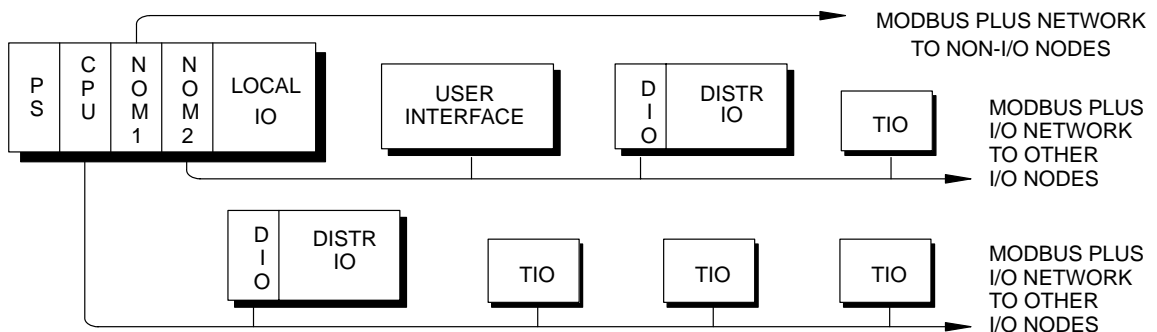


Figure 9 Example: Separating Networks by Function

## 2.2.4 Minimizing Node Counts for Speed

Where time-critical data must be handled, the network node count should be kept as low as possible to minimize token rotation time and to give each node timely access to the network. If many I/O drops are required, they can be distributed into several smaller networks.

Figure 10 shows an example of an application that requires 16 I/O drops and one remote User Interface. The application is organized into three networks, each with a minimum node count to allow rapid rotation of the network token. The figure also shows assignment of node addresses.

Networks 1 and 2 service I/O processes with time-critical data.  
Network 3 services multi-function devices in which timing is less critical.

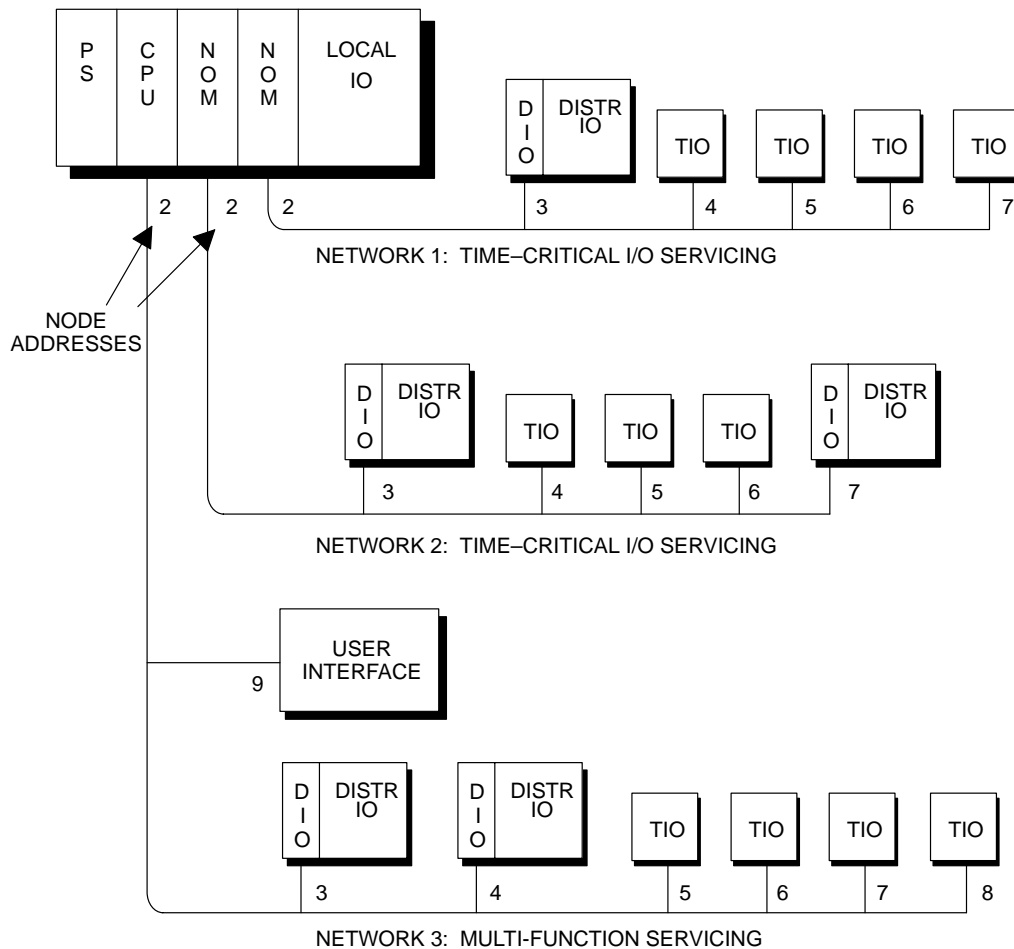


Figure 10 Example: Multiple Networks for Optimizing Speed

## 2.2.5 Assigning Node Addresses

The node addresses you assign to your devices will affect the way your network reacts to the event of a dropped or added node.

Figure 10 shows an example of address assignments on three networks. Note that the addresses are assigned sequentially on each network, with the CPU or NOM node having the lowest address.

You should follow the addressing guidelines in the Figure 10 example. Assign node addresses sequentially, with the lowest address always assigned to the network CPU or NOM. This should always be address 2. Address 1 is reserved to facilitate network maintenance (see Section 2.7). The next addresses are assigned to the I/O drops. Remaining addresses are assigned to any other types of devices on the network.

You assign addresses to nodes to make them known to your application software (e.g., as sources and destinations for messages). Addresses are not related to the physical locations of the nodes on the network cable.

Each Modbus Plus node has its address assigned in hardware switches. You should coordinate the node addresses for your entire application to ensure that each network has its nodes addressed sequentially without any gaps or duplication.

Section 2.7 provides recommendations for assigning node addresses. Follow those recommendations to handle the possibility of node devices dropping out and rejoining the network.

## 2.2.6 Connecting or Disconnecting Active Nodes

Before you allow any node to be added or deleted on an active network, you should be aware of the effect this causes on network timing. Adding or deleting a node forces the network to initialize and reconstitute its token sequence. This causes a detrimental change to the timing of I/O transactions, and invalidates deterministic servicing of I/O processes.

Section 2.7 details the effects of nodes being removed or connected to an active network.



**Caution:** Apply sufficient margin in your application to provide for worst-case conditions, such as multiple nodes leaving or rejoining the network due to area power faults. The removal or connection of nodes on an active network is not recommended to be performed on a routine basis.

## 2.2.7 Considerations for Using Bridge Mode

Modicon Quantum controllers provide a Bridge Mode for bridging between a Modbus device and remote nodes on Modbus Plus. The bridge is made between the controller's Modbus port and its Modbus Plus port. A switch on the controller's front panel sets the Bridge Mode.

Figure 11 shows an example of a Modbus Plus network with two CPUs (A and B). A Modicon P230 Programmer is connected to the Modbus port at CPU A. Using that CPU's Bridge Mode, the Programmer can logically attach to CPU B. The P230 can then program CPU B as if it were attached locally to that CPU.

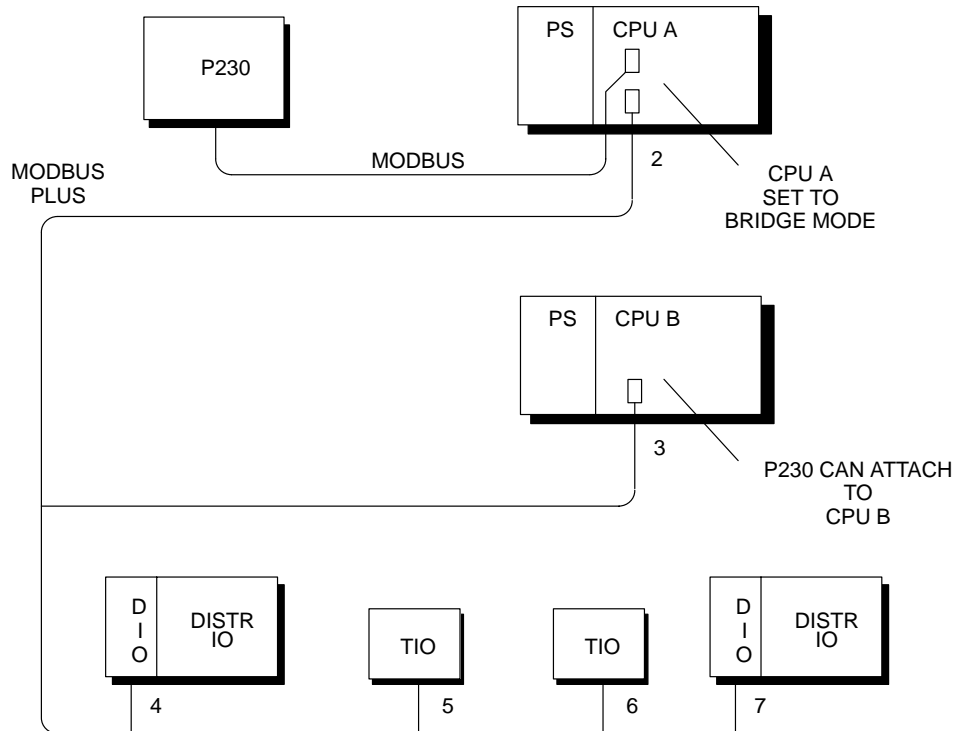


Figure 11 Example: Network With ModConnect Devices

The use of Bridge Mode causes a delay in the servicing of I/O drops while the Bridge Mode commands are being executed. For example, if CPU B has transactions in progress with any of the I/O drops, the transactions will be delayed while CPU B executes its commands from the P230.





**Recommendation:** If you plan to use Bridge Mode on the same network that services I/O processes, you should be aware of its possible impact on the timing of those processes. If your application requires fully deterministic timing of I/O processes, you should avoid the use of Bridge Mode on networks that are servicing I/O devices.

## 2.2.8 Defining Output Module Default States

Each message to an output module sets the module's output circuits to a new state. Modules retain that state until they receive a new message, or until a timeout condition occurs. You can configure each module to define the length of time that it will hold its present state. This time is termed the module's 'holdup time'.

If a module loses communication for a period of time long enough for its holdup time to expire, its output circuits will assume a default state. Holdup times and default states are managed differently between DIO Drop Adapter modules and Terminal Block I/O modules.

### DIO Drop Adapter Modules

You can configure each output module's holdup time in the controller's DIO Map. You can also configure the module to assume one of the following states if its holdup time expires:

- ☐ Hold Last Value: The module retains its present output level (e.g., ON, OFF, or an analog value).
- ☐ Set Predefined State: The module assumes an output level that you have defined in the controller's DIO Map.

### TIO Modules

You can use an MSTR function block to load a holdup time value into an output module's holdup time register. TIO output modules will assume an output state of logical 0 (zero) if their holdup time expires.

Refer to the guidebook for your panel software for further information about setting I/O module default states.

## 2.3 Networks for Deterministic I/O Servicing

Figure 12 illustrates a network designed for deterministic timing of I/O processes. The I/O network consists only of the CPU and I/O drops. A User Interface (UI) device is connected to a separate network at the NOM port.

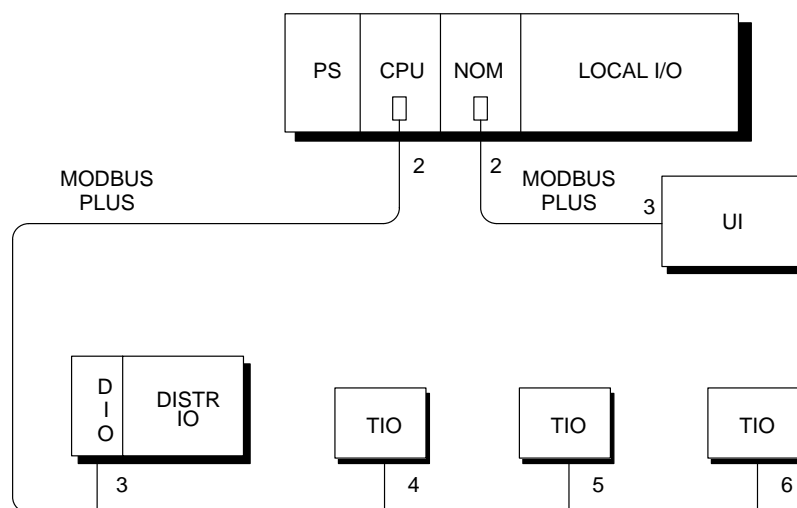


Figure 12 Example: Network for Deterministic I/O Timing

### 2.3.1 CPU/NOM Servicing Time

In Figure 12 the I/O network is connected to the CPU Modbus Plus port. The UI is connected to the NOM Modbus Plus port. When a NOM is used, additional CPU scan time is required to service the NOM. Typically a NOM adds 6 ... 9 ms time into the application program to service the NOM's network.

### 2.3.2 Calculating Token Rotation Time

Token rotation time is determined by the node count and the average size of messages. The formula for calculating token time for a network consisting only of a CPU and I/O nodes (as in Figure 12) is:

$$\begin{aligned} \text{Token Rotation Time (ms)} = & \\ & (\text{Node Count}) \\ & + ((\text{Node Count} - 1) * 2) * (\text{Average Message Size}) * 0.016 \end{aligned}$$

### 2.3.3 Example: Token Rotation Time

Figure 13 shows token rotation time estimates for I/O networks like the example in Figure 12. The CPU/NOM limit for a given direction of I/O data (500 words in, 500 words out) will be reached at the combinations of node count and average message size shown.

