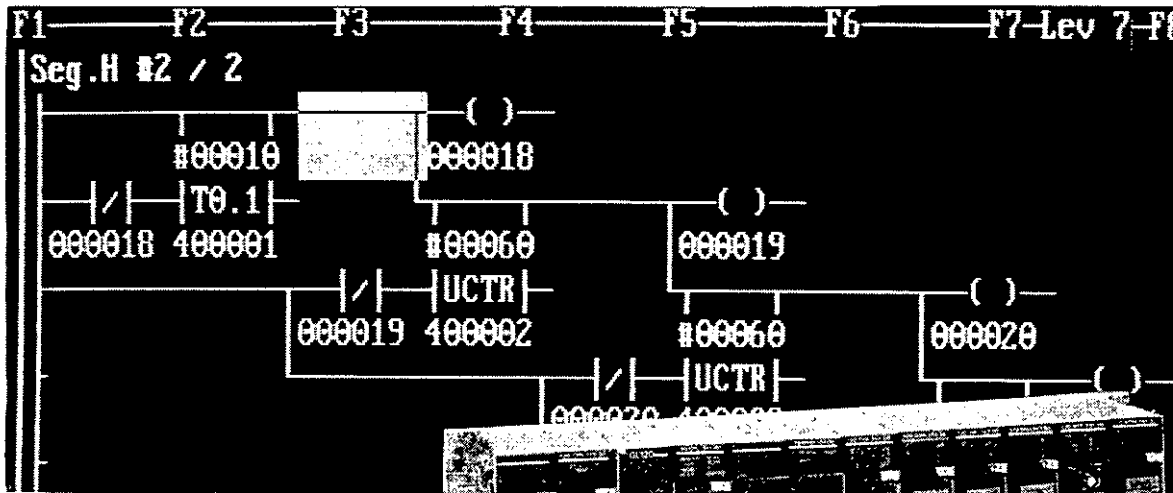


# MEMOCON GL120, GL130 SOFTWARE USER'S MANUAL Vol.1



# Manual Contents

This manual describes the methods used to program the MEMOCON GL120 and GL130 Programmable Controllers (PLCs). It also introduces the ladder logic elements and instructions. Please read this manual carefully and be sure you understand the information provided before attempting to program a MEMOCON PLC.

This is volume 1 of the two volumes of the *MEMOCON GL120, GL130 Software User's Manual*.

## Visual Aids

The following aids are used to indicate certain types of information for easier reference.



Indicates references for additional information.



Indicates important information that should be memorized.



Indicates application examples.



Indicates supplemental information.



Indicates a summary of the important points of explanations.

### Note

Indicates inputs, operations, and other information required for correct operation but that will not cause damage to the device.



Indicates definitions of terms used in the manual.

## NOTICE

The following conventions are used to indicate precautions in this manual. Failure to heed precautions provided in this manual can result in injury to people or damage to the products.



**WARNING** Indicates precautions that, if not heeded, could possibly result in loss of life or serious injury.



**Caution** Indicates precautions that, if not heeded, could result in relatively serious or minor injury, damage to the product, or faulty operation.

©Yaskawa, 1996

---

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, mechanical, electronic, photocopying, recording, or otherwise, without the prior written permission of Yaskawa. No patent liability is assumed with respect to the use of the information contained herein. Moreover, because Yaskawa is constantly striving to improve its high-quality products, the information contained in this manual is subject to change without notice. Every precaution has been taken in the preparation of this manual. Nevertheless, Yaskawa assumes no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained in this publication.

# CONTENTS

3.1.3	Number of I/O Allocation Points .....	3-6
3.1.4	I/O Module Arrangement Example .....	3-7
3.1.5	I/O Reference Numbers .....	3-8
3.2	Local Channel I/O Allocation .....	3-9
3.2.1	I/O Reference Allocation .....	3-9
3.2.2	Module Type .....	3-11
3.2.3	I/O Data Format (Zoom Operation) .....	3-12
3.2.4	Local Channel Allocation Example .....	3-16
3.3	High-speed Segment I/O Allocation .....	3-18
3.3.1	Local Channel High-speed Segment I/O Allocation .....	3-18
3.3.2	Remote Channel High-speed Segment I/O Allocation .....	3-18
3.4	Remote Channel I/O Allocation .....	3-19
3.4.1	I/O References .....	3-19
3.4.2	Module Type .....	3-19
3.4.3	I/O Data Format .....	3-19
3.4.4	Remote Channel Allocation Example .....	3-20
<b>CHAPTER 4</b>	<b>Overview of Instructions .....</b>	<b>4-1</b>
4.1	Instructions .....	4-2
4.2	Instruction Outlines .....	4-8
4.2.1	Basic Instructions .....	4-9
4.2.2	Math Instructions .....	4-14
4.2.3	Data Transfer Instructions .....	4-41
4.2.4	Indexed Block Transfer Instructions .....	4-55
4.2.5	Matrix Instructions .....	4-60
4.2.6	Bit Manipulation Instructions .....	4-72
4.2.7	Data Conversion Instructions .....	4-76
4.2.8	Other Data Manipulation Instructions .....	4-81
4.2.9	System Status Monitoring Instruction .....	4-91
4.2.10	Sequencers .....	4-92
4.2.11	Program Control Instructions .....	4-94
4.2.12	I/O Control Instructions .....	4-101
<b>APPENDIXES</b>		
A	Instruction Processing Times .....	A-1

# Introduction and Precautions

# 1

This chapter introduces general information, including basic information and precautions for the use of this manual and the software. **You must read this chapter before attempting to read the rest of the manual or using the product.**

<b>1.1 Overview of Manuals .....</b>	<b>1-2</b>
<b>1.2 Safety Precautions .....</b>	<b>1-5</b>
<b>1.3 Using this Manual .....</b>	<b>1-6</b>

## 1.1 Overview of Manuals

- This manual describes the following items for MEMOCON GL120 and GL130 Programmable Controllers.
  - 1) Operating principles
  - 2) I/O allocation
  - 3) Overview of instructions
  - 4) Instruction processing times
- Read this manual carefully in order to properly use ladder programming for MEMOCON GL120 and GL130 Programmable Controllers. Also, keep this manual in a safe place so that it can be used whenever necessary.
- Refer to the following related manuals as required.

	Manual Name	Manual Number	Content
CPU Module	MEMOCON GL120, GL130 Hardware User's Manual	SIEZ-C825-20.1	Describes the following for the GL120 and GL130: <ol style="list-style-type: none"> <li>1) System configuration</li> <li>2) System components</li> <li>3) Functions and specifications</li> <li>4) Installation and wiring</li> <li>5) Panel layout and hole dimensions</li> <li>6) External dimensions</li> </ol>
	MEMOCON GL120, GL130 Software User's Manual, Volume 2	SIEZ-C825-20.12	Describes the programming instructions used to create ladder programs for the GL120 and GL130. The following instructions are described in other manuals. <ol style="list-style-type: none"> <li>1) Expansion Math Instructions: Software User's Manual (Vol. 3)</li> <li>2) Process Control Instructions: Software User's Manual (Vol. 4)</li> <li>3) Communications Instructions COM: COM Instructions User's Manual FBUS: PC Link Module User's Manual MSTR: MEMOBUS PLUS User's Manual</li> <li>4) Motion Control Instructions (ladder motion instructions) 4-axis Motion Module MC20 Software User's Manual</li> <li>5) Motion Language 4-axis Motion Module MC20 Software User's Manual</li> </ol>

	Manual Name	Manual Number	Content
CPU Module	MEMOCON GL120, GL130 Software User's Manual, Volume 3	SIEZ-C825-20.13	Describes expansion math instructions (floating point math instructions, etc.) used for the GL120 and GL130.
	MEMOCON GL120, GL130 Software User's Manual, Volume 4	SIEZ-C825-20.14	Describes process control instructions used for the GL120 and GL130.
I/O Modules	MEMOCON GL120, GL130 120-series I/O Module User's Manual	SIEZ-C825-20.22	Describes the functions, specifications, and usage of the 120-series I/O Module.
Special Purpose Modules	MEMOCON GL120, GL130 120-series Counter Module User's Manual	SIEZ-C825-20.24	Describes the functions, specifications, and usage of the 120-series Counter Module.
	MEMOCON GL120, GL130 120-series Uniwire Interface Module User's Manual	SIEZ-C825-20.26	Describes the functions, specifications, and usage of the 120-series Uniwire Interface Module.
Motion Modules	MEMOCON GL120, GL130 Motion Module MC10 User's Manual	SIEZ-C825-20.41	Describes the functions, specifications, and usage of the 1-axis Motion Module MC10.
	MEMOCON GL120, GL130 Motion Module MC20 Hardware User's Manual	SIEZ-C825-20.51	Describes the functions, specifications, and usage of the 4-axis Motion Module MC20.
	MEMOCON GL120, GL130 Motion Module MC20 Software User's Manual	SIEZ-C825-20.52	Describes motion control instructions (ladder motion instructions) and motion program language used for the 4-axis Motion Module MC20.
Man-machine Interface	MEMOCON GL120, GL130 Teach Pendant TB120 User's Manual	SIEZ-C825-60.3	Describes the functions, specifications, and usage of the Teach Pendant TB120.
	MEMOCON Micro, GL120, GL130 Programming Panel P120 (MEMOSOFT) User's Manual	SIEZ-C825-60.7	Describes the functions, specifications, and usage of the Programming Panel P120 (with built-in MEMOSOFT).
	MEMOCON Micro, GL120, GL130 MEMOSOFT for DOS User's Manual	SIEZ-C825-60.10	Describes the functions and usage of the MEMOSOFT for DOS.
Communications Modules	MEMOCON GL120, GL130 PC Link Module User's Manual	SIEZ-C825-70.4	Describes the functions, specifications, and usage of the PC Link Module for the GL120 and GL130.
	MEMOCON GL120, GL130 MEMOBUS PLUS User's Manual	SIEZ-C825-70.5	Describes the functions, specifications, and usage of the MEMOBUS PLUS.
	MEMOCON GL120, GL130 Coaxial Remote I/O System User's Manual	SIEZ-C825-70.8	Describes the functions, specifications, and usage of the Coaxial Remote I/O System for the GL120 and GL130.

	<b>Manual Name</b>	<b>Manual Number</b>	<b>Content</b>
Commu- nications Modules	MEMOCON Micro, GL120, GL130 MEMOBUS User's Manual	SIEZ-C825-70.13	Describes the functions, specifications, and usage of the MEMOBUS.
	MEMOCON Micro, GL120, GL130 COM Instructions User's Manual	SIEZ-C825-70.14	Describes the functions, specifications, and usage of the COM instructions. It also describes the specifications and usage of the MEMOBUS Module.

- Thoroughly check the specifications and conditions or restrictions of the product before use.

## 1.2 Safety Precautions

- MEMOCON was not designed or manufactured for use in devices or systems that concern human lives. Users who intend to use the product described in this manual for special purposes such as devices or systems relating to transportation, medical, space aviation, atomic power control, or underwater use must contact Yaskawa Electric Corporation beforehand.
- This product has been manufactured under strict quality control guidelines. However, if this product is to be installed in any location in which a failure of MEMOCON involves a life and death situation or in a facility where failure may cause a serious accident, safety devices **MUST** be installed to minimize the likelihood of any accident.
- Any illustrations, photographs, or examples used in this manual are provided as examples only and may not apply to all product to which this manual is applicable.
- The products and specifications described in this manual or the content and presentation of the manual may be changed without notice to improve the product and/or the manual. A new version of the manual will be re-released under a revised document number when any changes are made.
- Contact your Yaskawa representative or a Yaskawa office listed on the back of this manual to order a new manual whenever this manual is damaged or lost. Please provide the document number listed on the front cover of this manual when ordering.
- Contact your Yaskawa representative or a Yaskawa office listed on the back of this manual to order new nameplates whenever a nameplate becomes worn or damaged.
- Yaskawa cannot make any quality guarantee for products which have been modified. Yaskawa assumes no responsibility for any injury or damage caused by a modified product.



## 1.3 Using this Manual

This manual is written for the following people:

- Workers responsible for designing GL120 or GL130 ladder programs.
- Workers responsible for testing GL120 or GL130 ladder programs.
- Workers responsible for debugging GL120 or GL130 ladder programs during trial operation.
- Workers responsible for maintaining GL120 or GL130 ladder programs.
- **Basic Terms**

In this manual, the following terms have the meanings described below.

- **PLC = Programmable (Logic) Controller**
- **PP = Programming Panel**
- **GL120, GL130 = MEMOCON GL120 and/or MEMOCON GL130 Programmable Controller**
- **Technical Terms**

The bold technical terms in this manual are briefly explained in the **Glossary** provided at the bottom of the page. An example is shown below.



---

### Glossary

The following types of terms are described.