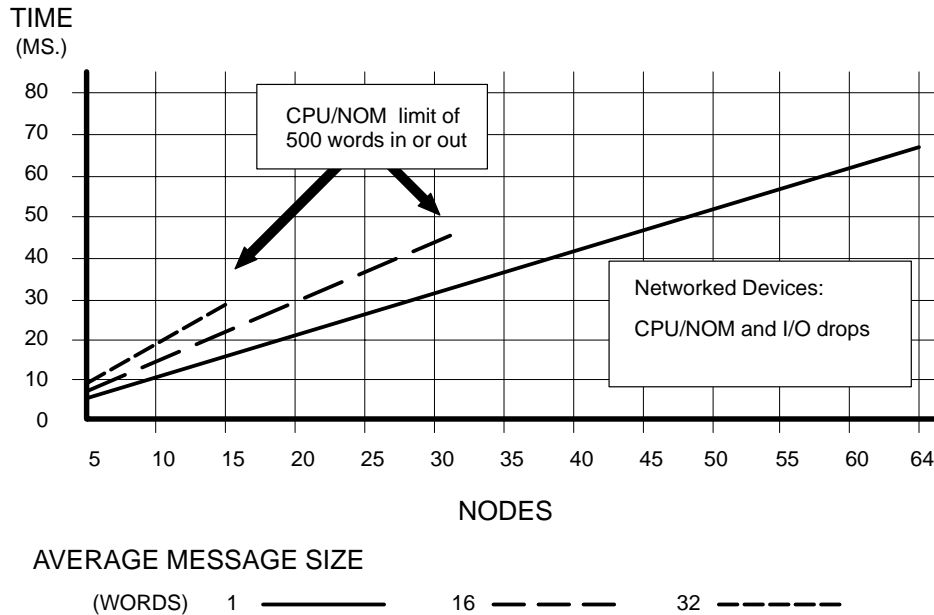


Figure 13 Token Rotation Time: Network for I/O Servicing



**Recommendation:** For truly deterministic timing of I/O servicing, follow these recommendations:

- ★ **Assigning Node Addresses** Assign the lowest address (2) to the CPU. Address 1 is reserved (see Section 2.7). Assign the next addresses to the I/O nodes, without any gaps in the sequence.
- ★ **Adding or Deleting Nodes** Your application should be able to support the downtime required to reconstitute the token rotation, in the event of node devices dropping out or rejoining the network. Section 2.7 shows how to estimate the time required to reconstitute the token rotation sequence.
- ★ **Adding Non-I/O Devices** If you require a networked UI or other non-I/O device in your application, connect it to a separate network at a NOM port. Reserve the CPU network for I/O servicing only.

## 2.4 The Effects of Paths on Network Timing

### 2.4.1 How Paths are Used

A communication 'path' is a logical connection between a source node and a destination node. A path is established automatically between the two nodes when either of these events occurs:

- The CPU's application program executes an MSTR function block to read/write data, or get/clear remote statistics, at a networked node.
- A networked Programmer device executes a remote programming command to a networked node.

### 2.4.2 How Paths Affect Timing

You should be aware of the impact on the timing of your I/O processes whenever MSTR or remote programming transactions are processed. These types of transactions can inject variable timing into the network's token rotation, reducing the determinism of I/O process timing.

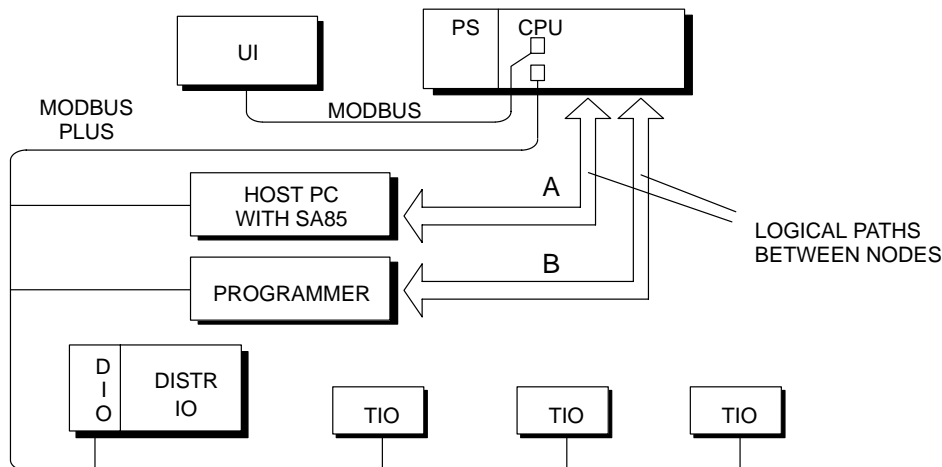


Figure 14 Example: Network With Two Paths in Use

For example, consider two events occurring on the network in Figure 14. In the CPU program, an MSTR block is executed to request data from an SA85 network adapter in a host computer. Asynchronously to this event, an operator at the Programmer node invokes a command to interrogate the CPU. Two paths (A and B) will be opened by these two events.

### **Effect of the Path A Transaction**

When the CPU program executes the MSTR to read data from the SA85, the network's token rotation time will be extended twice.

First, when the CPU holds the token, the read command must be sent to the node. Second, when the SA85 node acquires the token, and has a data response ready, it must send the response to the CPU. The remote computer's program is running asynchronously to the token rotation. The data response may be ready at the node's next token acquisition, or not until a subsequent token.

Each event (sending the command, and receiving the data response) adds time to the current token rotation.

### **Effect of the Path B Transaction**

If the Programmer is used to send programming commands (such as a Search command) to the CPU, the CPU's scan time can be extended for up to 10 ms (per scan) while the command is being executed in the CPU. This can occur over multiple scans until the command is fully executed. With the CPU's scan time extended, it is possible that new I/O data may not be available when the CPU next acquires the token.

### **Total Effect of Path A and B Transactions**

The total time for a given token rotation will depend upon the states of all concurrent transactions, and may vary between subsequent rotations. This variability in token rotation time introduces non-deterministic timing into I/O processes on the network.

## **2.4.3 Normal I/O Transactions: Paths are Not Used**

Paths are not applicable to typical DIO/TIO (non-MSTR) transactions. For example, transactions between the CPU/NOM and DIO Drop Adapters are handled as specified in the CPU's DIO Map configuration. Transactions between the CPU/NOM and TIO modules are specified in the CPU's Peer Cop configuration.

These transactions do not use MSTR blocks, and do not require a path. Each transaction is of a defined data length and occurs regularly with each token pass. By using these transactions exclusively on your I/O network, you can achieve deterministic timing of your I/O processes.

If you use MSTR blocks to retrieve status information from I/O drops or to modify their configurations, you should be aware of effects on timing. Appendix A describes use of the MSTR block for DIO Drop Adapters. Appendix B describes its use for TIO modules.

## 2.4.4 Path Types

Modbus Plus applications use the following types of paths.

Path Type (Abbreviation)	Purpose
Data Master (DM)	Opened for data read/write and get/clear remote statistics commands originated at the node.
Data Slave (DS)	Opened for data read/write commands received by the node. Incoming get/clear remote statistics transactions do not use a path.
Program Master (PM)	Opened for remote programming commands originated at the node.
Program Slave (PS)	Opened for remote programming commands received by the node.

For example, in Figure 15 an MSTR command to read data from the SA85 opens a Data Master path in the CPU, and a Data Slave path in the SA85. A command at the Programmer node to remotely program the CPU opens a Program Master path in the Programmer, and a Program Slave path in the CPU.

The *Modbus Plus Network Planning and Installation Guide* provides a detailed description of Modbus Plus transactions with the use of paths.

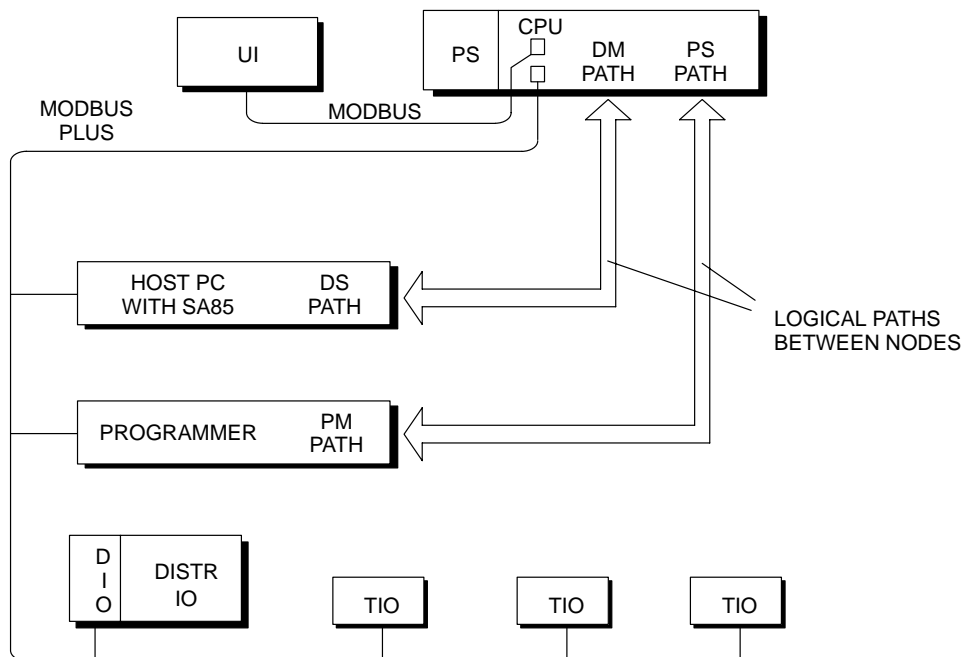


Figure 15 Example: Network With Two Paths in Use

## 2.4.5 Path Quantities

The following quantities of paths are available in Modicon products.

Product	Model	DM	DS	PM	PS
Programmable Controller	Quantum, 984	5	4	1	1
DIO Drop Adapter	All	None	8	None	None
TIO Module	All	None	8	None	None
Host-based Adapter	SA85, SM85, SQ85	8	8	8	8
Bridge Multiplexer	BM85	4	4	4	4
Programmer	P230	8	8	8	8
User Interface	PanelMate Plus	None	8	None	None

Modicon programmable controllers provide a total of five DM paths. One of these is reserved for use by the controller's Modbus port in the bridge mode between Modbus and Modbus Plus. The remaining four paths are available for use by MSTR blocks in the application program.



**Recommendation:** For the best performance in servicing I/O processes in multi-function applications, follow these recommendations:

- ★ **Separating Networks by Function** For truly deterministic timing of I/O processes, include only the CPU and I/O drop nodes on the network that controls those processes. Place devices that use MSTR or programming commands on a separate network connected to a NOM port.
- ★ **Planning for Paths** If your application requires devices that use MSTRs or programming commands to be on the same network with I/O, you should incorporate timing margins into your I/O processes to account for the worst-case timing of concurrently open paths.
- ★ **Adding or Deleting Nodes** Your application should be able to support the downtime required to reconstitute the token rotation, in the event of node devices dropping out or rejoining the network. Note that typical token reconstitution times may range up to 250 ms. Section 2.7 shows how to estimate the time required to reconstitute the token rotation sequence.
- ★ **Provide Service Taps and Drops** Provide at least one spare tap and drop cable on each network. This will allow you to connect a Programmer node for testing the network, without having to disconnect another node. Note that connecting and disconnecting the Programmer will still cause the network token sequence to be initialized and reconstituted, as described in Section 2.7.

## 2.5 Multifunction Networks With a Programmer

Figure 16 is an example of a network application that requires one networked Programmer in addition to the CPU and I/O drop nodes. The Programmer consists of a Modicon SA85 network adapter installed into a host PC. An application running in the PC/SA85 allows that node to remotely access the CPU.

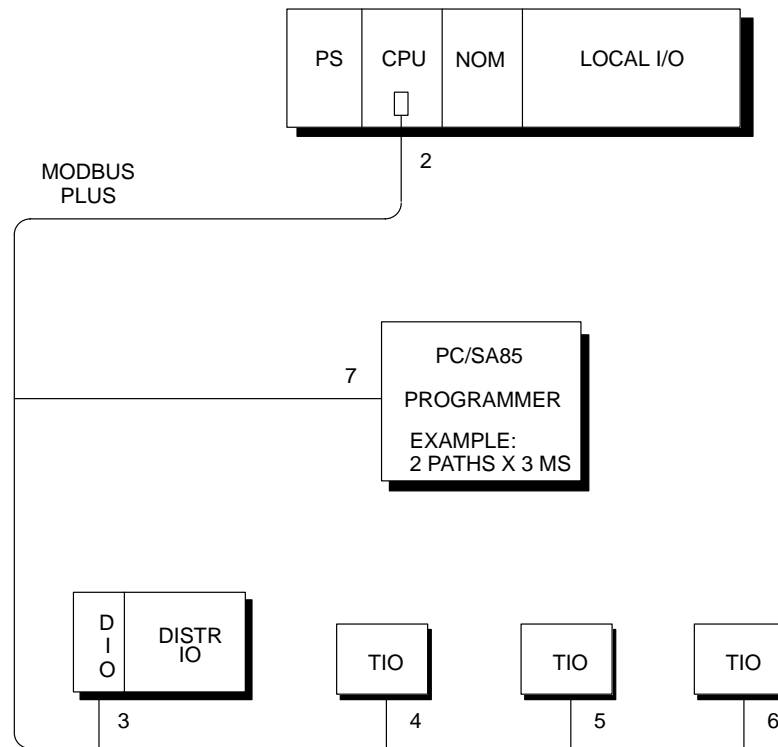


Figure 16 Example: Multifunction Network With Programmer Node

### 2.5.1 Impact on I/O Servicing

When the CPU executes commands from the PC/SA85, typically up to 10 ms can be added to the CPU scan time.



## 2.5.2 Impact on Token Rotation Time

The impact on token rotation time caused by inclusion of the PC/SA85 node will depend upon the quantity of paths the PC/SA85 application uses. A PC/SA85 combination running Modicon Modsoft software uses two paths, each of which requires 3 ms.

In the example in Figure 16, connecting the PC/SA85 to the I/O network will add 6 ms to the time required for servicing the I/O drops.

## 2.5.3 Connecting a Node Temporarily to the I/O Network

If your network has a spare tap, and you use it to connect a Programmer temporarily, you should be aware of the impact this causes on the network. When you connect the Programmer, and again when you disconnect it, the token rotation sequence will be disrupted. This will cause an interruption in the servicing of your I/O nodes.

Section 2.2.6 describes the effect of connecting or disconnecting devices on an active network. Section 2.7 gives estimates of the time required for the token sequence to be re-established.

## 2.5.4 Calculating Token Rotation Time

Token rotation time is determined by the node count, average size of messages, and quantity of paths used. The formula for calculating token time for a network with a CPU/NOM, Programmer, and I/O nodes (as in Figure 16) is:

$$\begin{aligned} \text{Token Rotation Time (ms)} = & \\ & (\text{Node Count}) \\ & + ((\text{Node Count} - 1) * 2) * (\text{Average Message Size}) * 0.016 \\ & + (3 * \text{Quantity of Programmer Paths}) \end{aligned}$$

## 2.5.5 Example: Token Rotation Time

Figure 17 shows token rotation time estimates for the example connection in Figure 16 (an I/O network with one Programmer connected and using a total of two paths).

The CPU/NOM limit for a given direction of I/O data (500 words in, 500 words out) will be reached at the combinations of node count and average message size shown.

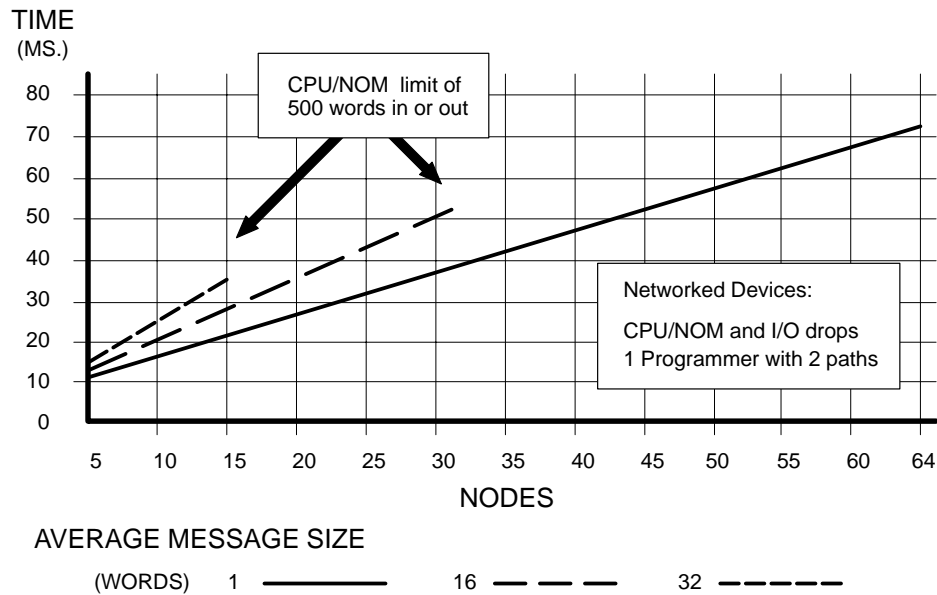


Figure 17 Token Rotation Time: Network With Programmer



**Recommendation:** For minimal interruption of I/O servicing on multi-function networks, follow these recommendations:

- ★ **Assigning Node Addresses** Assign the lowest address (2) to the CPU. Address 1 is reserved (see Section 2.7). Assign the next addresses to the I/O nodes, without any gaps in the sequence.
- ★ **Adding or Deleting Nodes** Your application should be able to support the downtime required to reconstitute the token rotation, in the event of node devices dropping out or rejoining the network. Note that typical token reconstitution times may range up to 250 ms. Section 2.7 shows how to estimate the time required to reconstitute the token rotation sequence.
- ★ **Provide Service Taps and Drops** Provide at least one spare tap and drop cable on each network. This will allow you to connect a Programmer node for testing the network, without having to disconnect another node. Note that connecting and disconnecting the Programmer will still cause the network token sequence to be initialized and reconstituted, as described in Section 2.7.
- ★ **Adding Non-I/O Devices** If you require a networked User Interface or other non-I/O device, connect it to a separate network at a NOM port. Reserve the CPU network for I/O servicing only.
- ★ **Separating Networks by Function** Distribute your application into separate networks to minimize the impact of a node dropping out or rejoining a network.

## 2.6 Multifunction Networks With Mixed Nodes

Figure 18 is an example of an application that requires a combination of a networked Programmer, User Interface (UI), and ModConnect device in addition to the CPU and I/O drop nodes.

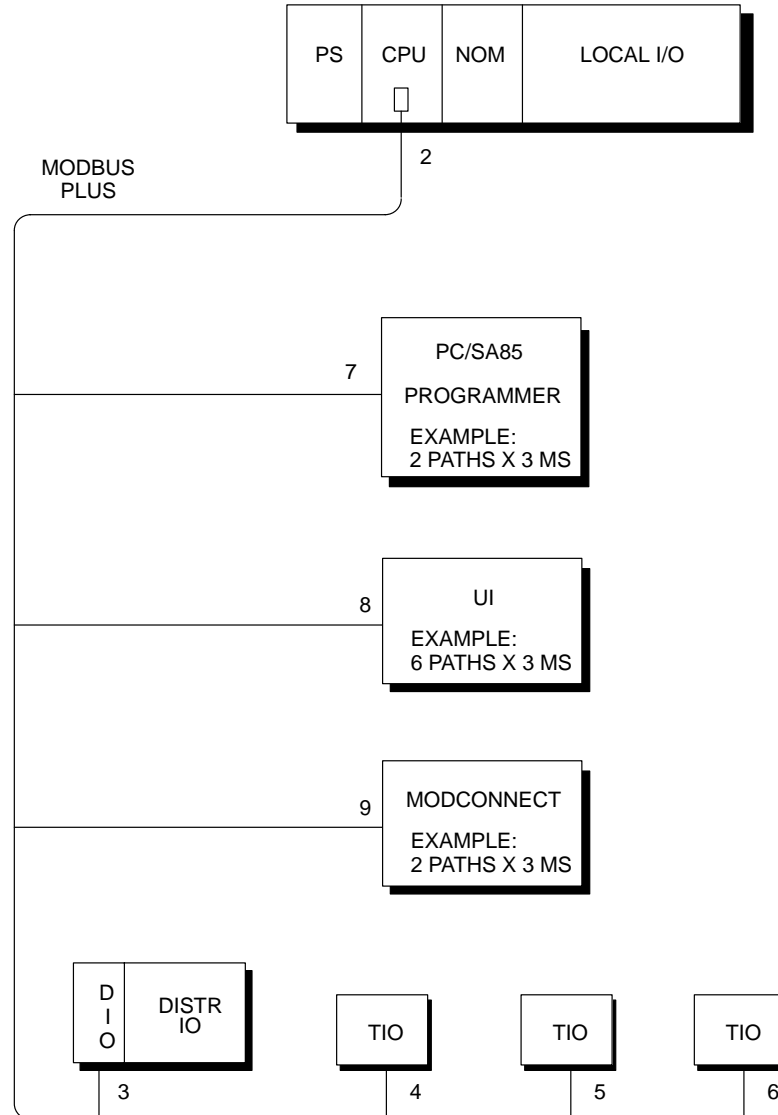


Figure 18 Example: Multifunction Network With Mixed Nodes

### 2.6.1 Impact on I/O Servicing

When the CPU executes commands from the Programmer, typically up to 10 ms can be added to the CPU scan time.

In addition, you may want to transact messages with some devices through MSTR blocks, and may need to move variable amounts of data. This can affect the deterministic timing of I/O processes. Data handling requirements are determined by each device's application. Consult the device manufacturer for specifications, including path quantities.

### 2.6.2 Impact on Token Rotation Time

The impact on token rotation time caused by inclusion of mixed devices will depend upon the total quantity of paths the devices use.

A Programmer consisting of a PC/SA85 combination running Modicon Modsoft software uses two paths, each of which requires 3 ms.

Typical UIs use up to eight paths, each of which requires 3 ms.

For each ModConnect device, refer to the manufacturer's specification to determine the quantity of paths used.

### 2.6.3 Calculating Token Rotation Time

Token rotation time is determined by the node count, average size of messages, and quantity of paths used. The formula for calculating token time for a network with a CPU/NOM, Programmer, UI, and I/O nodes (as in Figure 18) is:

$$\begin{aligned} \text{Token Rotation Time (ms)} = & \\ & (\text{Node Count}) \\ & + ((\text{Node Count} - 1) * 2) * (\text{Average Message Size}) * 0.016 \\ & + (3 * \text{Quantity of Programmer Paths}) \\ & + (3 * \text{Quantity of UI Paths}) \\ & + (3 * \text{Quantity of ModConnect Paths}) \end{aligned}$$

## 2.6.4 Example: Token Rotation Time

Figure 19 shows token rotation time estimates for the example connection in Figure 18 (an I/O network with one Programmer connected and using a total of two paths, and with one UI using a total of eight paths).

The CPU/NOM limit for a given direction of I/O data (500 words in, 500 words out) will be reached at the combinations of node count and average message size shown.

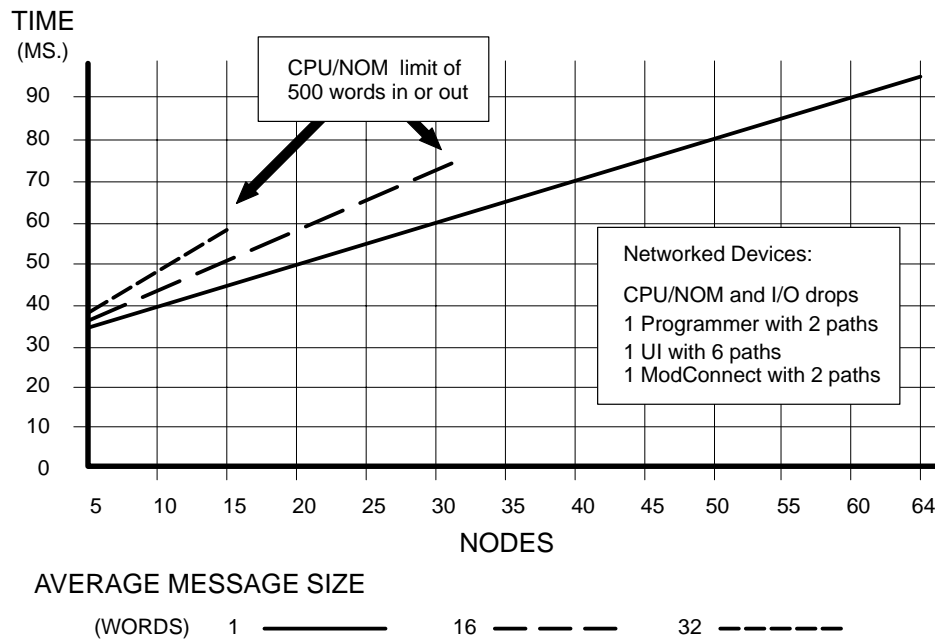


Figure 19 Token Rotation Time: Network With Mixed Nodes



**Recommendation:** For minimal interruption of I/O servicing on multi-function networks, follow these recommendations:

- ★ **Assigning Node Addresses** Assign the lowest address (2) to the CPU. Address 1 is reserved (see Section 2.7). Assign the next addresses to the I/O nodes, without any gaps in the sequence.
- ★ **Adding or Deleting Nodes** Your application should be able to support the downtime required to reconstitute the token rotation, in the event of node devices dropping out or rejoining the network. Note that typical token reconstitution times may range up to 250 ms. Section 2.7 shows how to estimate the time required to reconstitute the token rotation sequence.
- ★ **Provide Service Taps and Drops** Provide at least one spare tap and drop cable on each network. This will allow you to connect a Programmer node for testing the network, without having to disconnect another node. Note that connecting and disconnecting the Programmer will still cause the network token sequence to be initialized and reconstituted, as described in Section 2.7.
- ★ **Adding Non-I/O Devices** If you require a networked User Interface or other non-I/O device, connect it to a separate network at a NOM port. Reserve the CPU network for I/O servicing only.
- ★ **Separating Networks by Function** Distribute your application into separate networks to minimize the impact of a node dropping out or rejoining a network.

## 2.7 Estimating Token Reconstitution Time

Token Reconstitution Time is the time required for the network to establish a new token rotation sequence following the dropout of a node or group of nodes from the network, or the joining of a node or group to the network. It is possible that two or more nodes may drop out and rejoin simultaneously (for example, due to an area power failure).

### Calculating Token Reconstitution Time

Token reconstitution time is affected by two factors: the quantity of nodes remaining on the network, and their node addresses. The formula for calculating the time is:

$$\begin{aligned}\text{Token Reconstitution Time (ms)} = & \\ & 50 + (4 * \text{Lowest Remaining Address}) \\ & + (16 * (\text{Total Dropped Nodes} - 1)) \\ & + (\text{Total Remaining Nodes} - 1)\end{aligned}$$

Figure 20 shows token reconstitution time estimates for a network with one dropped node. Figure 21 shows estimates for two dropped nodes. Both figures illustrate the improvement in token reconstitution time that is achieved by assigning low node addresses.