- Specific sequence control terms required for explanation of functions.
- Terms that are specific to programmable controllers and electronic devices.

# Operating Principles

# 2

This chapter explains the operating principles of the GL120 and GL130.

<b>2.1</b>	<b>User Program Configuration</b> .....	<b>2-3</b>
2.1.1	Segments .....	2-3
2.1.2	Networks .....	2-4
2.1.3	Subroutines .....	2-9
<b>2.2</b>	<b>References</b> .....	<b>2-11</b>
2.2.1	What is a Reference? .....	2-11
2.2.2	I/O References .....	2-13
2.2.3	Internal References .....	2-16
2.2.4	Link References .....	2-23
2.2.5	Stepping Switch References .....	2-25
2.2.6	Motion References .....	2-26
2.2.7	Reference Data and User Programs .....	2-29
2.2.8	Changing Reference Numbers .....	2-30
<b>2.3</b>	<b>Internal Processing</b> .....	<b>2-31</b>
2.3.1	Internal Processing Flow .....	2-32
2.3.2	Startup Sequence .....	2-33
2.3.3	Scanning Cycle .....	2-33
2.3.4	Watchdog Timer .....	2-35
2.3.5	Network Numbers .....	2-35
2.3.6	Total Check .....	2-36
<b>2.4</b>	<b>Scanning</b> .....	<b>2-37</b>
2.4.1	Ladder Logic Program Solving and Scanning .....	2-37
2.4.2	Scan Time .....	2-42
2.4.3	Sweep Functions .....	2-44

## Chapter Table of Contents, Continued

---

<b>2.5</b>	<b>Segment Scheduler</b> .....	<b>2</b>
2.5.1	High-speed Scan .....	
2.5.2	The Segment Scheduler .....	
<b>2.6</b>	<b>Start Mode</b> .....	<b>2</b>
2.6.1	Starting Mode Setting .....	
2.6.2	Automatic RUN Operating Mode .....	
2.6.3	Normal Operating Mode .....	
<b>2.7</b>	<b>Disable Operation</b> .....	<b>2</b>
2.7.1	Disable Operation .....	

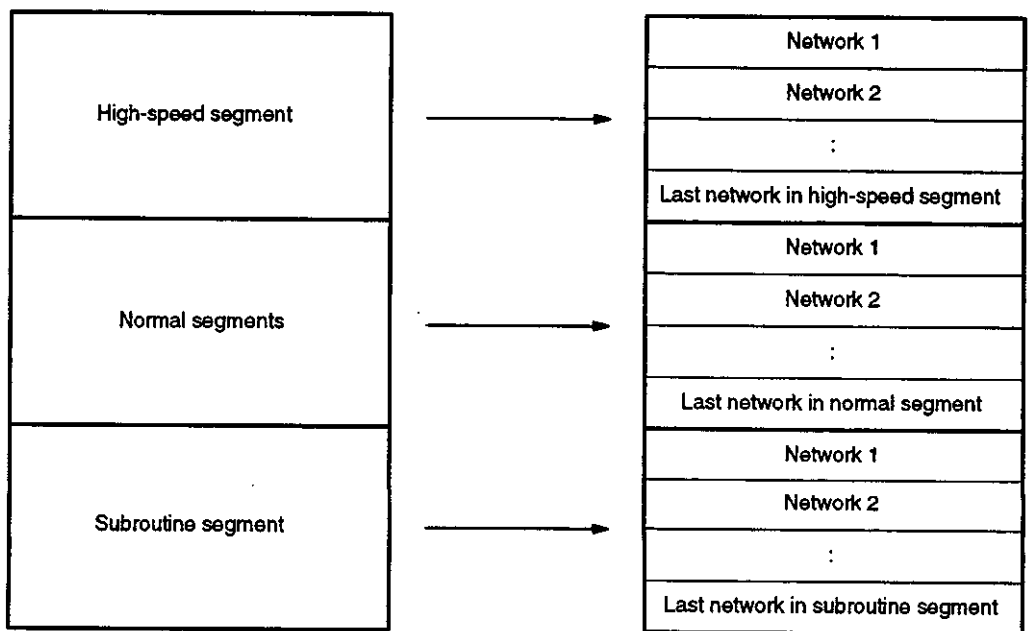
## 2.1 User Program Configuration

As shown in *Figure 2.1*, the user program is composed of a high-speed segment, normal segments, and a subroutine segment, and each of these is composed of networks.

2.1.1	Segments .....	2-3
2.1.2	Networks .....	2-4
2.1.3	Subroutines .....	2-9

### 2.1.1 Segments

- 1) There are three kinds of segments: high-speed, normal, and subroutine. The high-speed segment stores ladder logic that is solved during the high-speed scan, and the normal segments store ladder logic that is solved by the normal scan. The subroutine segment stores ladder logic that is solved as subroutines.



**Figure 2.1 User Program Configuration**

- 2) A maximum of 30 normal segments can be created, but there is only one normal segment set by default in the factory settings. For details regarding high-speed and normal segments, refer to section 2.5 *Segment Scheduler* of this chapter.
- 3) Up to 1,023 subroutines can be stored in the subroutine segment.

- 4) As shown in *Figure 2.2*, networks are automatically numbered in order from the beginning of each segment. If the high-speed scan is not used, the user program configuration will consist of only the normal segments, or only the normal and subroutine segments.

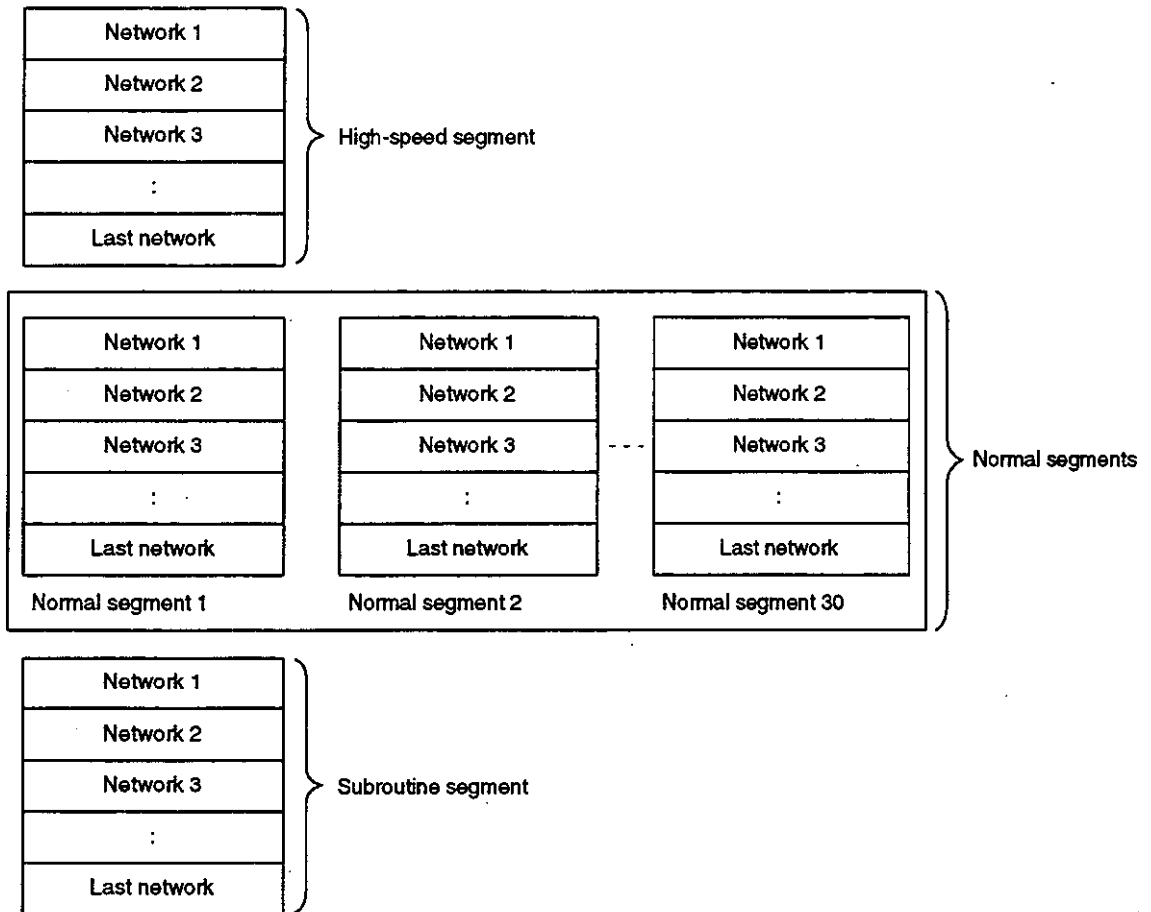


Figure 2.2 Segment and Network Numbers

## 2.1.2 Networks

- 1) A network is the smallest unit in the user program that could be considered as a separate program. One or more networks combine to form a "segment."
- 2) The horizontal lines in a network are called "rows," and the vertical lines are called "columns." As shown in *Figure 2.3*, a single network consists of seven rows and eleven columns. It can contain up to eleven elements per row and seven elements per column, for a maximum total of 77 elements.
- 3) The four types of coil (normal coils, link coils, MC coils, and MC control coils) can be programmed at any of the nodes in any row, from column 1 to column 11, but no element other than a coil can be programmed at column 11. In addition, no element can be programmed to the right of a coil in the same row as the coil.