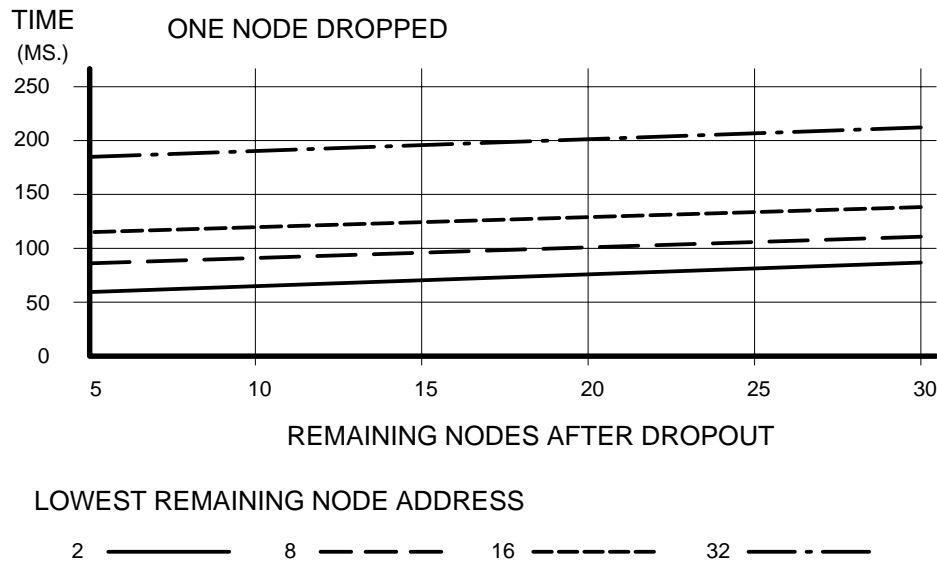
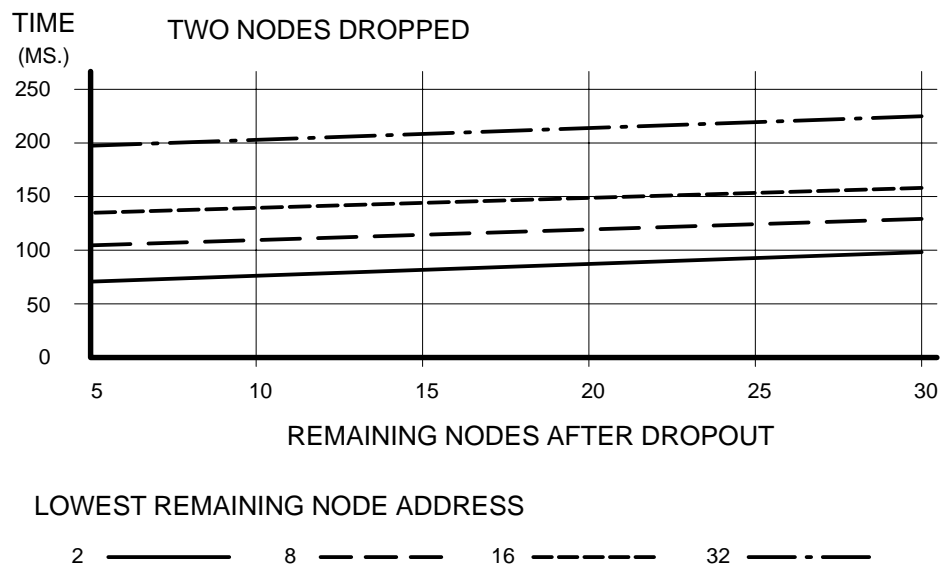


Figure 20 Estimated Token Reconstitution Time (One Node Dropped)





**Figure 21 Estimated Token Reconstitution Time (Two Nodes Dropped)**



**Recommendation:** For the best performance in handling the event of nodes dropping out and rejoining, follow these recommendations:

- ★ Assign node addresses starting at address 2 (avoid using address 1).

Some programmable controllers may revert to address 1 in the event of a loss of their stored configuration (e.g., through a power failure, combined with a defective backup battery). In addition, Modicon programmer devices used in network maintenance have a factory default address of 1. By reserving address 1 you eliminate the possibility of two nodes having the same address, and allow the network to be more easily diagnosed.

- ★ Assign the lowest address to the CPU or NOM. If multiple controllers or NOMs are present on a single network, assign the lowest addresses to them.
- ★ Assign sequential addresses, e.g. 2 - 3 - 4 etc., without any gaps in the address sequence.
- ★ After you assign the lowest address to the CPU or NOM, assign the next addresses in sequence to the I/O drops (the DIO Drop Adapters and TIO modules).
- ★ Finally, assign any remaining addresses to other node devices on the network, such as programmers, User Interfaces, or host adapters.

## 2.8 Verifying the I/O Configuration

---

There are your considerations for verifying the proper configuration of a network with a single master node. Considerations for networks with multiple master nodes are presented in Section 4.4.

### 2.8.1 Verifying Node Addresses Before Startup

Before applying power to the controller, NOMs, and I/O node devices, you should verify the address you have assigned to each node. Ensure that the addresses are in a continuous sequence without any gaps, and that the controller or NOM has the lowest address on its network (address 2). Ensure that I/O drops have the next addresses in sequence above the controller or NOM. Finally, ensure that any remaining devices such as programmers, User Interfaces, or host adapters have the next addresses in sequence above the I/O drops.

### 2.8.2 Verifying Controller Tables Before Startup

Before starting your network's I/O nodes, you can verify the I/O mapping configurations you have programmed into your controller. Verify the contents of your DIO Maps and Peer Cop tables. Ensure that each DIO Drop Adapter and TIO module has the correct entry for its data source and destination logic references. Ensure that only one master node has write privilege to any given drop.

### 2.8.3 Verifying DIO Drop Adapter Configurations While the Network is Running

You can use the MSTR function block in your application programs to interrogate each DIO Drop Adapter's internally stored DIO Map. Each drop maintains two words in its map that identify the privileged node for writing the drop's DIO Map and data. These words are accessible as 4X register references to your MSTR. Information for using the MSTR is provided in Appendix A .

### 2.8.4 Verifying TIO Module Configurations While the Network is Running

You can use the MSTR function block to interrogate each TIO module's internally stored status table. Each module maintains nine words of status information. This identifies the last node that performed a write to the module, as well as module's type, revision number, and remaining holdup time. These words are accessible as 4X references to your MSTR. The layout of the module status table is provided in Appendix B .

# Chapter 3

## Estimating I/O Network Performance

---

- Estimating Data Throughput Time

## 3.1 Estimating Data Throughput Time

---

Data Throughput Time is the time required for sensing the state of a field device and using that information to control another field device. The time is affected by the following factors:

- ☐ The scan time of the controller, determined by the controller's speed and the size and content of its application program.
- ☐ The choice of network master device (CPU or NOM).
- ☐ The quantity of networked I/O drops being serviced.
- ☐ The quantity of I/O modules being serviced at each I/O drop.

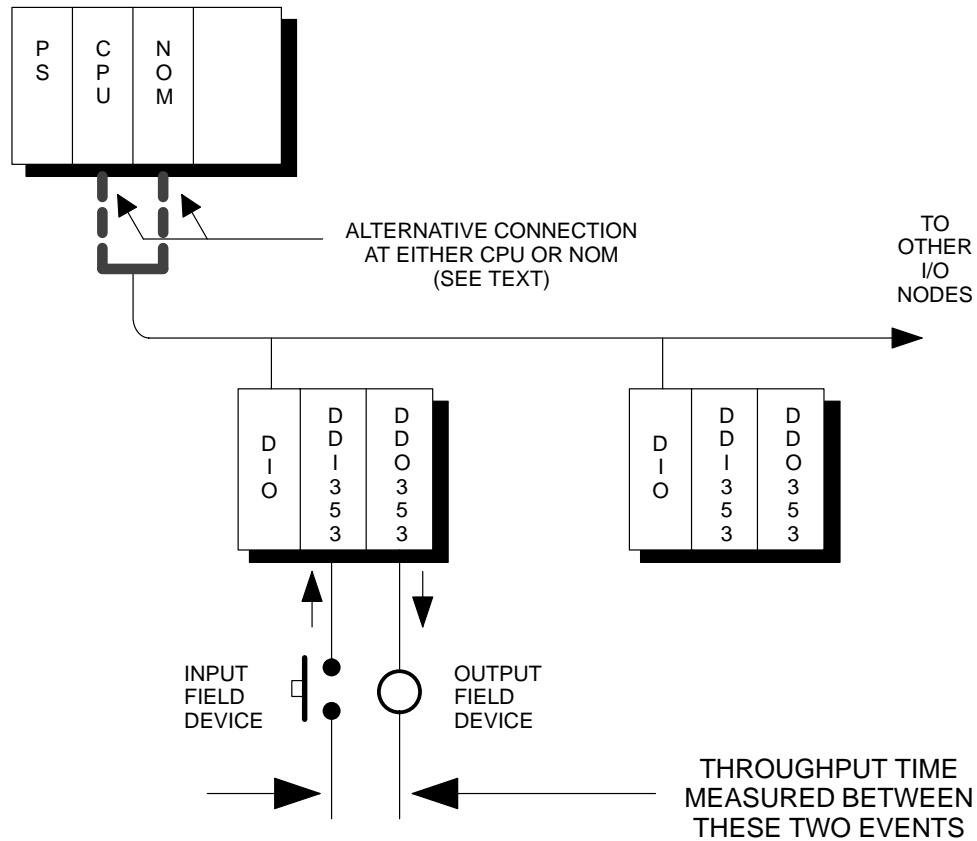
### 3.1.1 Examples in This Chapter

This chapter provides throughput time estimates based on the following configurations:

- ☐ CPU types: Quantum models CPU 113/213, CPU 424.
- ☐ Ladder Logic program sizes: 2K, 4K, 6K, 8K, 10K words.
- ☐ Program contents: 80% Relay, 11% Data Moves, 9% Timers and Counters.
- ☐ Network master device: CPU, NOM.
- ☐ Quantities of I/O drops: 2 to 6.
- ☐ Quantities of I/O modules per drop: 2 or 16.

### 3.1.2 Network Model for Measuring Throughput Time

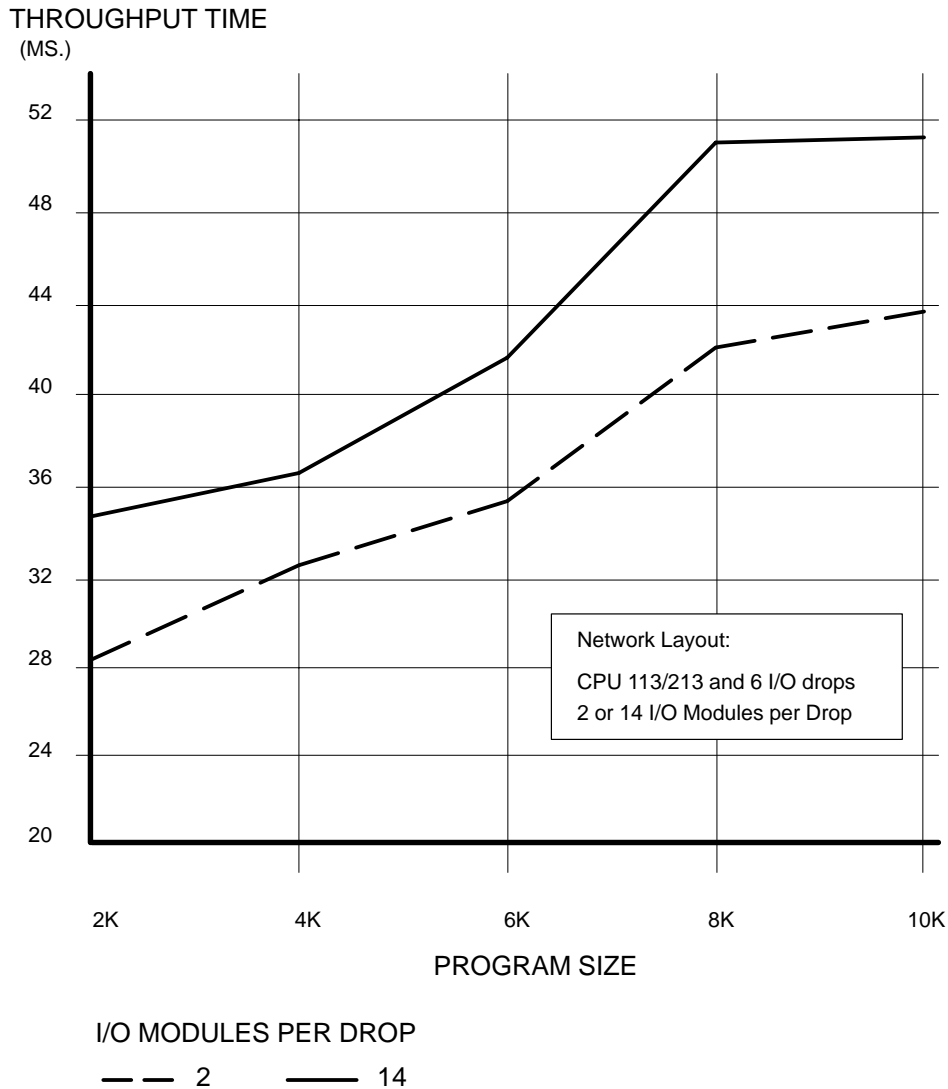
Figure 22 presents a network model for measuring throughput time. Time is measured between two events occurring at field devices. An input state is sensed and processed by the application program, resulting in the control of an output state.



**Figure 22 Network Model for Measuring Throughput Time**

Either the CPU Modbus Plus port or the NOM can be used to control the Modbus Plus I/O network. This is shown in the alternative connection in Figure 22. The resulting throughput time will be different according to the type of connection used in the application. Timing estimates are provided on the following pages for either connection.