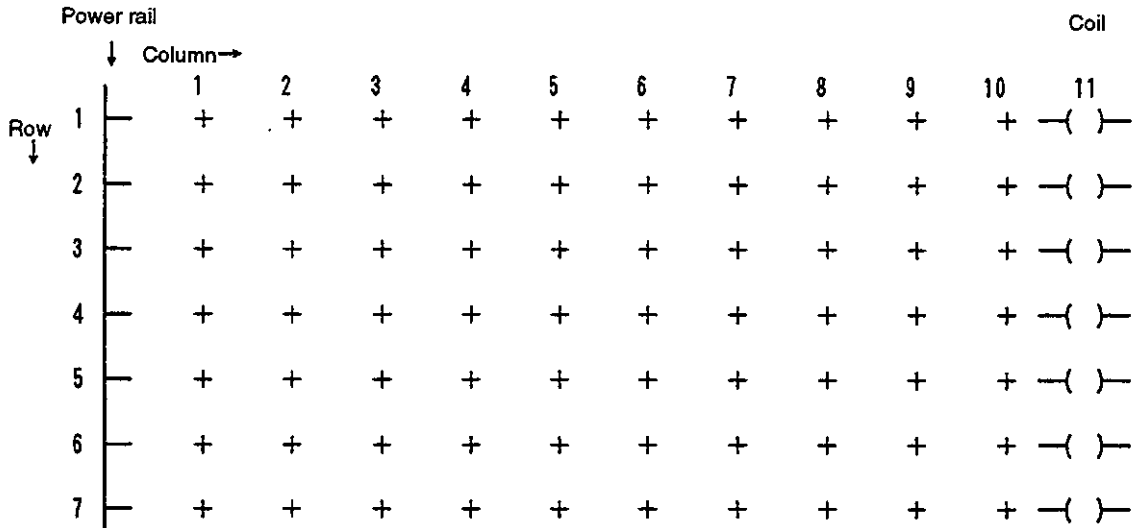


Figure 2.3 Network Configuration

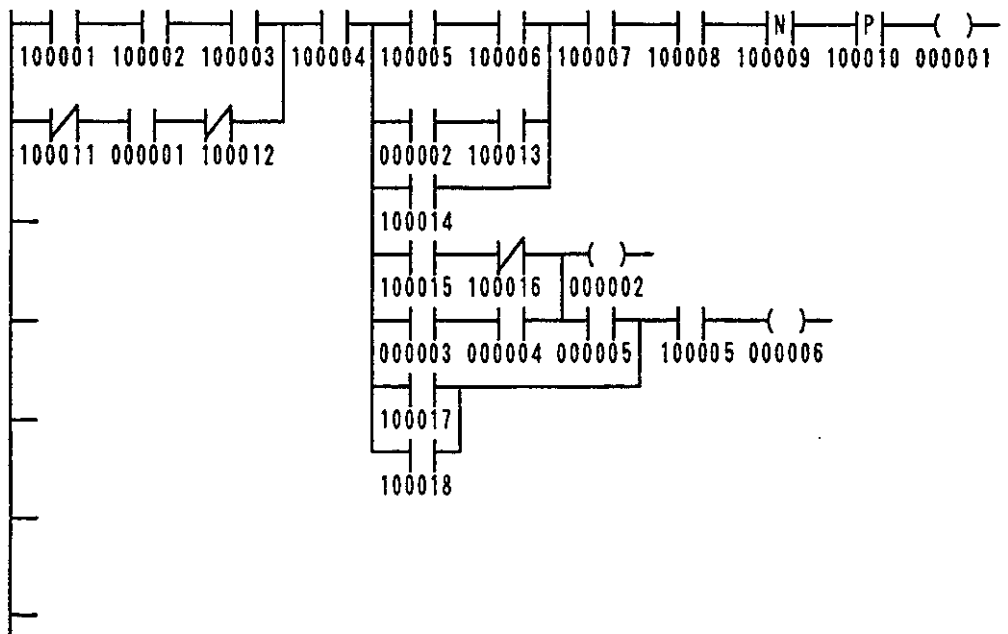
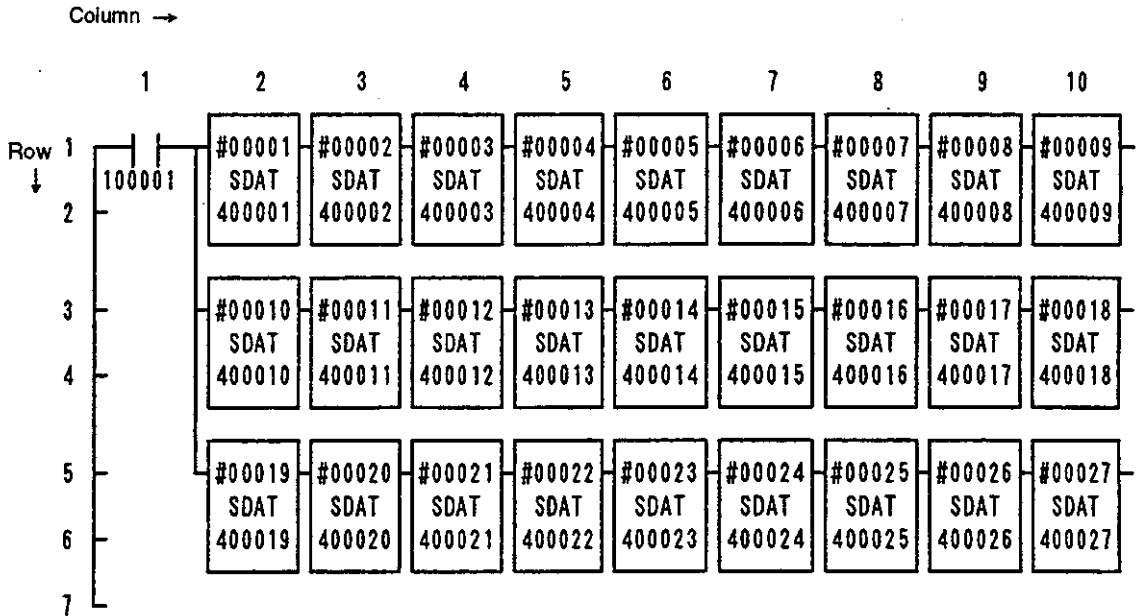


Figure 2.4 Network Program Example

- 4) Ladder logic can be freely created in a single network as long as it is kept within the range of seven rows and eleven columns. In the programming example shown in *Figure 2.5*, consecutive application instructions are connected horizontally.
- 5) Network numbers are automatically assigned to networks in order. There are two kinds of numbering. One relates to the segment position and the other to the user program as a whole.

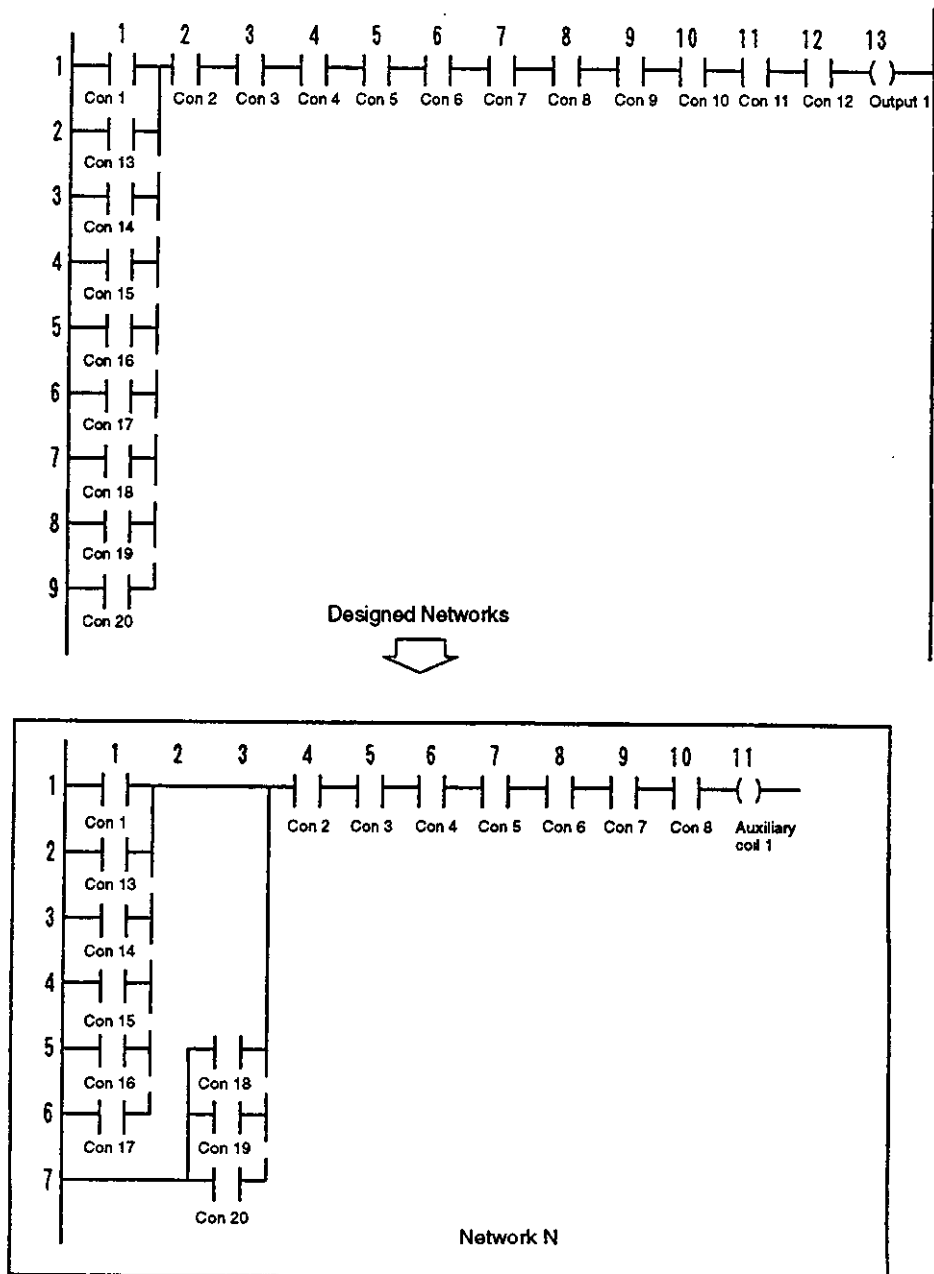
6) Network number processing is performed automatically for additions and deletions of elements within a single network, additions and deletions of networks, and insertion of a new network between two consecutive existing networks.



- When input relay 100001 turns ON, constants 1 to 27 are written to holding registers 400001 to 400027.

Figure 2.5 Ladder Logic Program Example

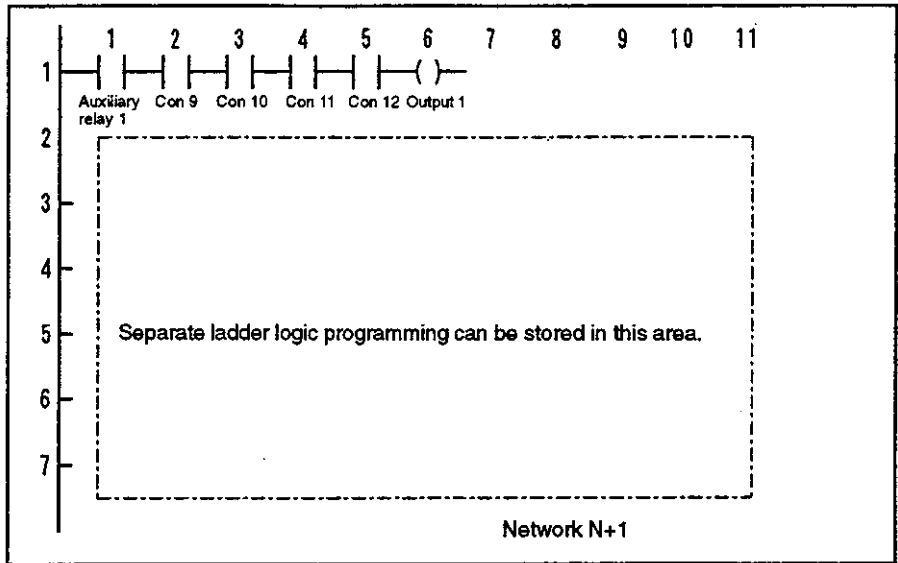
**Note** If a network contains a large number of interlock conditions (e.g., more than ten AND conditions or more than seven OR conditions), then use internal coils (auxiliary coils) and divide it into two or more networks as shown in *Figure 2.6*.



Note Con = Condition

Figure 2.6 Example of Division Into Two Networks





Note Con = Condition

Figure 2.6 Example of Division into Two Networks (Continued)

### 2.1.3 Subroutines

- 1) Up to 1,023 subroutines can be stored in the subroutine segment.
- 2) Networks enclosed between SUBROUTINE LABEL (LAB) and SUBROUTINE RETURN (RET) instructions are treated as a single subroutine. Individual subroutines are allocated subroutine numbers 1 to 1,023.
- 3) Each subroutine segment is configured by networks, just like other segments, and the order of solving elements within networks is the same as for a normal segment.
- 4) Subroutines are called by means of SUBROUTINE JUMP (JSR) instructions that have been programmed in the high-speed or normal segments. Nesting of subroutine calls within subroutines is also possible, by programming SUBROUTINE JUMP (JSR) inside of one subroutine to call another subroutine.

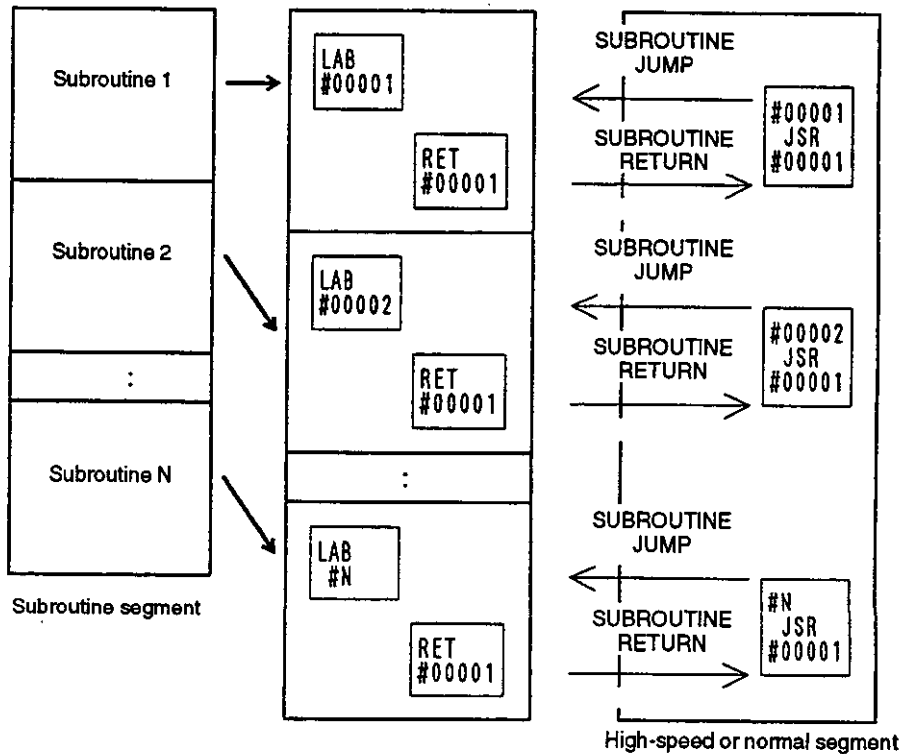


Figure 2.7 Subroutine Segments and Subroutines

- 5) LAB must be programmed only in the first row and first column of a network.
- 6) RET can be programmed anywhere in the network except for the coil column (i.e., the 11th column), but it is recommended that it be programmed in the first row and column of the network following the last network of the subroutine, as shown in Figure 2.8.

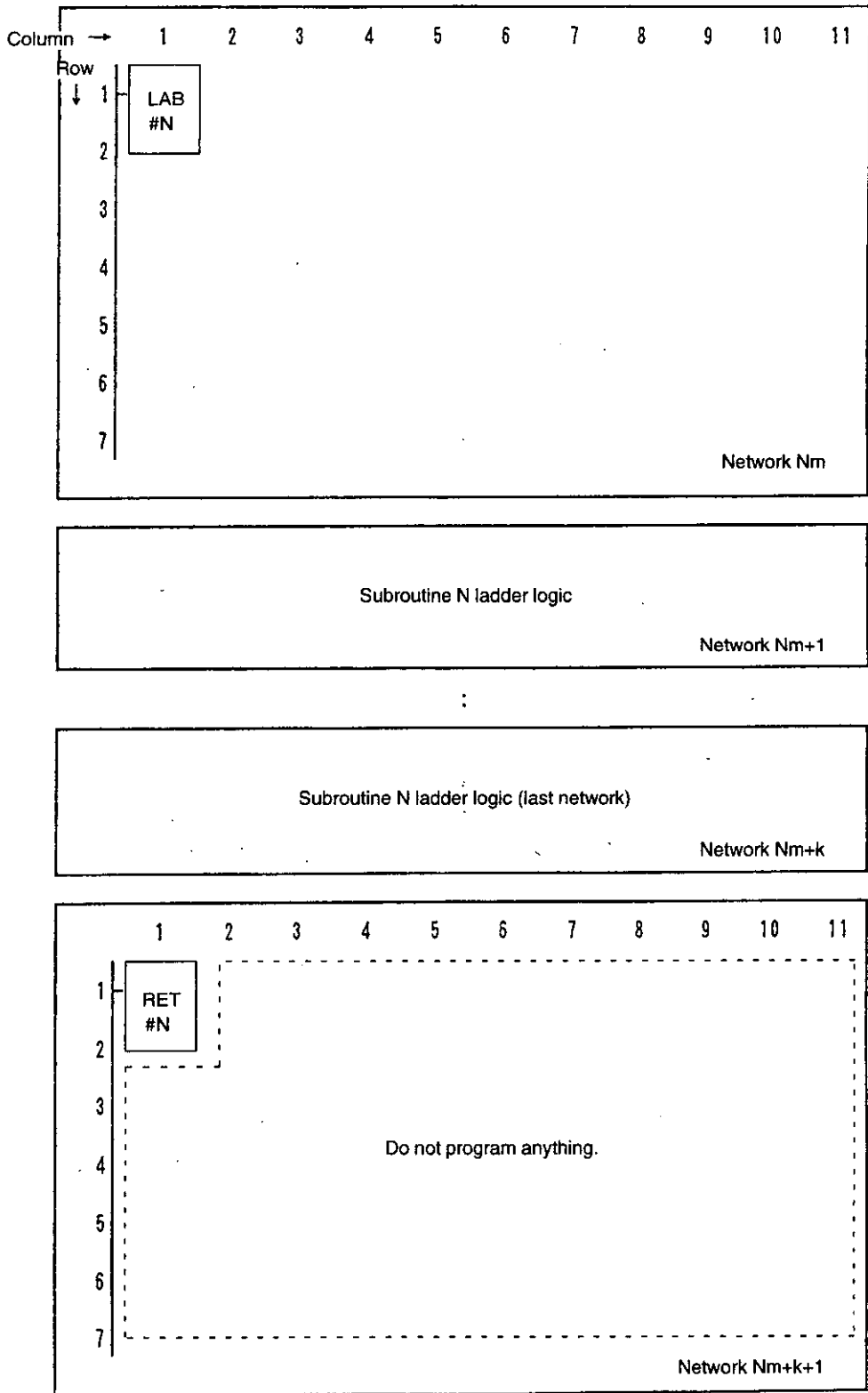


Figure 2.8 Subroutine Program Example

## 2.2 References

Contacts, registers (i.e., numeric storage locations), and coils are all assigned numbers called "references." These reference numbers are used to store, modify, and read the user program.

2.2.1	What is a Reference? .....	2-11
2.2.2	I/O References .....	2-13
2.2.3	Internal References .....	2-16
2.2.4	Link References .....	2-23
2.2.5	Stepping Switch References .....	2-25
2.2.6	Motion References .....	2-26
2.2.7	Reference Data and User Programs .....	2-29
2.2.8	Changing Reference Numbers .....	2-30

### 2.2.1 What is a Reference?

- 1) There are two kinds of references: digital (for ON/OFF data) and register (for numeric data). *Table 2.1* shows the references used for the GL120 and GL130.

**Table 2.1 List of References**

Type of Reference		Range of Reference for GL120 or GL130	
Digital references	Input relays	100001 to 101024 (I00001 to I01024)	
	Coils (Coils allocated for output are called output coils.)	000001 to 008192 (O00001 to O08192)	
	Link coils	Link coil 1	D10001 to D11024
		Link coil 2	D20001 to D21024
	Stepping switches	201001 to 232099 (S01001 to S32099)	
	MC relays	MC relay 1	X10001 to X10256
		MC relay 2	X20001 to X20256
	MC coils	MC coil 1	Y10001 to Y10256
		MC coil 2	Y20001 to Y20256
	M code relays	M code relay 1	M10001 to M10096
		M code relay 2	M20001 to M20096
	MC control relays	MC control relay 1	P10001 to P10256
		MC control relay 2	P20001 to P20256
	MC control coils	MC control coil 1	Q10001 to Q10160
MC control coil 2		Q20001 to Q20160	

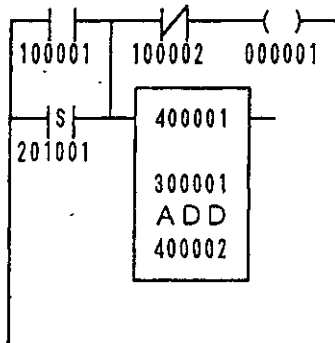
**Operating Principles**

2.2.1 What is a Reference? cont.

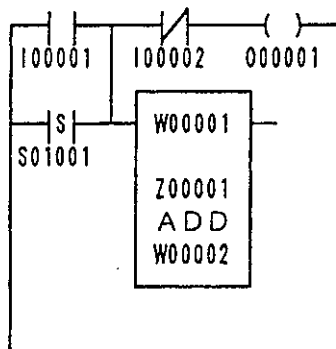
Type of Reference		Range of Reference for GL120 or GL130
Register references	Input registers	300001 to 300512 (Z00001 to Z00512)
	Holding registers (Registers allocated for output are called output registers.)	400001 to 409999 (W00001 to W09999)
	Constant registers	700001 to 704096 (K00001 to K04096)
	Link registers	R10001 to R11024 R20001 to R21024

The references enclosed by parentheses are used for alphanumeric input.

- 2) When numeric input is selected from the Environmental Settings Menu (under the MEMOSOFT Tool Menu), reference numbers will be represented either by six-digit numbers or by letters of the alphabet followed by five numeric digits. (See the reference numbers enclosed by parentheses in the table).
  
- 3) *Figure 2.9* and *Figure 2.10* show programming examples using numeric and alphanumeric input. The two programming examples are identical except for the type of input used.



**Figure 2.9 Program Example Using Numeric Input**



**Figure 2.10 Program Example Using Alphanumeric Input**

## 2.2.2 I/O References

- 1) References that can input and output data via the I/O Modules are called I/O references. They include input relays, output coils, input registers, and output registers.

### 2) Input Relays (1xxxxx)

- a) Input relays specify ON/OFF signals (digital signals) that are input via an Input Module. They can be regarded as image input coil contacts in the CPU Module. Relationships between input signals and input relay reference numbers are defined by the I/O allocation. For details, refer to chapter 3 *I/O Allocation*.

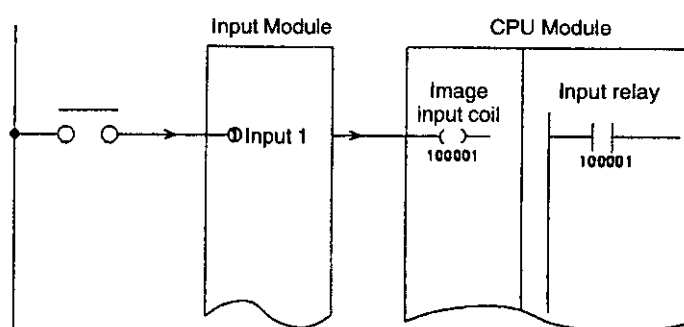


Figure 2.11 Input Signals and Input Relay References

- b) Input relays can be referenced as often as required, but their status cannot be changed. Except for those that are disabled or are not allocated, all input relays are turned OFF at startup.

### 3) Output Coils (0xxxxx)

- a) Output coils specify the ON/OFF signals that are output to external devices via a digital output module. Relationships between output signals and output coils are defined by the I/O allocation (of MEMOSOFT). The I/O allocation of MEMOSOFT defines the relationships between output signals and output coils.

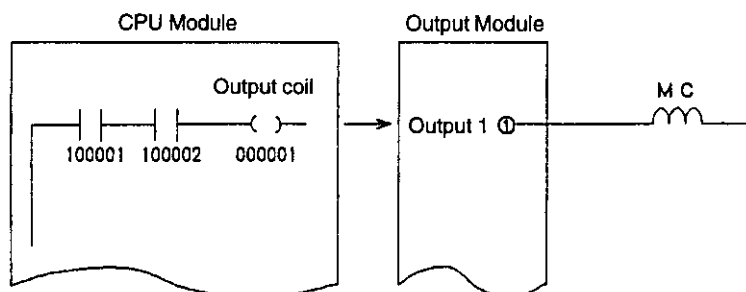


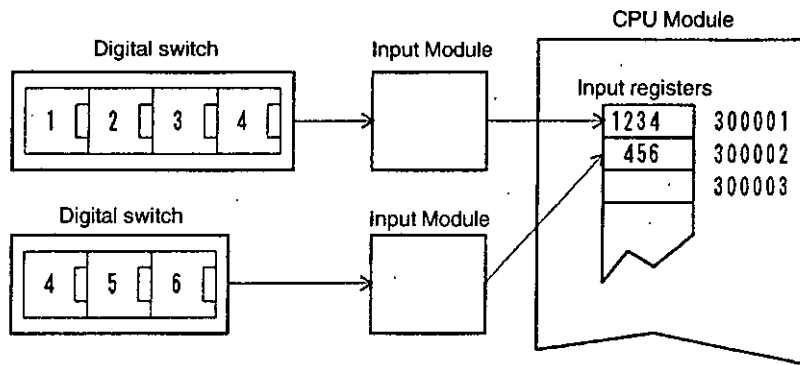
Figure 2.12 Output Signals and Output Coil References

- b) Coils that are used in ladder logic programs but which are not actual external outputs, auxiliary relays, are called "internal coils". For details of internal coils, refer to 2.2.3 *Internal References*. Coils not allocated for I/O can be used as internal coils.
- c) Output and input coils are all referred to as "coils." As well as being connected to relay contacts, coils can also be connected to outputs from timers, counters, arithmetical operations, data transfers, data conversions, matrices, and other instructions.
- d) Some coils with non-retentive memory turn off during the startup sequence, but latched coils with retentive memory keep their on/off status from just before the power was interrupted.
- e) An output reference number can only be used for one coil. Coil contacts can be used as often as required in any network.

**4) Input Registers (3xxxxx)**

- a) Input registers make up a temporary memory area (16 bits/register) for reading numeric signals sent from external devices such as digital switches, card readers, and A/D converters. Input registers are numbered individually in the form 3xxxxx.

The relationships between input signals and input registers are defined by the I/O allocation. For details, refer to Chapter 3 *I/O Allocation*.



**Figure 2.13 Input Register References and Input Devices**

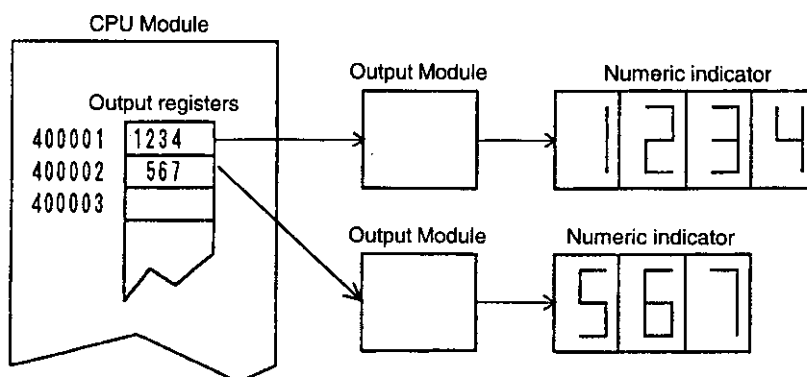
- b) The data input to a given input register is input in a 16-bit pattern. If the inputs from an external device are binary, then set the input data format for the I/O allocation to "BIN." If it is in BCD code, then set the format to "BCD." For details, refer to chapter 3 *I/O Allocation*.
- c) The contents of input registers are reset to "0" at startup.