### 3.1.3 Example: Time Versus Program Size

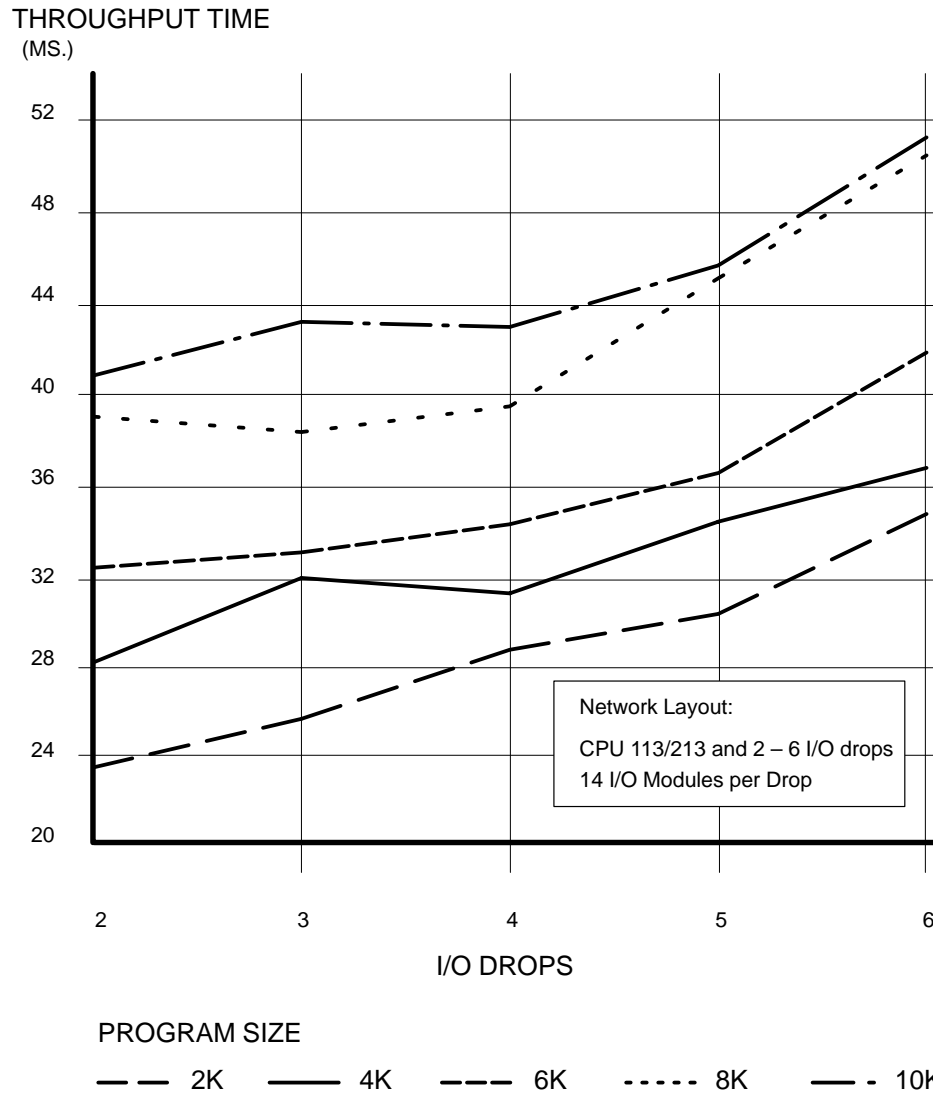


**Figure 23 Estimated Throughput Time: Program Size and Modules/Drop**

Figure 23 shows estimated throughput times for a Quantum 113/213 CPU with program sizes from 2K to 10K. The program contents are: 80% Relay, 11% Data Moves, 9% Timers and Counters.

The network in this example has 6 I/O drops, with either 2 or 14 I/O modules per drop.

### 3.1.4 Example: Time Versus Program Size and I/O Drops



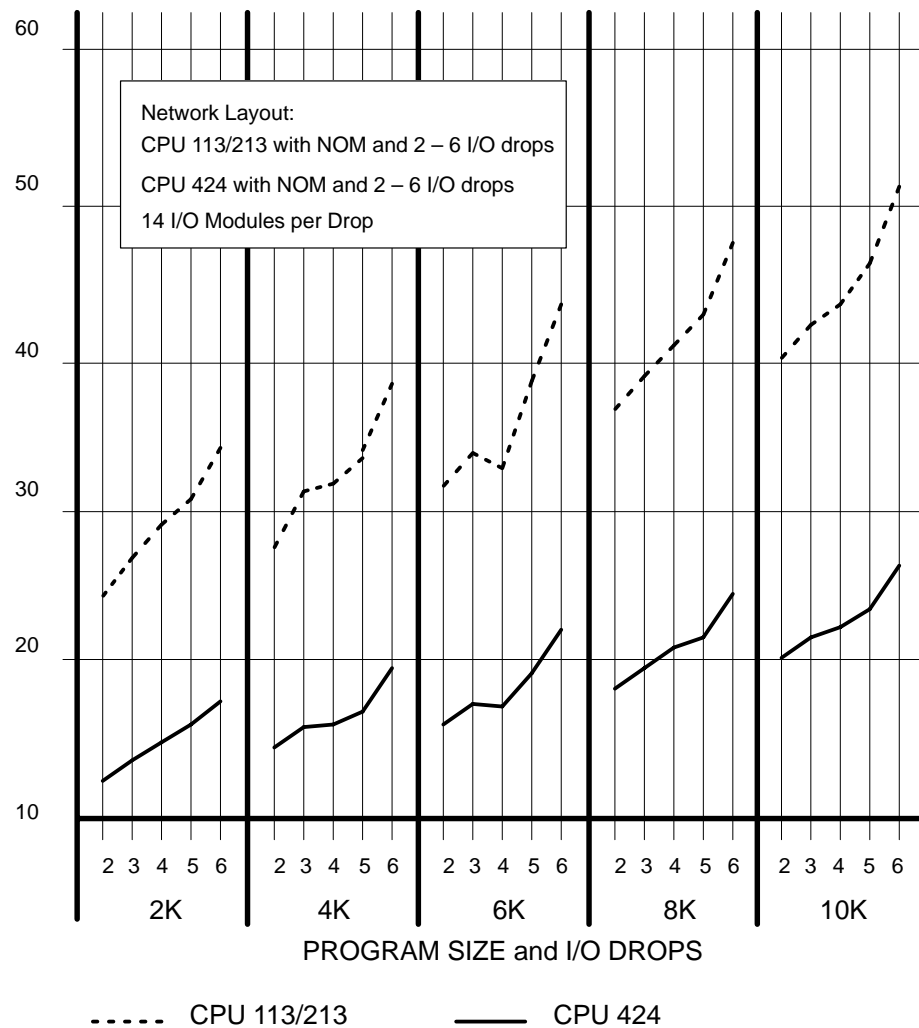
**Figure 24 Estimated Throughput Time: Program Size and Modules/Drop**

Figure 24 shows estimated throughput times for a Quantum 113/213 CPU with program sizes from 2K to 10K. The program contents are: 80% Relay, 11% Data Moves, 9% Timers and Counters.

The networks in this example have 2 to 6 I/O drops. Each drop contains 14 I/O modules.

### 3.1.5 Comparison Between Quantum 113/213 and 424 CPU

THROUGHPUT TIME  
(MS.)



**Figure 25 Estimated Throughput Time: Program Size and Modules/Drop**

Figure 25 shows a comparison of estimated throughput times for a Quantum 113/213 CPU and Quantum 424 CPU, with program sizes from 2K to 10K. The program contents are: 80% Relay, 11% Data Moves, 9% Timers and Counters.

The networks in this example are connected to a NOM and have 2 to 6 I/O drops. Each drop contains 14 I/O modules.



# Chapter 4

## Using Multiple Controllers

---

- ☐ Using More Than One Controller
- ☐ Example: Drops With Multiple Controllers
- ☐ Configuration Requirements for Multiple Controllers
- ☐ Verifying the I/O Configuration

## 4.1 Using More Than One Controller

You can connect two or more controllers to a common I/O network. Controllers can communicate with each other using Modbus Plus Peer Cop or MSTR messages. Each controller transacts messages with its own mapped DIO Drop Adapters and TIO modules. Multiple controllers must not be configured to write to the same I/O drop.



**Warning** Having two or more nodes with write privilege to the same DIO Drop Adapter or TIO module is a violation of I/O network configuration requirements. It can cause erratic communication on the network, and can result in possible injury in applications that affect the safety of personnel.

### 4.1.1 Multiple Controllers With DIO Drop Adapters

Figure 26 illustrates a network in which two controllers are I/O mapped to a group of DIO Drop Adapters. Each controller is configured to write to its own group of DIO Drop Adapters. Multiple controllers must not be configured to write to the same drop, however any number of controllers may read from any drop.

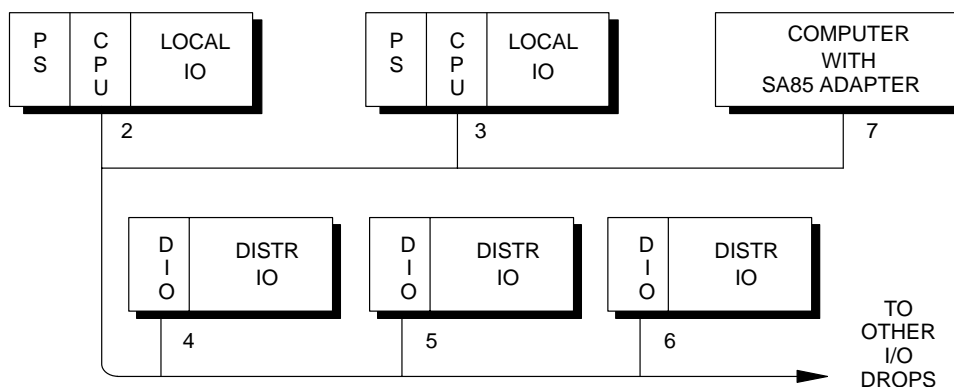


Figure 26 Multiple Controllers With I/O Drops



**Recommendation:** You must closely coordinate the separate DIO Map tables in the controllers. Ensure that each DIO Drop Adapter has only one controller with write privilege to that adapter. You must also design the SA85 host's application to ensure that it does not write to any DIO Drop Adapter which is being written to by a controller node. Section 1.8.1 describes the write privilege to DIO Drop Adapters.

Figure 26 also illustrates how node addresses must be assigned:

- ☐ Addresses are in a continuous sequence without gaps.
- ☐ The two controllers are assigned the lowest addresses. The I/O drops are assigned the next following addresses in the sequence. The SA85 adapter is assigned the remaining address.

#### 4.1.2 Multiple Controllers With TIO Modules

Only one controller can have the privilege for writing data to a given TIO module. Multiple controllers can read data or status from any drop.

In Figure 27, three controllers operate as controlling devices for three separate parts of the I/O application. Each controller communicates with a separate set of TIO modules.

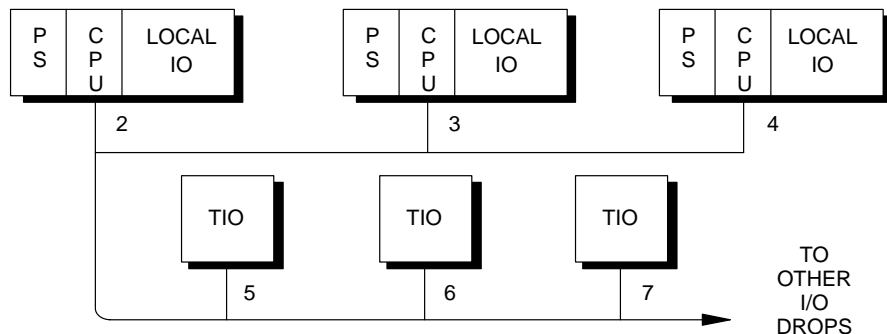


Figure 27 Multiple Controllers With TIO Modules



**Recommendation:** Note that each TIO module maintains an internal 60-second timer that controls write privilege to the module. The first controller that writes to the TIO module acquires the sole write privilege to the module, and retains it so long as the timer is not allowed to expire. You must closely coordinate the Peer Cop tables in the three controllers to ensure that each TIO module has only one controller with write privilege to that module. Section 1.8.2 describes the write privilege to TIO modules.

Figure 27 also illustrates how node addresses must be assigned:

- ☐ Addresses are in a continuous sequence without gaps.
- ☐ The two controllers are assigned the lowest addresses. The I/O drops are assigned the next following addresses in the sequence.

## 4.2 Example: Drops With Multiple Controllers

Figure 28 shows an example of part of an I/O network system during its startup. Two controllers are present at node addresses 2 and 3. Six I/O drops are used in the application. Three DIO Drop Adapters are present at addresses 4, 5, and 6 (each adapter controls an assortment of input and output modules). Three TIO modules are at addresses 7, 8, and 9.

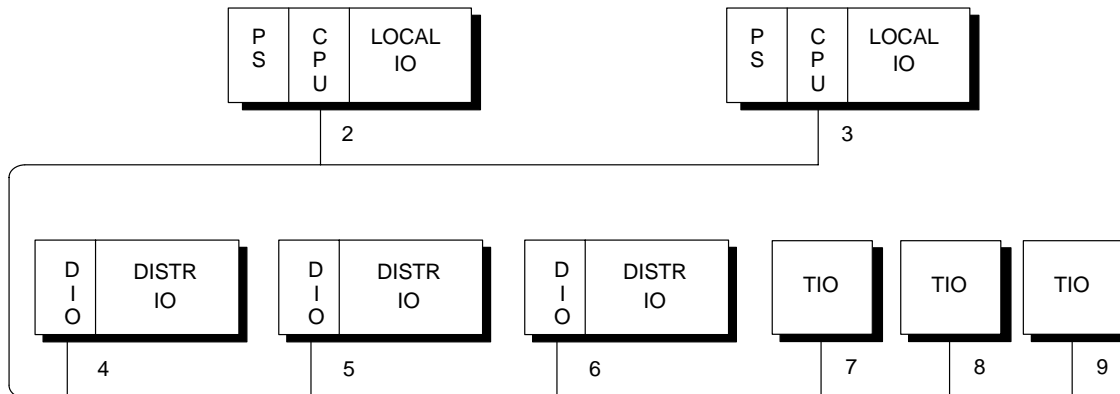


Figure 28 Startup Example With Multiple Controllers

### Mapping the DIO Drop Adapters

The two controllers' DIO Maps specify the following directions for communications with the three DIO Drop Adapters:

DIO Drop Address	Config Write to Drop	Data Write to Drop	Data Read from Drop
4	by Controller 2	by Controller 2	by Controller 2
5	by Controller 2	by Controller 2	by Controllers 2 and 3
6	by Controller 3	by Controller 3	by Controller 3

### Mapping the TIO Modules

The two controllers' Peer Cop tables specify the following directions for communications with the three TIO modules:

TIO Module Address	Module Type	Data Write to Module	Data Read from Module
7	Output	by Controller 2	Not applicable
8	Bidirectional Input/Output	by Controller 2	by Controllers 2 and 3
9	Input	Not applicable	by Controller 3

## 4.2.1 Processes During Startup

When the network initializes at startup, the token rotation sequence is established with the lowest-addressed node acquiring the first token. The token rotation sequence thus begins with the controller at node 2.

The following events occur:

### **Controller 2 Has Its First Token**

Controller 2 downloads drop DIO Maps to DIO Drop Adapters 4 and 5. The controller writes output data to TIO modules 7 and 8.