**Note** (1) All of the input register data in the CPU Module is processed in binary.

- (2) Input register data can be referenced by ladder logic instructions, but it cannot be changed.

### 5) Output Registers (4xxxxx)

- a) Output registers make up a memory area (16 bits/register) for storing data to output numeric signals to external devices such as numeric indicators and D/A converters. The relationships between output signals and output register reference numbers are defined by the I/O allocation. For details, refer to Chapter 3 *I/O Allocation*.



**Figure 2.14 Output Register References and Output Devices**

- b) The data output from a given output register is output in a 16-bit pattern. If the external device uses binary data as input data, then set the output data format for the I/O allocation to "BIN." If it uses BCD code, then set the format to "BCD." For details, refer to Chapter 3 *I/O Allocation*.
- c) The contents of output registers are retained during power interruption.

**Note** All of the output register data in the CPU Module is processed in binary.

### Binary and BCD Data

In binary format, numeric data is expressed as either "1" or "0." The internal data for devices such as calculators is normally expressed in binary.

BCD is an acronym for Binary Coded Decimal. It uses four bits (i.e., four binary digits) to express a single decimal digit, and these are combined to express decimal numbers. Some devices connected to the PLC use BCD for I/O signals, and this data must be converted to binary for input to the GL120/GL130 and converted back to BCD for output to those devices.

The GL120/GL130 provides two different methods for converting data between binary and BCD: I/O allocation and ladder logic instructions. Conversion using I/O allocation is carried out by specifying either BIN (BCD to binary) or BCD (binary to BCD) for the I/O



data format. Conversion using ladder logic instructions is carried out by means of either the BIN (BCD to binary) or BCD (binary to BCD) instruction. *Figure 2.15* shows the binary and BCD formats.

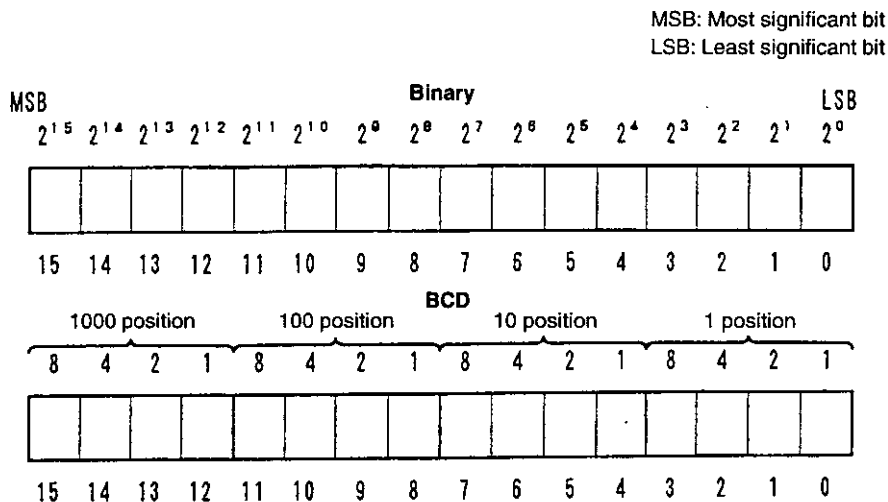


Figure 2.15 Binary and BCD Data Formats

Decimal	Binary	BCD
5	0000 0000 0000 0101	0000 0000 0000 0101
12	0000 0000 0000 1100	0000 0000 0001 0010
100	0000 0000 0110 0100	0000 0001 0000 0000
2,000	0000 0111 1101 0101	0010 0000 0000 0000

### 2.2.3 Internal References

- 1) References that are used only in the user program and for which the data is not output externally are called "internal references."
- 2) There are three kinds of internal references:
  - Internal coils
  - Holding registers
  - Constant registers
  - a) Internal Coils (0xxxxx)
    - (1) All coils are internal coils except output coils. All internal coils, except for disabled or latched coils, are turned off during the startup sequence.

- (2) A coil reference number can only be used for one coil, except for SBIT and RBIT instructions.

Coil contacts can be used as often as required in any network.

**Note** Coil 008192 is used as a battery monitor coil, so it cannot be programmed. Contacts for 008192, however, can be freely used as often as required.

Coil 008192 status	Meaning
ON	CPU Module's memory backup battery voltage output range is normal.
OFF	CPU Module's memory backup battery voltage output range is low.

b) Holding Registers (4xxxxx)

- (1) Holding registers make up a memory area (16 bits/register) that stores timer and counter set values, present values, results from math instructions, data for data transfers, data conversions and matrices, and the various constants required for operations.
- (2) Holding registers are assigned individual numbers in the form 4xxxxx. Their contents can be referenced by the user program, and they can also be changed. The contents are retained during power interruptions. The contents of holding registers specified as output registers can be output to external devices via an Output Module.
- (3) As shown in *Table 2.2*, some holding registers are used for calendars, stepping switches, constant sweeps, etc.

**Table 2.2 Holding Register Reference List**

Reference number	Function	Remarks
400001 to 402000	Normal holding registers	Registers allocated for output are called "output registers."
402001 to 402032	Stepping switch control registers	Can be used as normal holding registers if stepping switches are not used. (For details, refer to the following description.)
402033 to 409841	Normal holding registers	—
Allocate a normal holding register	Ladder modification flag register	Can be used for the CPU30 and the CPU35. Can be used as a normal holding register if the ladder modification flag register is not used. (For details, refer to <i>Ladder modification flag register</i> on page 2-23.)
409842 to 409914	MC link register table 1	Can be used as normal holding registers if a 4-axis MC20 Motion Module is not used. (For details, refer to <i>MC Link Register Table 1 below</i> and <i>MC Link Register Table 2</i> on page 2-19.
409915 to 409987	MC link register table 2	
409988 to 409995	Calendar registers	Refer to <i>Calendar Registers</i> on page 2-20.
409996	Timer register	Refer to <i>Timer Register</i> on page 2-21.
409997	High-speed scan time register	Can be used as normal holding registers if high-speed segment is not used. (For details, refer to <i>High-speed Scan Time Register</i> on page 2-21.)
409998	Constant sweep register	Refer to <i>Constant Sweep Registers</i> on page 2-22.
409999	Normal scan time register	

• Stepping Switch Control Registers (402001 to 402032)

These registers are used to control 32 stepping switches. The stepping switch numbers and their corresponding control register reference numbers are shown in *Table 2.3*. For details regarding stepping switches, refer to *2.2.5 Stepping Switch References*.

If stepping switches are not used, these registers can be used as normal holding registers.

**Table 2.3 Stepping Switch Control Registers**

Stepping Switch Number	Control Register
1	402001
2	402002
:	:
31	402031
32	402032

• MC Link Register Table 1 (409842 to 409914)

*Table 2.4* describes the registers used to handle data being transmitted between the CPU Module and the 4-axis MC20 Motion Module operating as Module no. 1. This table shows the applications of these registers.

For details regarding the data for each register, refer to MEMOCON GL120, GL130 Motion Module MC20 User's Manual for Programming (SIEZ-C825-20.52).

If MC20 Module no. 1 is not used, these registers can be used as normal holding registers.

**Table 2.4 MC Link Register Table 1 Applications**

Reference number	Application	Remarks
409842	Saves in memory the lower 16 bits of data for the MC20's axis 1 current position.	Input
409843	Saves in memory the upper 16 bits of data for the MC20's axis 1 current position.	Input
409844	Saves in memory the lower 16 bits of data for the MC20's axis 2 current position.	Input
409845	Saves in memory the upper 16 bits of data for the MC20's axis 2 current position.	Input
409846	Saves in memory the lower 16 bits of data for the MC20's axis 3 current position.	Input
409847	Saves in memory the upper 16 bits of data for the MC20's axis 3 current position.	Input
409848	Saves in memory the lower 16 bits of data for the MC20's axis 4 current position.	Input
409849	Saves in memory the upper 16 bits of data for the MC20's axis 4 current position.	Input
409850	Saves in memory the MC20's alarm code.	Input
409851	Register table for inputting data from the MC20.	Input
409852		
:		
409882		
409883	Register table for outputting data to the MC20.	Output
409884		
:		
409914		

• MC Link Register Table 2 (409915 to 409987)

*Table 2.5* describes the registers used to handle data being transmitted between the CPU Module and the 4-axis MC20 Motion Module operating as Module no. 2. This table shows the applications of these registers.

For details regarding the data for each register, refer to MEMOCON GL120, GL130 Motion Module MC20 User's Manual for Programming (SIEZ-C825-20.52).

If MC20 Module no. 2 not used, these registers can be used as normal holding registers.

Table 2.5 MC Link Register Table 2 Applications

Reference number	Application	Remarks
409915	Saves in memory the lower 16 bits of data for the MC20's axis 1 current position.	Input
409916	Saves in memory the upper 16 bits of data for the MC20's axis 1 current position.	Input
409917	Saves in memory the lower 16 bits of data for the MC20's axis 2 current position.	Input
409918	Saves in memory the upper 16 bits of data for the MC20's axis 2 current position.	Input
409919	Saves in memory the lower 16 bits of data for the MC20's axis 3 current position.	Input
409920	Saves in memory the upper 16 bits of data for the MC20's axis 3 current position.	Input
409921	Saves in memory the lower 16 bits of data for the MC20's axis 4 current position.	Input
409922	Saves in memory the upper 16 bits of data for the MC20's axis 4 current position.	Input
409923	Saves in memory the MC20's alarm code.	Input
409924	Register table for inputting data from the MC20.	Input
409925		
:		
409955		
409956		
409957	Register table for outputting data to the MC20.	Output
:		
409987		

• Calendar Registers (409988 to 409995)

This is a table of registers for use by a calendar. The CPU Module's internal calendar is read and set using this register table. The CPU Module checks the contents of holding register 409988 at the start of each scan.

When register 409988 is used to read the calendar, the calendar's time is read and then written to registers 409989 through 409995. When it is used to set the calendar, the time in registers 409989 through 409995 is set in the calendar.

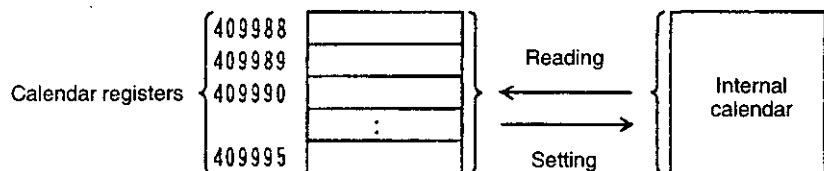
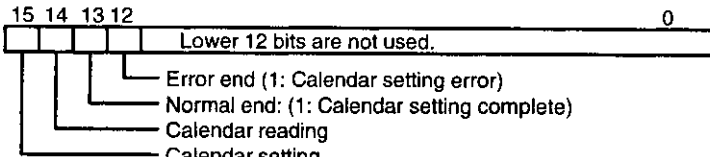


Figure 2.16 Relationship between Holding Registers and Calendar

Table 2.6 shows the calendar register specifications.

Table 2.6 Calendar Register Specifications

Reference number	Application	Specifications
409988	Calendar control	 <p><b>Reading</b> Set register 409989 to 4000 (hexadecimal). At the start of the next scan cycle, the contents of the calendar will be written to registers 409989 through 409995, and the contents of 409988 will be changed to 6000 (hexadecimal).</p> <p><b>Setting</b> Set the time in registers 409989 through 409995, and set register 409988 to 0. At the next scan, set register 409988 to 8000 (hexadecimal). After the calendar setting has been completed, 409988 will be set to 2000 (hexadecimal) at the next scan. If a value assigned to 409989 through 409995 is outside of the allowable range and a calendar setting error occurs, 409988 will be set to 1000 (hexadecimal).</p>
409989	Reading/Setting of day of week	1 = Sunday, 2 = Monday, 3 = Tuesday, 4 = Wednesday, 5 = Thursday, 6 = Friday, 7 = Saturday
409990	Reading/Setting of month	1 = January, 2 = February, 3 = March, 4 = April, etc.
409991	Reading/Setting of day	1 to 31
409992	Reading/Setting of year	Lower two digits, for example, 95 for 1995
409993	Reading/Setting of hour	0 to 23
409994	Reading/Setting of minute	0 to 59
409995	Reading/Setting of second	0 to 59

- Timer Register (409996)

The timer register measures time in units of 10 ms. The CPU Module increments this timer every 10 ms. The timer register can be referenced as a high-speed timer in ladder logic.

The timer range is 0 to 65,535 (i.e., 1 = 10 ms, 2 = 20 ms, ...65,535 = 655.35 s).

- High-speed Scan Time Register (409997)

When the high-speed scan is used, the high-speed scan time is set in this register from the CPU Module in units of 1 ms.