### **Controller 3 Has Its First Token**

At its token time, controller 3 downloads the DIO Map to DIO Drop Adapter 6.

### **DIO Drop Adapters Have Their First Tokens**

After each DIO Drop Adapter has received its DIO Map, it is ready to process incoming and outgoing data messages. All three drops will send their output data as they receive their tokens.

### **TIO Modules Have Their First Tokens**

TIO module 7 is an output module and thus has no data to send. It passes the token to the module at address 8. When TIO module 8 acquires the token, it sends its input data. Both controllers 2 and 3 receive the data. When TIO module 9 has the token it sends its input data, which is received by controller 3.

### **Controllers 2 and 3 Have Their Succeeding Tokens**

As controllers 2 and 3 acquire their succeeding tokens, each controller with data write privilege to a drop writes a data message to that drop. DIO Drop Adapters 4 and 5, and TIO Modules 7 and 8, must each receive a write message from controller 2. DIO drop 6 must receive a write message from controller 3. In addition to updating the states of the outputs at these addresses, succeeding writes prevent the output holdup timers from expiring.

Note that the proper coordination of data in the application between controllers 2 and 3 is determined by the configurations of their DIO Maps and Peer Cop tables. For example, controllers 2 and 3 are both mapped to read data from DIO Drop Adapter 5. Neither controller can successfully read from drop 5 until that drop's DIO Map has been written from controller 2.



**Recommendation:** When you configure multiple controllers on a common I/O network, you must treat the controllers as parts of a single, integrated configuration. DIO Maps and Peer Cop tables in the controllers must be coordinated to eliminate conflicting data transfers.

## 4.3 Configuration Requirements for Multiple Controllers

---

Each controller has its own DIO Map and Peer Cop table that define its read/write privileges for the drops on the I/O network. When multiple controllers are being configured on a common network, their DIO Maps and Peer Cop tables must be closely coordinated to ensure that the overall application has no conflicting write privilege assignments.

Multiple controllers must not be configured to perform write operations to the same DIO Drop Adapter or TIO module. However, any number of controllers can be configured for read-only operations from the same DIO Drop Adapter or TIO module.

### Requirements for DIO Drop Adapters

Only one controller may have privilege for downloading the drop DIO Map to a given DIO Drop Adapter. That controller must also be the only one that writes data to the drop. Therefore any given DIO Drop Adapter must have only one controller with the write privilege to that drop.

### Requirements for TIO Modules

Only one controller may have privilege for writing to a given TIO module. Although each TIO module maintains an internal 60-second write privilege timer for reserving write privilege, you should ensure that only one controller is configured to perform writes to the module.



**Warning** Having two or more nodes with write privilege to the same DIO Drop Adapter or TIO module is a violation of I/O network configuration requirements. It can cause erratic communication on the network, and can result in possible injury in applications that affect the safety of personnel.

## 4.4 Verifying the I/O Configuration

---

### 4.4.1 Verifying Node Addresses Before Startup

Before applying power to the controllers, NOMs, and I/O node devices, you should verify the address you have assigned to each node. Ensure that the addresses are in a continuous sequence without any gaps, and that the controllers or NOMs have the lowest addresses on the network. Ensure that I/O drops have the next addresses in sequence above the controllers or NOMs. Finally, ensure that any remaining devices such as programmers, User Interfaces (UIs), or network adapters have the next addresses in sequence above the I/O drops.

### 4.4.2 Verifying Controller Tables Before Startup

Before starting your network's I/O nodes, you must verify the I/O mapping configurations you have programmed into each controller. Check their DIO Maps and Peer Cop tables and verify that only one controller has been mapped for write privilege to any given I/O drop node on your network. If you find the same drop node defined for write privilege in more than one DIO Map or Peer Cop table, you must reconfigure your controllers to eliminate this configuration error.

### 4.4.3 Verifying DIO Drop Adapter Configurations While the Network is Running

You can use the MSTR function block in your application programs to interrogate each DIO Drop Adapter's internally stored DIO Map. Each drop maintains two words in its map that identify the privileged node for writing the drop's DIO Map and data. These words are accessible as 4X register references to your MSTR. Information for using the MSTR to access DIO Drop Adapters is provided in Appendix A .

### 4.4.4 Verifying TIO Module Configurations While the Network is Running

You can use the MSTR function block to interrogate each TIO module's internally stored status table. Each module maintains nine words of status information. This identifies the last node that performed a write to the module, as well as module's type, revision number, and remaining holdup time. These words are accessible as 4X references to your MSTR. Information for using the MSTR to access TIO modules is provided in Appendix B .





# Appendix A

## Checking DIO Drop Adapter Status

---

- ☐ DIO Drop Adapter Status Information
- ☐ Checking Drop Health With DIOH
- ☐ Checking Drop RAM Contents With MSTR
- ☐ Monitoring I/O Module Status

## A .1 DIO Drop Adapter Status Information

---

### A .1.1 Health Reporting by the Drop Adapter

Each data message read from a DIO Drop node contains two words that describe the current health of the drop. The first word received is a 16-bit checksum of the drop's DIO Map contents. The second word contains 16 bits describing the current health of each I/O module present in the drop's backplane.

The received checksum is compared with the original checksum value stored in the controller DIO Map. If the two are not identical, the entire drop is marked as unhealthy. The controller or Network Option Module having the write privilege for the drop's DIO Map will attempt to rewrite the DIO Map, restart the drop, and return it to the healthy state.

In the second received word, each bit from left to right corresponds to a slot position in the backplane, with the most significant bit representing the leftmost slot. A logic '1' indicates that the module in that slot is healthy. A logic '0' (zero) indicates that some error exists at the module.

Both words can be read by the MSTR function block. The second word alone can be read by the DIO Health (DIOH) function block.

## **A .1.2 How the Drop Adapter RAM Contents are Stored**

DIO Maps are downloaded to DIO Drop Adapters from the controller or Network Option Module node. The DIO Map information is stored in a RAM area in the DIO Drop. DIO data for incoming and outgoing messages are also stored in the RAM area.

The DIO Drop's RAM area can be accessed directly, using the MSTR function block or Modbus 'Read 4X' commands and the appropriate 4X data references. The area can also be accessed using standard Peer Cop Specific In techniques. Note that the data length limit for each type of communication may require multiple commands to read the RAM table.

For example, a master node can read a drop's RAM table to determine which node currently has write privilege for writing the drop DIO Map or data. Some of the RAM table's references have Read access only. Others have both Read and Write access. Note that, although some references have Read/Write access, the proper RAM contents must be configured into the controller's DIO Map and downloaded to the drop, and not through MSTR writes. MSTR access is for monitoring purposes only.

## A .2 Checking Drop Health With DIOH

---

### Using the DIOH

Here is an overview of the DIOH function block, to assist your I/O network planning. Details about using ladder logic functions in your application program are provided in the *Modicon Ladder Logic Block Library User Guide*.

Figure 29 shows an overview of the DIOH block structure.

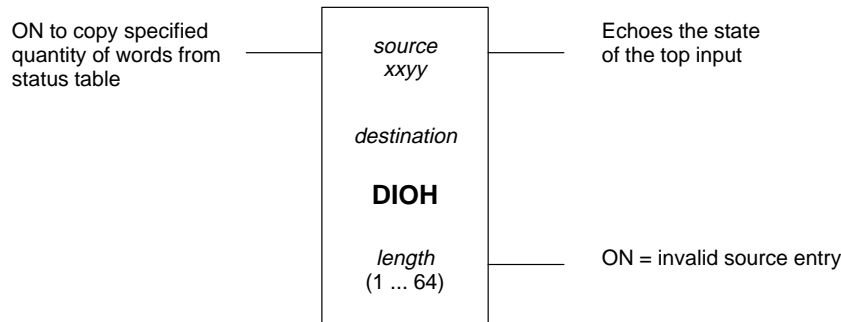


Figure 29 DIOH Block Overview

### The DIOH Source Field

The *source* field is in the format: *xxyy*, where:

*xx* Decimal value in the range 00 ... 16. This value specifies the master node device's location. If the DIOH block is getting the health status of a DIO Drop Adapter on the network connected to a Network Option Module, this value identifies the backplane slot in which that NOM module resides. If the status request is for a drop on the controller's own I/O network, you can specify either the controller's physical slot location or a value of 00.

*yy* Decimal value in the range 01 ... 64. This value identifies the Modbus Plus node address of the DIO Drop Adapter for which the health status is being requested.

For example, a *source* value of 0305 decimal requests the health status of DIO Drop Adapter node 5 on the I/O network connected to the NOM in slot 3.

**The DIOH Destination Field**

The *destination* field specifies the starting register (4x) at which the drop status will be stored.

**The DIOH Length Field**

The *length* field specifies the quantity of registers that will be written beginning at the starting reference. If a length is specified that would cause the DIOH to return values beyond the health table, the block will return a length only up to the end of the table.

## A .3 Checking Drop RAM Contents With MSTR

### Using the MSTR

Here is an overview of the MSTR function block, to assist your I/O network planning. Details about using ladder logic functions in your application program are provided in the *Modicon Ladder Logic Block Library User Guide*.

Figure 30 shows an overview of the MSTR block structure.

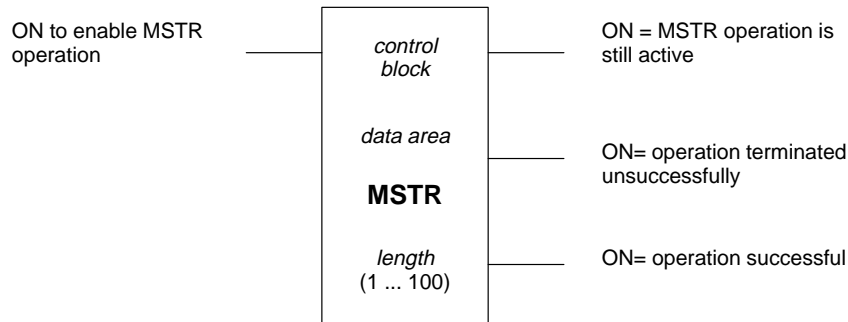


Figure 30 MSTR Block Overview

### The MSTR Control Block Field

The *control block* field specifies the starting register (4x) of a group of nine contiguous registers that define the operation of the MSTR block.

Register	Purpose
4XXXX (starting)	Operation: 2 = Read (Write not allowed)
4XXXX + 1	Returned error status
4XXXX + 2	Data length: quantity of registers
4XXXX + 3	Starting register: 1 ... 65 for drop Read
4XXXX + 4	High byte: 0 = CPU, other = NOM slot: low byte = Drop node address
4XXXX + 5	0 for I/O network
4XXXX + 6	0 for I/O network
4XXXX + 7	0 for I/O network
4XXXX + 8	0 for I/O network

### The MSTR Data Area Field

The *data area* field specifies the starting register (4x) of a group of contiguous registers. This area is the destination for incoming data for a read operation.

#### The MSTR Length Field

The *length* field specifies the maximum quantity of registers allocated in the data area, in the range 1 ... 100.

### A .3.1 The Drop Adapter RAM Table

The table below shows the layout of the DIO Drop Adapter's RAM area.

Word	Access	Description	Reference
0	R only	Drop checksum	40001
1	R only	Module health: MSB = slot 1; LSB = slot 16 1 = healthy; 0 = unhealthy	40002
2 ... 31	R only	Outgoing Data Area (data from DIO Drop's input modules)	40003 ... 32
32 ... 63	R/W	Incoming Data Area (data to DIO Drop's output modules)	40033 ... 64
64	R/W	Node Address, DIO Map Write privilege	40065
65	R/W	Node Address, Data Write privilege	40066

## A .4 Monitoring I/O Module Status

---

Each I/O module present in a DIO Drop Adapter's backplane provides one byte (8 bits) of status to the Drop Adapter. The status information is specific to each type of module - for example, blown fuse, broken wire or overvoltage detection can be reported in the status byte. The drop reads and saves the status information from its modules.

The controller requests the module status from each drop that has been configured in the DIO Map to provide that status. The request is sent, one drop at a time, on a rotating basis. At the end of each controller scan, the controller will initiate a request for the status of the next drop (the request is sent at the controller or NOM's next token time). When the drop responds at its token time, the module status is retrieved by the controller into a series of input (3X) registers that were specified as part of the controller's DIO Map configuration. The status of the module in the first slot position is retrieved into the low byte of the starting register.

For example, if the module status has been mapped to start at register 30010, the status for a 16-slot DIO Drop would be received as shown in Figure 31.

30010	Slot 2	Slot 1
30011	Slot 4	Slot 3
	...	...
30017	Slot 16	Slot 15

**Figure 31** Module Status Example

Status information for each type of module is described in the *Quantum Automation Series Hardware Reference Guide*. Instructions for setting up the status table using Modicon's Modsoft panel software are contained in the *Modsoft Programmer User Guide*.



# Appendix B

## Checking TIO Module Status

---

- ☐ TIO Module Registers
- ☐ Checking TIO Registers With MSTR
- ☐ Defining Write Privilege to TIO Modules

## B .1 TIO Module Registers

Each Modbus Plus TIO module contains three groups of internal registers. These can be accessed through the network to transfer input or output data, to set or retrieve the module's configuration, or to monitor its status.