The high-speed scan is the actual time taken to solve the high-speed segment and service the high-speed I/O. For example, if the fixed high-speed scan time

(Ts) is 8 ms and the actual time (Tr) is 6 ms, then the value set in this register will be 6.

If high-speed scans are not used, this register can be used as a normal holding register.

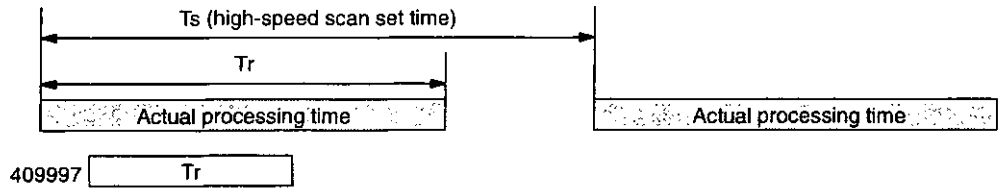


Figure 2.17 High-speed Scan Time

• Constant Sweep Registers (409998, 409999)

The constant sweep adjusts the scan time to a constant value.

When the constant sweep scan time is set using the MEMOSOFT, the set time is saved in register 409998. The set values range from 10 to 200 ms, in increments of 10 ms. (10 = 10 ms, 20 = 20 ms, etc.)

The CPU Module stores the actual scan time in register 409999 in units of 10 ms.

If the constant sweep function is not used, 409998 can be used as a normal holding register. Even when the constant sweep function is not used, the scan time is stored in 409999.

The actual scan time is the sum of the user program solving time, I/O servicing time, and overhead such as self-diagnosis.

If the actual scan time is shorter than the time set in 409998, the CPU Module will delay the next scan processing until the set time has been reached.

If the actual scan time is longer than the set value, the CPU Module will begin the next scan processing as soon as the current scan processing has been completed. In this case, the setting for the constant sweep scan time must be increased to achieve a constant time.

Figure 2.18 shows a timing chart for the constant sweep function.

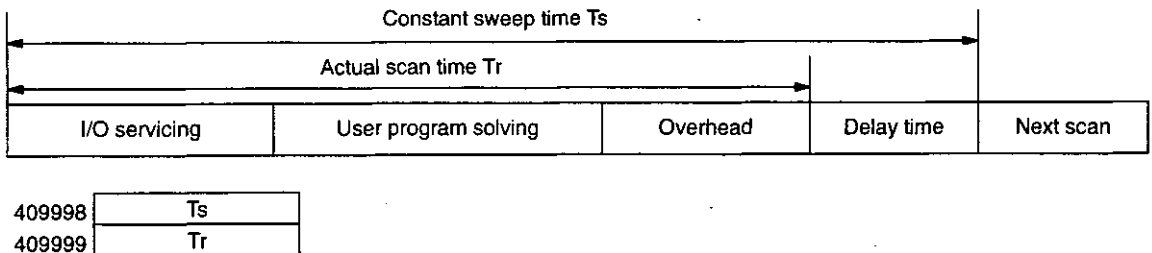


Figure 2.18 Constant Sweep Function

- Ladder modification flag register (a normal holding register)

The ladder modification flag register stores the event of modification in the ladder program of CPU30 and CPU35. This function is only available with the CPU30 and CPU35 modules.

When the ladder program in the CPU30 or the CPU35 is modified, 1 is assigned to the least significant bit of the ladder modification flag register.

With MEMOSOFT, allocate a normal holding register to the ladder modification flag register reference to validate the ladder modification flag register function.

If the ladder modification flag register is not used, do not allocate a holding register to the ladder modification flag register reference.

Use the programming panel or the ladder program to clear the contents of the ladder modification flag register.

The ladder modification flag register function is available with the following version No's of the CPU30 and the CPU35.

Name	Model Name	Model No.	Version No.	Where to find the version No.
CPU module (32 kW)	CPU30	DDSCR-130CPU54100	<input type="checkbox"/> <input type="checkbox"/> C05 and later	Nameplate on the right side of the module
CPU module (40 kW)	CPU35	DDSCR-130CPU54110	<input type="checkbox"/> <input type="checkbox"/> A06 and later	Nameplate on the right side of the module

#### c) Constant Registers (7xxxxx)

- (1) The constant registers (700001 to 704096) are references (16 bits/register) for storing the various constants essential for operations, such as system initial values, function tables, code charts, and so on.
- (2) Constant registers can be referenced within instructions, but they cannot be used as destinations for operation results.
- (3) The contents of constant registers are retained during power interruptions. Data settings for constant registers are carried out from the MEMOSOFT reference menu.

## 2.2.4 Link References

- 1) With the GL 120 or GL130, a PC Link Module (model no.: JAMSC-120NFB23100) can be used to create a PC Link system (see *Figure 2.19*) for exchanging data with all linked PLCs. The references for these PC Link communications are called "link references."



2) A maximum of two PC Link Modules can be mounted to either the GL120 or GL130, and there are link references for each of them.

3) There are two kinds of link references:

- **Link coils** for sending and receiving ON/OFF data
- **Link registers** for sending and receiving numeric data

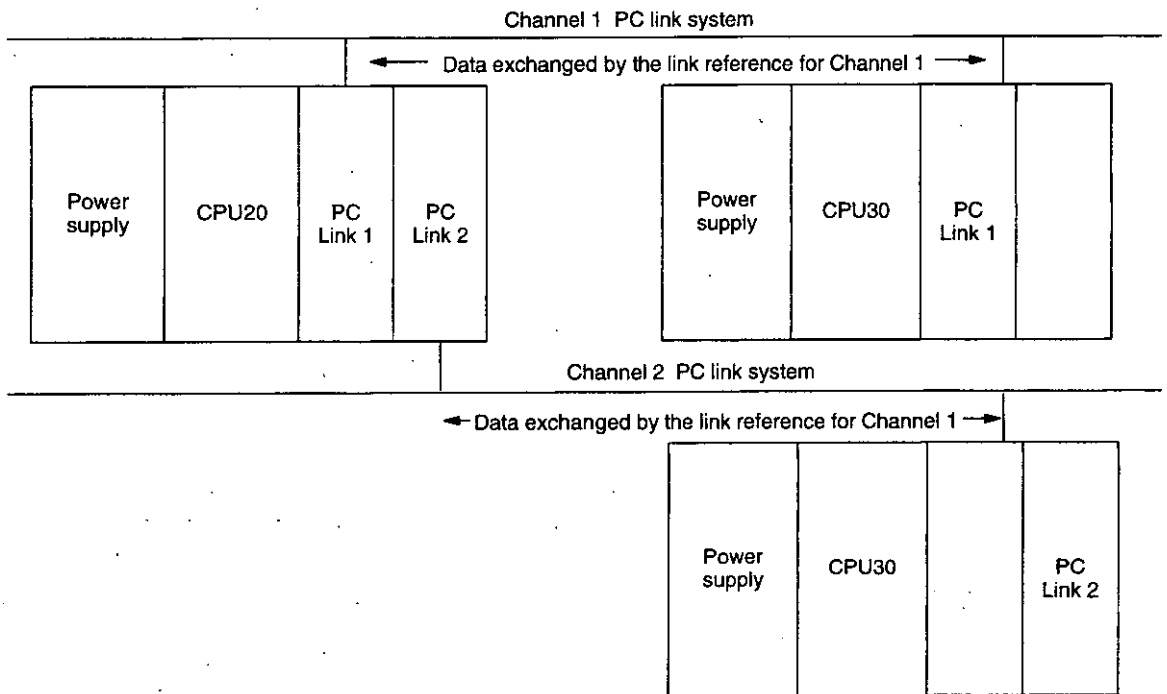


Figure 2.19 PC Link Communications System

**Note** The PLCs listed in the following table can be connected to a PC Link.

PLC	Usable PC Link Modules	Number of Modules Allowed
GL40S Series GL60S Series GL60H Series GL70H Series	JAMSC-IF64	1
GL120 Series GL130 Series	JAMSC-120NFB23100	2

a) Link Coils (Duxxxx)

(1) Link coils references Duxxxx are references for exchanging ON/OFF data with other PLCs on the PC link. The “u” represents the PC link channel number, so

references D10001 through D11024 are for Channel 1 and references D20001 through D21024 are for Channel 2.

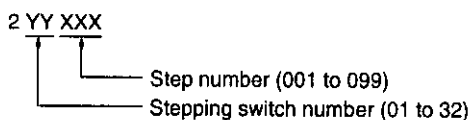
- (2) Each link coil can be specified for either sending or receiving data by assigning links in MEMOSOFT.  
The coil symbols of the link coils specified for sending ON/OFF data to other PLCs can be used in the ladder programs.
- (3) The contacts of the link coils specified for receiving ON/OFF data from other PLCs can be used in the ladder programs. All the link coils except for disabled or latched coils are turned off at startup.

#### b) Link Registers (Rxxxx)

- (1) Link registers are references for exchanging numeric data with other PLCs on the PC link. Link registers R10001 through R12048 are for Channel 1, and link registers R20001 through R22048 are for Channel 2.
- (2) Each link register can be specified for either sending or receiving data by assigning links in MEMOSOFT.
- (3) The contents of the link registers are retained during a power interruption.

## 2.2.5 Stepping Switch References

- 1) Stepping switch references consist of switch numbers and step numbers. There are 32 switch numbers with 99 step numbers each.



- 2) There is a holding register for controlling the stepping switch that corresponds to each switch number. The registers for stepping switches 1, 2, and 3 are 402001, 402002, 402003, and so on, up to 402032 for switch 32.
- 3) The number stored in each of these holding registers indicates the step number to be turned ON. For example, the switch number for stepping switch 205011 is "5" and the step number is "11." The holding register corresponding to switch no. 5 is "402005. Therefore, if the contents of 402005 is "11," then stepping switch 205011 will turn ON.
- 4) If stepping switches are not used, holding registers 402001 through 402032 can be used as normal holding registers.
- 5) For details regarding stepping switches, refer to the following manual:

- MEMOCON GL120, GL130 Software USER'S MANUAL (Vol. 2)  
(SIEZ-C825-20.12)

**Note** When stepping switches are used in a network, the NO and NC contacts must be used. Transitional contacts cannot be used.

## 2.2.6 Motion References

- 1) Motion references are used for transmitting ON/OFF data between the CPU Module and a 4-axis Motion Module (MC20).
- 2) The GL120 and GL 130 support up to two MC20s, and motion references are prepared for each of them. The module number is set by the I/O allocation of MEMOSOFT.
- 3) There are five kinds of motion references:
  - MC relays
  - MC coils
  - MC control relays
  - MC control coils
  - M code relays

For more details, refer to the following manual:

- MEMOCON GL120, GL130 MOTION MODULE MC20 USER'S MANUAL for Programming (SIEZ-C825-20.52)

a) MC Relays (Xuxxxx)

- (1) Table 2.7 describes how the MC relays, which are references, are used for inputting the values of MC20 output variables to the CPU Module .
- (2) MC relays are used in the form of contacts in ladder logic programs.
- (3) The "u" represents the MC20 Module number, so references X10001 through X10256 are for Module 1 and references X20001 through X20256 are for Module 2.

**Table 2.7 MC Relay Functions**

MC20 Module Nos.	MC Relay Reference Nos.	MC Relay Functions
1	X10001	Inputs ON/OFF data for MC20 output variable #O1.
	X10002	Inputs ON/OFF data for MC20 output variable #O2.
	:	:
	X10256	Inputs ON/OFF data for MC20 output variable #O256.
2	X20001	Inputs ON/OFF data for MC20 output variable #O1.
	X20002	Inputs ON/OFF data for MC20 output variable #O2.
	:	:
	X20256	Inputs ON/OFF data for MC20 output variable #O256.

## b) MC Coils (Yuxxxx)

- (1) MC coils are references used for sending ON/OFF data from the CPU Module to an MC20.
- (2) MC coils are used as coils in ladder logic programs.
- (3) The “u” represents the MC20 Module number, so references Y10001 through Y10256 are for Module 1 and references Y20001 through Y20256 are for Module 2.

Table 2.8 MC Coil Functions

MC20 Module Nos.	MC Coil Reference Nos.	MC Coil Functions
1	Y10001	Outputs ON/OFF data to MC20 input variable #I01.
	Y10002	Outputs ON/OFF data to MC20 input variable #I02.
	:	:
	Y10256	Outputs ON/OFF data to MC20 input variable #I256.
2	Y20001	Outputs ON/OFF data to MC20 output variable #O01.
	Y20002	Outputs ON/OFF data to MC20 output variable #O02.
	:	:
	Y20256	Outputs ON/OFF data to MC20 output variable #O256.