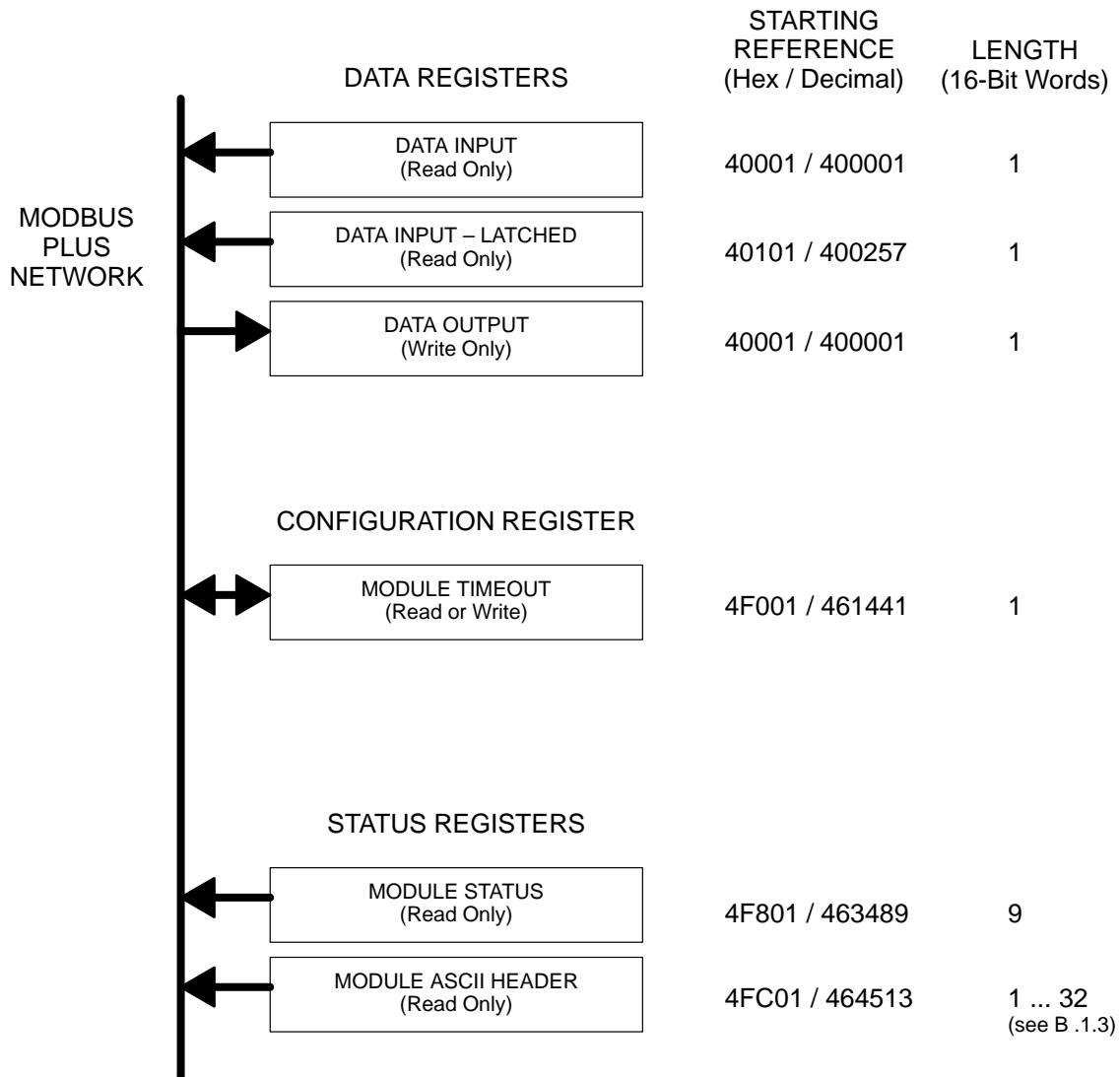


Figure 32 Modbus Plus TIO Module Registers Layout

## B .1.1 Data Registers

These registers contain the input or output data for each module.

### 40001 hex / 400001 dec — Data Input or Output

Reference 40001 is used to address input data from field inputs, and output data to field outputs.

This reference is the only one that is accessible through Peer Cop data transfers. All other registers can be accessed using MSTR blocks.

### 40101 hex / 400257 dec — Data Input, Latched

Reference 40101 latches inputs ON for a minimum of 500 milliseconds, if an ON input occurs with a time duration less than this. It ensures that shorter input pulses are captured and made available to the TIO application. If an input remains ON for more than this duration, the reference does not continue to latch it, but follows the input's state.

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Input/Output Terminal Pin:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Figure 33 Input/Output Registers Bit Layout

### Quantity of Data Bytes Transferred

Each TIO module transfers the following quantities of data bytes for a read or write operation.

Part Number	Module Type	Read (Bytes)	Write (Bytes)
170 BDI 342 00	Discrete 16 in 24 V dc	2	1
170 BDO 342 00	Discrete 16 out 24 V dc	1	2
170 BDM 342 00	Discrete 16 in/16 out 24 V dc	3	2
170 BDI 542 50	Discrete 16 in 115 V ac	2	1
170 BDO 542 50	Discrete 16 out 24 ... 230 V ac	1	2

## B .1.2 Configuration Register

This register controls the holdup timeout value for output modules.

### **4F001 hex / 461441 dec — Outputs Holdup Time**

Reference 4F001 specifies the amount of time that outputs will be held in their current state, if they are not updated by a new Modbus Plus Write command.

The timeout value is expressed in units of 10 milliseconds, with a minimum register value of 30 (300 milliseconds) and maximum value of 6000 (60 seconds). The default value is 100 (1 second).

If the module's holdup value expires before a new Write command is received, all outputs are set to logical 0 (zero).

The current contents of the register can be read using a Modbus Plus Read command.

## B .1.3 Status Registers

### 4F801 ... 4F809 hex / 463489 ... 463498 dec — Status Block

These registers provide information about the module's revision level and current operating parameters. Nine words are used to contain the module's status.

Ref (hex)	Meaning	Contents
4F801	Length of status area	9
4F802	Quantity of module input bytes	2 for input module, 1 for output module
4F803	Quantity of module output bytes	1 for input module, 2 for output module
4F804	Reserved	
4F805	Module revision number	Format: XR where: X = upper 4 bits , reserved for station management commands (always 0). R = revision number. Example: 200 hex shows revision 2.00.
4F806	ASCII header block length, words	1 ... 32
4F807	Last node address to communicate with this module	1 ... 64
4F808	Remaining write reservation time, seconds	0 ... 60
4F809	Remaining outputs holdup time, units of 10 milliseconds	30 ... 6000

### 4FC01 ... 4FC20 hex / 464512 ... 464544 dec — ASCII Header Block

These registers contain a brief ASCII text description of the module. The length of the block is 1 ... 64 bytes, as specified in word 6 of the module status block (reference 4F806). The registers can be read, but cannot be written into.

## B .2 Checking TIO Registers With MSTR

### Using the MSTR

Here is an overview of the MSTR function block, to assist your I/O network planning. Details about using ladder logic functions in your application program are provided in the *Modicon Ladder Logic Block Library User Guide*.

Figure 30 shows an overview of the MSTR block structure.

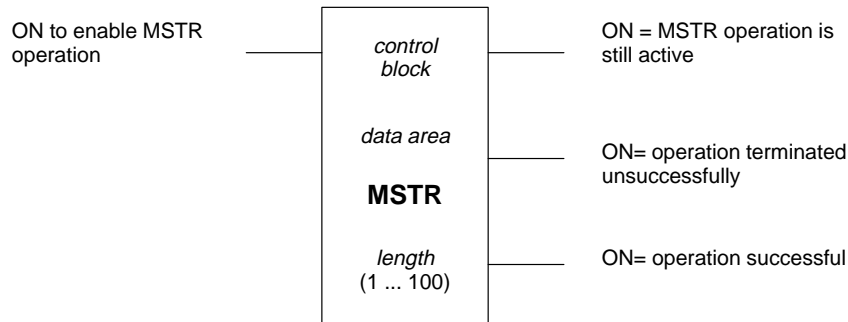


Figure 34 MSTR Block Overview

### The MSTR Control Block Field

The *control block* field specifies the starting register (4x) of a group of nine contiguous registers that define the operation of the MSTR block.

Register	Purpose
4XXXX (starting)	Operation: 1 = Write, 2 = Read (Write not allowed to some TIO registers — see section B .1)
4XXXX + 1	Returned error status
4XXXX + 2	Data length: quantity of registers
4XXXX + 3	Starting register
4XXXX + 4	High byte: 0 = CPU, other = NOM slot: low byte = Drop node address
4XXXX + 5	0 for I/O network
4XXXX + 6	0 for I/O network
4XXXX + 7	0 for I/O network
4XXXX + 8	0 for I/O network

### The MSTR Data Area Field

The *data area* field specifies the starting register (4x) of a group of contiguous registers. This area is the source of outgoing data for a write operation, and the destination for incoming data for a read operation.

**The MSTR Length Field**

The *length* field specifies the maximum quantity of registers allocated in the data area, in the range 1 ... 100.

**B .2.1 TIO Module Registers**

See section B .1 for a description of the registers in TIO modules.

## B .3 Defining Write Privilege to TIO Modules

---

When a node performs the initial write to a TIO module, the module will reserve further write privilege to that node for a duration of 60 seconds, before timing out. If the current node continues to write to the module within 60-second intervals, that node retains the write privilege. If the timeout is allowed to expire, any node may write to the module and the write privilege will be given to the next node that does so.

The 60-second timeout value is fixed, and is not accessible to the application. Note that the timeout process applies only to the write privilege to a module. Reading of input data or status from the module is allowed to any node.



# Appendix C

## Comparing Modbus Plus I/O to Remote I/O

---

- ☐ Comparing Modbus Plus I/O and RIO Networks
- ☐ Comparing Modbus Plus I/O and RIO Throughput

## C .1 Comparing Modbus Plus I/O and RIO Networks

In addition to the use of Modbus Plus I/O networking in your I/O application, you can use Modicon S908-based Remote I/O (RIO) networking. The manner in which network communication is handled by the controller - solving for the states of field inputs, and controlling the field outputs - differs between Modbus Plus I/O and Remote I/O.

Figure 35 illustrates the events during a typical scan of a controller in which both Modbus Plus I/O and RIO networks are present.

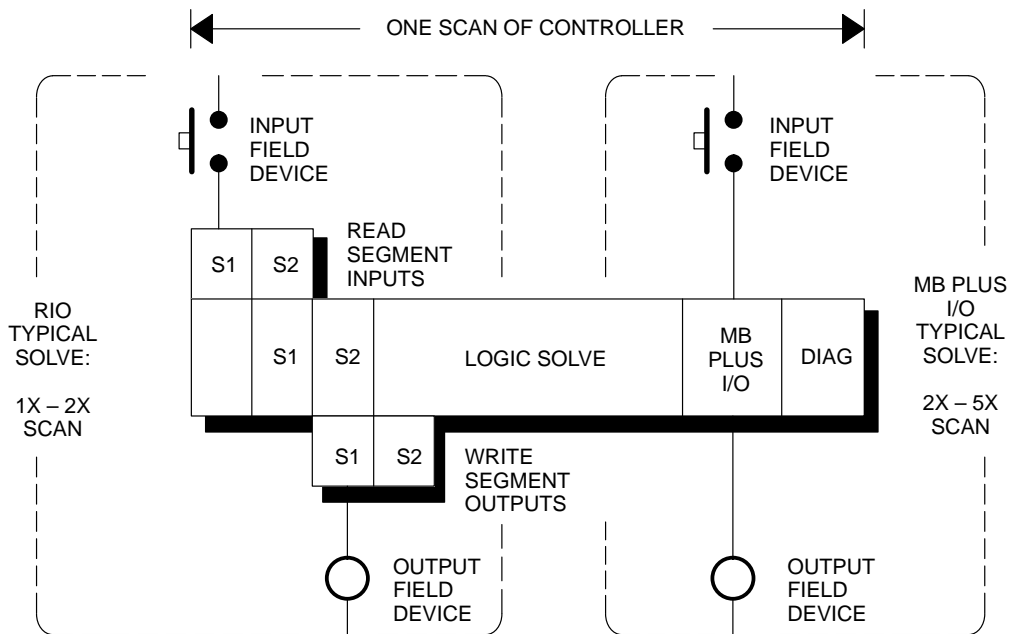


Figure 35 How the Controller Solves for Modbus Plus I/O and RIO Networks

### RIO

The controller handles RIO inputs and outputs in an 'order of solve' sequence that is determined by its Segment Scheduler, as setup during the controller's configuration. For a given segment, inputs are solved before outputs. The maximum time required for the controller to control a segment's outputs, based on sensing the states of its inputs, is (2 x controller Scan Time).

### **Modbus Plus I/O**

All of the application's current Modbus Plus I/O input and output states are processed at the end of the controller scan. Modbus Plus I/O control is not associated with the controller's Segment Scheduler.

In addition to updating the I/O states in its application program during its scan, the controller must service the I/O drops. The controller receives new input states from I/O drops, and sends its current output states to the drops, when it acquires the network token. The servicing speed is dependent upon the timing of the token rotation.

In applications in which a small logic program is running in the controller, with a network consisting of many I/O drops and a large average message size, the token rotation time can exceed the controller scan time. Because the token rotation is asynchronous to the controller scan, multiple controller scans may be required before a given set of outputs can be updated from the received states of a given set of inputs.

The minimum time required for the network I/O inputs to be sensed, and the I/O outputs to be controlled from the received input states, is (2 x controller Scan Time). Typical times can range up to (5 x controller Scan Time), depending upon the token rotation time.

The Modbus Plus I/O system designer can influence the token rotation time in the layout of the I/O network architecture. Primarily this can be done by separating the system into multiple smaller networks, and by keeping a network's node count as low as possible where maximum speed is needed.

## C .2 Comparing Modbus Plus I/O and RIO Throughput

The table below summarizes throughput times for three types of networks, based on actual test results:

- ☐ S908-based Remote I/O
- ☐ Controller-based Modbus Plus I/O
- ☐ Network Option Module-based Modbus Plus I/O

The network used in this comparison consists of five I/O drops, with four I/O modules per drop.