## c) MC Control Relays (Puxxxx)

- (1) MC control relays are references used for sending MC20 Module status signals to the CPU Module.
- (2) MC control relays are used in the form of contacts in ladder logic programs.
- (3) The “u” represents the MC20 Module number, so references P10001 through P10256 are for Module 1 and references P20001 through P20256 are for Module 2.

## d) MC Control Coils (Quxxxx)

- (1) MC control coils are references used for sending MFIN signals and overrides cannot be controlled by motion control instruction from the CPU Module to an MC20 Module.
- (2) MC control coils are used in the form of coils in ladder logic programs.
- (3) The “u” represents the MC20 Module number, so references Q10001 through Q10160 are for Module 1 and references Q20001 through Q20160 are for Module 2.

e) M Code Relays (Muxxxx)

- (1) M code relays are references used for inputting MC20 Module MFIN requests to the CPU Module.
  - (2) M code relays are used in the form of contact in ladder logic programs.
  - (3) The “u” represents the MC20 Module number, so references M10001 through M10096 are for Module 1 and references M20001 through M20096 are for Module 2.
- 4) To exchange numeric data between the CPU Module and an MC20, holding registers 409842 through 409914 and 409915 through 409987 are used. Refer to *2.2.3 Internal Reference 3) Holding Registers b) and c)*.

## 2.2.7 Reference Data and User Programs

- 1) Reference data is saved to memory called "state RAM" in the CPU Module. The relationship between reference data and the user program is shown in *Figure 2.20*. Left arrows in the figure indicate the writing of reference data, and right arrows indicate reading.

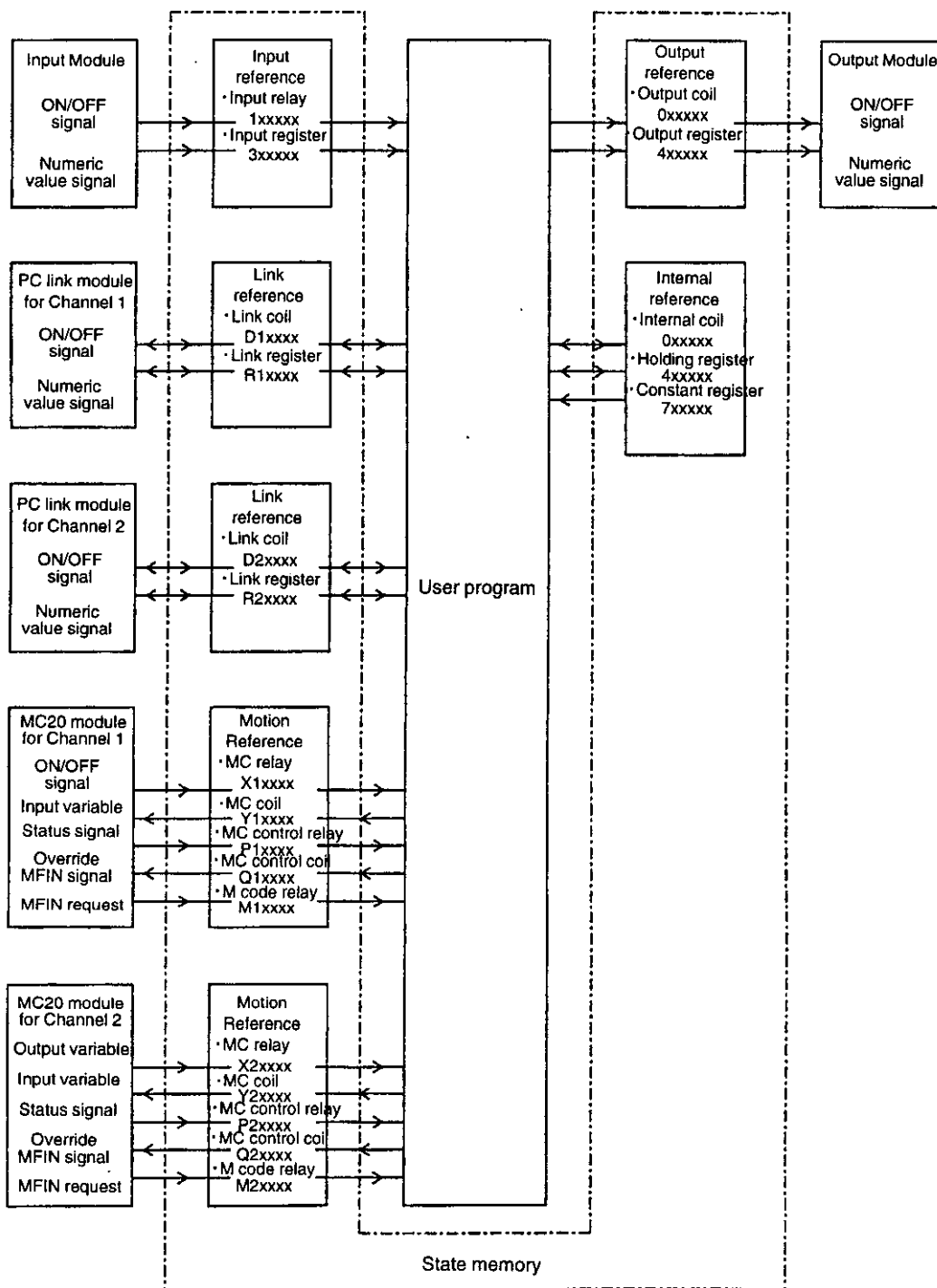


Figure 2.20 Reference Data and User Program

## 2.2.8 Changing Reference Numbers

- 1) The references shown in the *Table 2.9* can be assigned different reference numbers within the same reference group by means of the MEMOSOFT system configuration menu.
- 2) For example, the factory setting for the battery monitor coil is 008192, but it can be changed to another coil reference number. **It is strongly recommended, however, that the factory settings be retained as long as doing so would not cause any inconvenience.**

**Table 2.9 List of References that Can be Changed**

References	Factory Setting	Change Specifications
Battery monitor coil	008192	Another 0xxxxx
Stepping switch control registers	402001 to 402032	Another 4xxxxx
MC link register table 1	409842 to 409914	Another 4xxxxx
MC link register table 2	409915 to 409987	Another 4xxxxx
Calendar register	409988 to 409995	Another 4xxxxx
Timer register	409996	Another 4xxxxx
High-speed scan time register	409997	Another 4xxxxx
Constant sweep register	409998	Another 4xxxxx to (4xxxxx+1)
Normal scan time register	409999	

## 2.3 Internal Processing

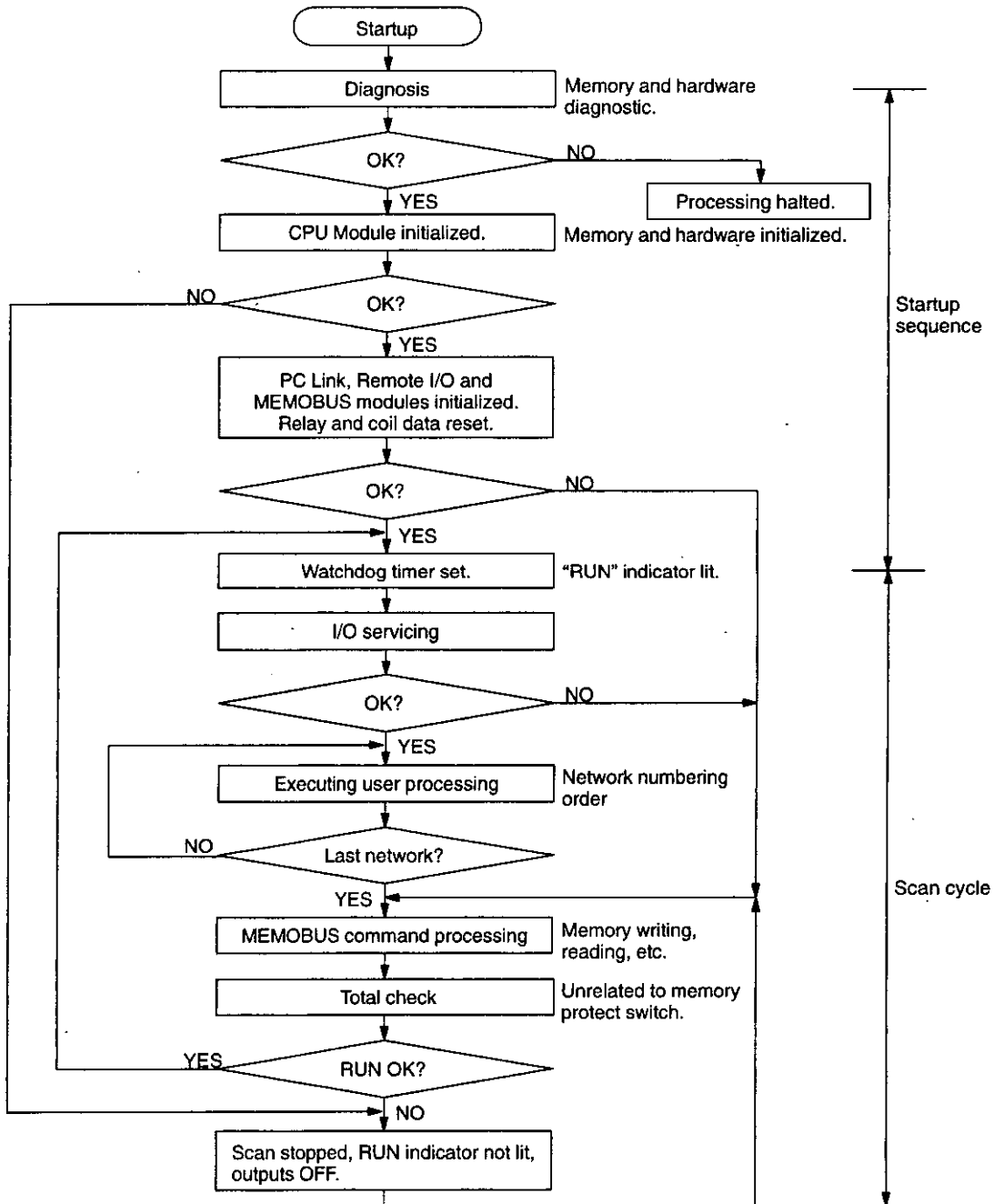
■ This section explains the internal processing of the GL120 and GL130.

2.3.1	Internal Processing Flow .....	2-32
2.3.2	Startup Sequence .....	2-33
2.3.3	Scanning Cycle .....	2-33
2.3.4	Watchdog Timer .....	2-35
2.3.5	Network Numbers .....	2-35
2.3.6	Total Check .....	2-36



### 2.3.1 Internal Processing Flow

The following flowchart illustrates GL120 and GL130 internal processing.



## 2.3.2 Startup Sequence

- 1) Memory contents are checked first when the power is turned ON. If they are okay, then the relays, coils, and so on, are all initialized. The status of the relays and coils after initialization is shown in *Table 2.10*.

**Table 2.10 Initialization at Startup**

	Elements	References	Status at Startup
Digital References	Coils	0xxxxx	All OFF except for latched or disabled coils.
	Link coils	Duxxxx	
	MC coils	Yuxxxx	
	MC control coils	Quxxxx	
	Input relays	1xxxxx	All OFF except for disabled relays.
	MC relays	Xuxxxx	
	MC control relays	Puxxxx	
	M code relays	Muxxxx	
	Latched coils Disabled relays Disabled coils	Digital references above	Retain status from just before power interruption.
Register References	Input registers	3xxxxx	All are reset to "0."
	Holding registers	4xxxxx	Retain contents from just before power interruption.
	Constant registers	7xxxxx	
	Link registers	Ruxxxx	

u: 1 or 2

- 2) The time required for the startup sequence depends on the system configuration, but it is approximately five seconds.
- 3) With the GL120 and GL130, the RUN status before power was interrupted is saved in memory. If there are no errors detected in the diagnosis at startup, and if the status immediately before the power interruption was RUN, then it will be regarded as "RUN OK" and the RUN scanning cycle will begin. If the status is other than "RUN OK," the CPU Module will go to STOP status and only MEMOBUS commands from the communications port will be processed.

**Note** Even if the status immediately before the power interruption was STOP (e.g., from the Programming Panel), it will still be regarded as "RUN OK" and the RUN scanning cycle will begin if there are no errors detected in the diagnosis at startup and if the DIP switch in the CPU Module is set for the automatic RUN operating mode.

## 2.3.3 Scanning Cycle

- 1) When the startup sequence has been completed, the scanning cycle begins and the I/O servicing is executed in order, first output and then input.
- 2) Solving the user program begins with network 1 of normal segment 1, and continues in order through networks 2, 3, and so on. If there is more than one normal segment, then,

as soon as the last network in normal segment 1 has been solved, solving the program continues in order beginning with network 1 of normal segment 2. (Refer to Figure 2.21.)