Average Throughput (ms)			
Program Size	S908 RIO	Modbus Plus I/O Controller	Modbus Plus I/O NOM
2K	11.79	21.09	29.24
4K	15.96	25.50	32.98
6K	19.56	27.89	36.26
8K	24.33	34.03	42.06
10K	27.33	37.05	44.55

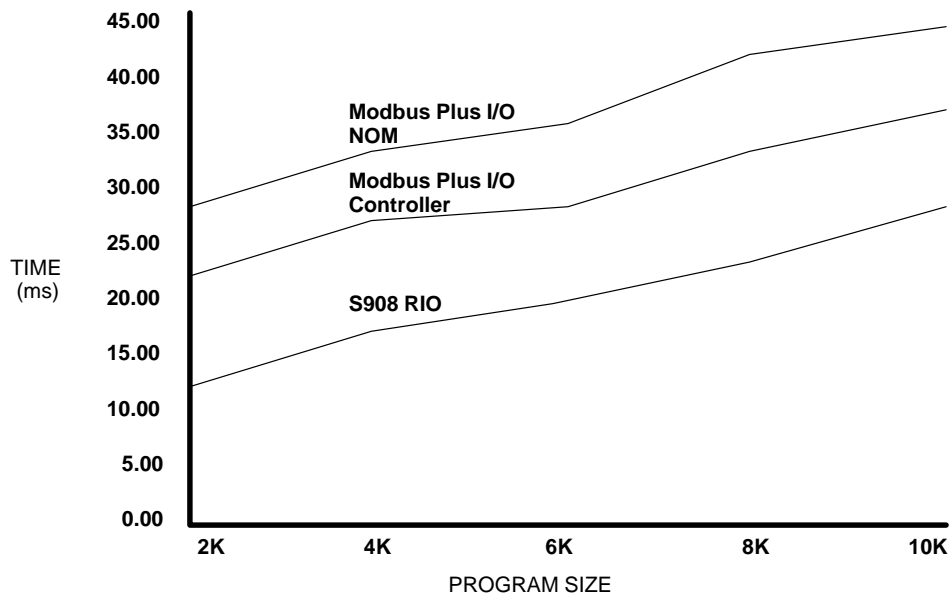


Figure 36 Comparison of Modbus Plus I/O and RIO

Program Size Segments/ Drops	Scan (ms) E785	Scan (ms) 113/213-04	Throughput E785	Throughput 113/213-04
<b>2K Program</b>				
2/2	7.39	6.20	11.95	9.60
3/3	8.96	7.51	12.34	11.04
4/4	11.19	9.48	14.02	11.24
5/5	13.78	11.39	15.21	13.64
6/6	16.35	13.71	17.15	13.42
Average	11.53	9.66	14.13	11.79
<b>4K Program</b>				
2/2	10.62	9.07	16.56	15.13
3/3	12.10	10.24	17.32	14.68
4/4	14.79	12.16	20.06	15.14
5/5	17.14	14.24	21.55	15.44
6/6	19.62	16.30	21.44	19.40
Average	14.85	12.40	19.39	15.96
<b>6K Program</b>				
2/2	13.78	11.25	20.70	17.43
3/3	15.31	12.54	20.49	17.76
4/4	17.69	14.67	23.74	20.45
5/5	20.07	16.58	23.28	20.58
6/6	22.38	18.56	27.01	21.59
Average	17.85	14.72	23.04	19.56
<b>8K Program</b>				
2/2	18.15	15.06	28.48	22.53
3/3	19.66	16.19	28.61	24.30
4/4	21.94	18.08	30.98	22.17
5/5	24.35	20.08	28.01	25.50
6/6	26.74	22.11	31.12	27.17
Average	22.17	18.30	29.44	24.33
<b>10K Program</b>				
2/2	20.65	17.03	32.13	25.30
3/3	22.06	18.14	31.00	27.23
4/4	24.38	20.12	34.36	27.90
5/5	26.68	22.05	32.78	27.77
6/6	29.08	23.94	36.01	28.45
Average	24.57	20.26	33.26	27.33



# Glossary

---

**address**

The logical identification of a networked node device for the purpose of sending or receiving messages. On Modbus Plus networks up to 64 nodes can be present, each with a unique address.

**bridge**

A device that interconnects two or more networks.

**Bridge Multiplexer**

A Modicon device that interconnects a Modbus Plus network with up to four Modbus devices or networks, or four RS232 or RS485 serial devices.

**Bridge Plus**

A Modicon device that joins two Modbus Plus networks together.

**coaxial cable**

A two-conductor cable in which an inner conductor is the signal path and an outer conductor is a shield. A dielectric separates the conductors.

**CPU**

Central Processing Unit, synonymous with Programmable Controller.

**DIO**

See **Distributed Input/Output**.

**DIO Drop Adapter**

A module that mounts in a backplane at a remote site in the application, and which provides the communication link between the Modbus Plus network and the backplane's I/O modules. The DIO Drop Adapter also has a power supply for furnishing operating voltages to the I/O modules.

**DIO Map**

The portion of a Programmable Controller's software configuration that specifies how communication between the controller and DIO Drop Adapter I/O modules is to take place. Table entries are typically made using panel software, such as the Modicon Modsoft product. Also see **Peer Cop Table**.

**Distributed Input/Output**

An industrial control system in which input and output modules are located in close proximity to the field devices in the industrial process. Modules communicate with a programmable controller by transmitting and receiving messages on a Modbus Plus network.

**holdup time**

A duration of time during which an output module will continue to hold its outputs in their current states. As a module receives each new command it updates its outputs as directed by the command. If a subsequent command is not received, the module continues to hold the outputs for the specified holdup time. If the time expires, the module sets its outputs to a predefined state.

For DIO drops, holdup times are specified in the controller's DIO Map. Modules can be configured in the DIO Map to hold their last value, or to assume a defined state, in the event of the holdup time expiration.

For TIO drops, holdup times can be loaded using MSTR function blocks. Modules revert to a logical 0 (zero) condition in the event of the holdup time expiration.

**host computer**

A computer which controls other computers and devices. In an industrial network, the host computer specifies the current requirements for the operation of remote nodes, and is the destination for summary data reports about the performance of the process.

**I/O**

Acronym for Input/Output. Refers to modules that are used to control devices in an industrial process. See **Distributed Input/Output**.

**I/O Network**

See **Modbus Plus I/O Network**.

**LAN**

Local Area Network. An interconnection of devices in which data is transferred without the use of public communications services. Modbus Plus is an example of a LAN for controlling and monitoring industrial processes.



**master**

A networked device which controls other devices to which it connects. It initiates transactions, and schedules and transmits tasks to a slave device. See **slave**.

On a Modbus Plus I/O network, the Programmable Controller or Network Option Module operates as the master node, controlling DIO Drop Adapters and TIO modules which operate as slave nodes.

**medium**

The entire cable system: the network cable, taps, connectors, and terminators.

**MMI**

Man-Machine Interface. See **User Interface**.

**Modbus**

An industrial networking system that uses RS232 serial master-slave communications at data transfer rates of up to 19.2 k baud.

**Modbus Plus**

An industrial networking system that uses token-passing peer-to-peer communications at data transfer rates of one megabits per second. The network medium is shielded twisted-pair cable. Up to 64 nodes can be present on each network.

**Modbus Plus I/O Network**

A Modbus Plus network consisting of hardware components that are specifically designed for high-speed control of field input and output devices in industrial automation applications.

Each Modbus Plus I/O network has a Programmable Controller or Network Option Module that operates as the master controlling node on the network. Field input/output devices are serviced through DIO Drop Adapters or TIO Modules placed at remote locations.

**network**

The interconnection of devices sharing a common data path and protocol for communication. On Modbus Plus, the devices share in the passing of a common token frame to gain sequential access for sending messages.

**Network Option Module**

A hardware module that is mounted into a common backplane together with a Programmable Controller. The module communicates with the controller over the backplane. It connects to a Modbus Plus network and provides the controlling point for communication between the controller's application program and the node devices on the network.

**node**

A device that has a direct point of access to a communications network. On Modbus Plus, a node is any device that is physically connected to the network.

Programmable Controllers, Network Option Modules, DIO Drop Adapters, and TIO modules are examples of nodes that can exchange data on the network. These devices are assigned node addresses to allow them to be addressed for messages.

Repeaters are nodes that are used to extend the network cable length or to allow the network to accommodate additional data-handling nodes. Because they do not function as final destinations for messages, these devices are not assigned node addresses.

**NOM**

See **Network Option Module**.

**path**

A logical connection established between two nodes for the purpose of exchanging messages. A path is established in the source node as it originates a command to a destination node, and is established in the destination node as it receives the command. The path is maintained by both nodes until the final information transfer occurs as defined by the command, or until an error exception is transferred.

Commands using paths are those originated by MSTR function blocks in the application program, or by operator action at a Programmer device connected to the network. Modbus Plus nodes typically allow for multiple concurrent transactions of these kinds to be in process, each using an independent path between the source and destination nodes.

Transactions that do not use paths are those that are defined in the controller's DIO Map or Peer Cop table. These are normally used for Modbus Plus I/O network applications, in which the CPU/NOM communicates with DIO Drop Adapters and TIO modules in the control of I/O processes.

**Peer Cop**

A method of peer-to-peer communication between networked devices in which data is transferred as part of the passing of tokens between nodes. Each node passes the token in the network's address sequence, and can be configured to transmit data in addition to the token. All nodes monitor the token passes, and can be configured to extract data from them. Nodes are setup for Peer Cop transfers as part of their initial configuration, and continue using Peer Cop as long as they are active on the network.

**Peer Cop Table**

The portion of a Programmable Controller's software configuration that specifies how communication between the controller and TIO modules is to take place. Table entries are typically made using panel software, such as the Modicon Modsoft product. Also see **DIO Map**.

**peer-to-peer communication**

A method of communication in which any device can initiate data transfer. This is the method used on Modbus Plus.

**port**

The external connector on a device at which the network cable is attached.

**protocol**

A set of rules used mutually by two or more nodes to communicate.

**Repeater**

A Modicon device that interconnects two sections of a Modbus Plus network. It is used to extend the network cable length or to allow the network to accommodate additional data-handling nodes.

**routing path**

In Modbus Plus, the sequence of device node addresses through which a message will be routed to its final destination.

**section**

A contiguous grouping of cable segments, together with their node devices, connected directly to form a signal path that does not pass through any Repeater. The minimum length of a section can be 10ft (3m), the same as one segment. The maximum length can be 1500 ft (450 m). One section supports up to 32 nodes. Repeaters can be used to join two sections for greater cable lengths and more nodes. See **segment**.

**segment**

The combination of: a continuous length of trunk cable connecting a pair of taps; the two taps; and the drop cables between the two taps and their node devices. One or more segments form a section of the network. See **section** and **tap**.

**serial port**

An communication port at which data is transferred one bit at a time.

**slave**

A networked device which is controlled by another device. Slave devices do not initiate data transactions. They respond to commands or requests initiated by a master device. See **master**.

On a Modbus Plus I/O network, DIO Drop Adapters and TIO modules operate as slave nodes under the control of a Programmable Controller or Network Option Module, which operates as a master node.

**slot time**

The amount of time representing the worst case time any station on the network must wait for a response from another station. It is based upon the response time of the network's slowest station and the cable system's propagation delay.

**system**

A set of hardware devices and their associated software capable of performing the functions of information processing and device control without significant dependence on other equipment.

**tap**

A passive electrical device that joins segments of the trunk cable, or terminates the trunk cable at its two end sites. It also provides a connection for the drop cable to the node device at the tap site. See **terminator**.

**Terminal Block I/O**

An input/output module that connects directly to the Modbus Plus network and to field devices at a remote site in the application. Terminal Block I/O modules provide cost-effective I/O servicing in applications in which ease of installation and servicing, panel space, and cost are major considerations.

**terminator**

A resistive load placed at the end of a cable to prevent data signals from reflecting back into the data path. The signals are terminated with the same impedance as the characteristic impedance of cable system. On Modbus Plus, each tap contains a terminating resistor with two jumpers. The termination is effective when the jumpers are installed. The tap at each of the two ends of the cable section has its terminating jumpers installed. The tap at each inline point has them removed. See **section**.

**TIO**

See **Terminal Block I/O**.

**token**

In data transmission, a frame passed on a network that gives a networked device the current authority to transmit.

**token bus**

A network access method between two or more devices in which the procedure for sending data is based upon the passing of a token for access to the network. See **token**.

**transaction**

The complete and successful transfer of a message between networked devices.

**trunk**

The main cable of the system that interconnects the network nodes. On Modbus Plus, the trunk cable runs directly between pairs of taps.

**UI**

See **User Interface**.

**User Interface**

A device that provides information to an operator about the status of an application, and provides a means for controlling the application. For example, the Modicon PanelMate Plus is a User Interface with a display panel and pushbuttons. It can be connected to a Modbus Plus network at a remote site for monitoring and control.



# Index

---

## B

BP85 Bridge Plus, 11

## C

cable

- drop, 7
- trunk, 7

cable segment, explained, 6

connecting network devices

- for deterministic I/O servicing, 30
- programmer, 36
- programmer and UI, 40

controllers, multiple, 54

## D

DIO Map

- drop, 66
- during startup, 14
- in drop, 14
- in host, 14
- in multiple hosts, 58

DIOH function block, 64

Drop Adapters

- configuration, 14
- with multiple controllers, 54, 56, 58
- write privilege, 16

drop node, explained, 6

## H

health

- DIO drop, 64
- module, 68

holdup time, 15, 29

## I

I/O system

- configuration, 2, 4
- configuration options, 24
- Drop Adapters, 4
- optimizing, 26
- TIO modules, 4

## M

master node, explained, 6

messages

- handling, 18
- non-DIO , 19

MSTR function block

- with DIO drops, 66
- with TIO modules, 74

multiple controllers

- example, 56
- requirements, 58

## N

network

- an overview, 6, 12
- comparing DIO and RIO, 78, 80
- deterministic timing, 30
- Drop Adapter configuration, 14
- how nodes are known, 14
- operation, 12
- startup, 14
- terminology, 6
- TIO Module configuration, 15

network section, explained, 6

node, explained, 6

## P

path

- explained, 86
- impact on I/O timing, 32

Peer Cop, during startup, 14

performance

- and node count, 22
- estimating data throughput, 48
- factors affecting, 22

privilege

- read only, 58
- read/write, 16
- single master only, 58
- write, single master only, 16

programmer, connecting, 36, 40

## **R**

Repeaters, FR85/RR85, 9

## **T**

tap, 7

TIO Modules

configuration, 15

registers layout, 70, 74

with multiple controllers, 55, 58

write privilege, 17, 76

token

explained, 12

rotation sequence, 12

token reconstitution time, 44

token rotation time

in deterministic I/O networks, 30

in multifunction networks, 37, 41

## **U**

user interface, connecting, 40