- 3) After the user program has been solved, then MEMOBUS commands received from the MEMOBUS and MEMOBUS PLUS ports are processed.
- 4) The total check is executed last, and processing from that scan is completed. Processing is repeated as shown below.

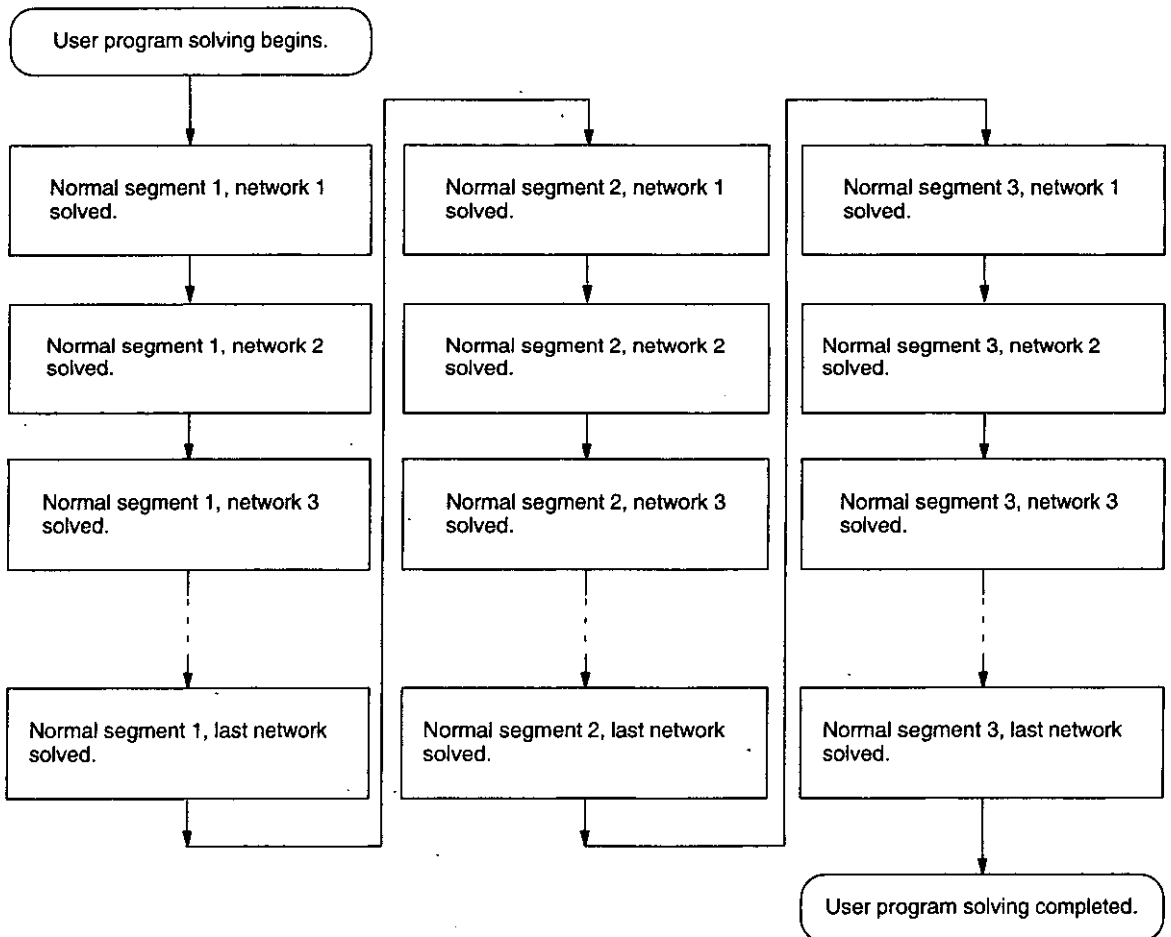


Figure 2.21 Solving the User Program

**Note** The GL120 and GL130 have high-speed scan and segment scheduling functions. When these functions are used, the entire program may not be solved in a single scan. For details regarding the high-speed scan and segment scheduler, refer to section 2.5 *Segment Scheduler*.

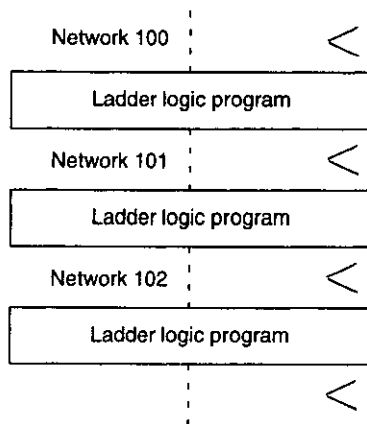
### 2.3.4 Watchdog Timer

The watchdog timer is set at the beginning of each scanning cycle, and the timer remains ON for approximately 240 ms after it starts. If the timer is not preset within 240 ms, a scan error will occur and the scan will be stopped. The CPU Module's RUN indicator light will go out and all outputs will be turned OFF.

### 2.3.5 Network Numbers

The user program is stored in the program memory in units of networks. The networks are numbered consecutively both within each segment and within the user program as a whole. (For details, refer to *Figure 2.4* in subsection 2.1.2 *Networks*.)

As shown in *Figure 2.22*, the CPU Module automatically takes the network numbers in order by specifying network delimiters.



The programmer sets the delimiters (represented by "<" in the figure). This is achieved in MEMOSOFT operation by pressing the ALT-A Keys.

**Figure 2.22 Network Numbers and Ladder Logic**

- Note**
- (1) Within the capacity of the user program, any number of network numbers can be created.
  - (2) One word of memory is used each time a network number is created.

## 2.3.6 Total Check

- 1) In the unlikely event that memory contents such as the user program or important system constants were changed during operation, serious damage could result if the change was not detected. In order to prevent this kind of situation from occurring, a self-diagnostic function called the "total check" is provided.
- 2) With this function, the sum total of the original unchanged memory area contents is obtained, and it is then compared to the sum total of the memory contents with each scan. If there is any discrepancy, it is regarded as an error and the scan is stopped.

**Note** When an error is generated in this way, all outputs (Counter Module notch outputs and all outputs connected to the Digital Output Modules) are either turned ON or OFF according to the timeout output designation. The factory setting is for the outputs to turn OFF.

- 3) The following memory areas are the ones checked by the "total check" function.
  - User program
  - The various allocation tables



The total check is carried out regardless of the position of the memory protect switch. If it is OFF, the program can be changed by means of the Programming Panel or other programming device without an error being generated. Each time it is changed, the new sum will be taken as the criteria for the check.

## 2.4 Scanning

In addition to the normal scan, the GL120 and GL130 also have a high-speed scan and a segment scheduler. The high-speed scan is used to solve the high-speed segment to service high-speed I/O at a fixed interval. The segment scheduler is for either solving or not solving segments depending on specified conditions. High-speed scans and the segment scheduler are explained in more detail in section 2.5 *Segment Scheduler*. This section explains basic scanning concepts, for solving ladder logic programs in normal segments.

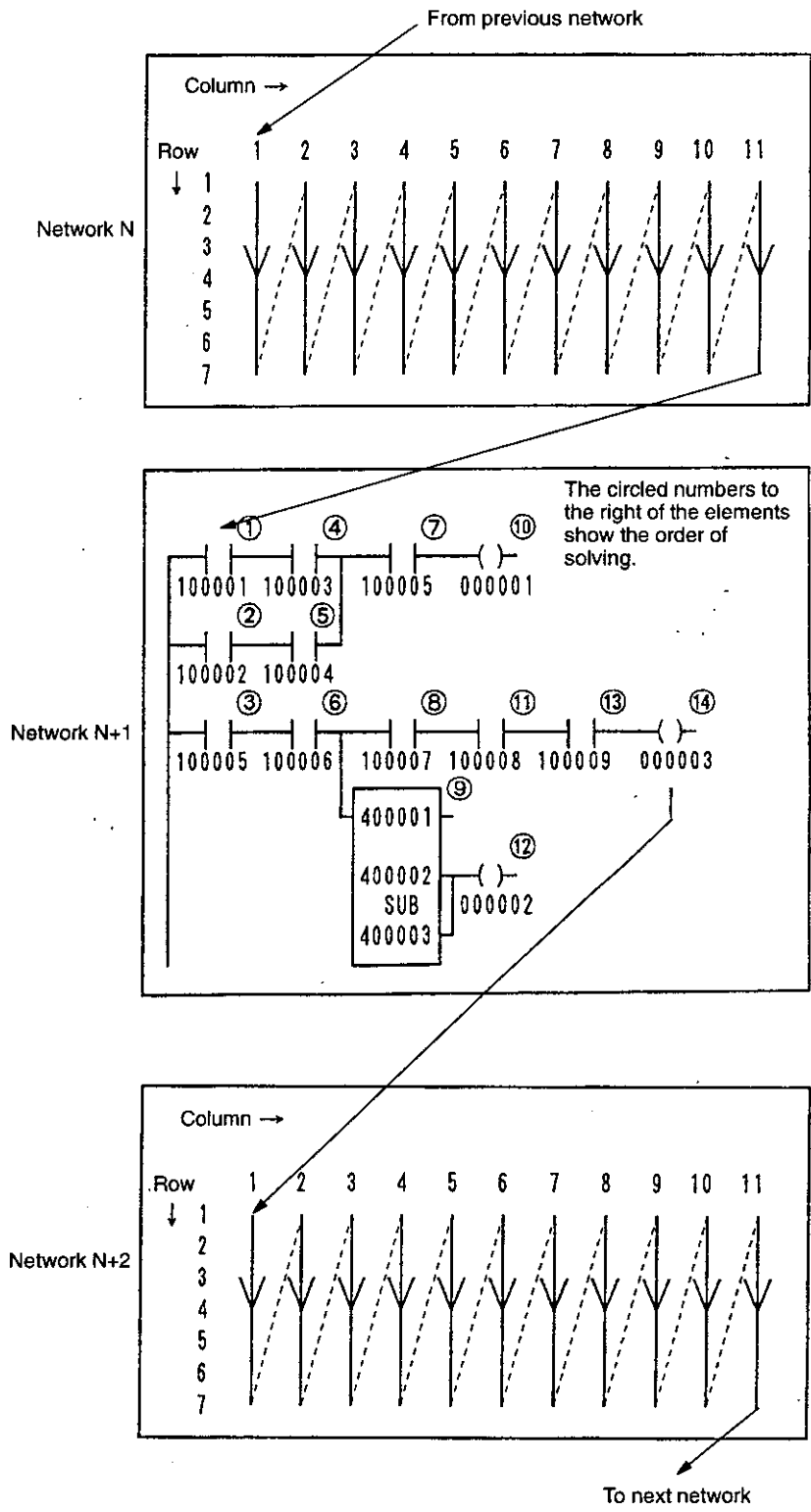
2.4.1	Ladder Logic Program Solving and Scanning .....	2-37
2.4.2	Scan Time .....	2-42
2.4.3	Sweep Functions .....	2-44

### 2.4.1 Ladder Logic Program Solving and Scanning

- 1) The ladder logic program stored in the user program memory is solved in order of networks.
- 2) Within a given network, the columns are solved in order from left to right. The elements in each column are solved in order from the top row to the bottom. In a single scan, the networks in a given segment are solved in order from network 1 through the last network. The order in which networks are solved is shown in *Figure 2.23*.
- 3) Nodes without elements are not solved.
- 4) Application instructions extending over multiple rows are solved as a single unit rather than row by row.

**Operating Principles**

**2.4.1 Ladder Logic Program Solving and Scanning cont.**



**Figure 2.23 Order of Network Solving**

- 5) In considering the operation of the ladder logic program, it is necessary to take into account the particular characteristics of the scan method. *Table 2.11* shows the status of elements during the scan. The status of the elements is stored in state RAM.

**Table 2.11 Status of Elements During Scan**

Elements	Status
Enabled relays <ul style="list-style-type: none"> <li>• Input relays (1xxxxx)</li> <li>• MC relays (X1xxxx) (X2xxxx)</li> <li>• MC control relays (P1xxxx) (P2xxxx)</li> </ul>	1) These relays are turned OFF at the startup sequence. 2) Status is refreshed at first I/O servicing after completion of startup sequence. 3) Input status is then refreshed at the I/O servicing timing with every scan. That status will not be changed until the same timing in the next scan. 4) When the high-speed scan is used, the input status for high-speed I/O is refreshed in synchronicity with the high-speed scan. Subsequent operations are the same.
Input registers (3xxxxx)	Input registers are reset to "0" at the startup sequence. Subsequent operations are the same as with input relays.
Enabled normal coils <ul style="list-style-type: none"> <li>• Coils (0xxxxx)</li> <li>• Link coils (D1xxxx) (D2xxxx)</li> <li>• MC coils (Y1xxxx) (Y2xxxx)</li> <li>• MC control coils (Q1xxxx) (Q2xxxx)</li> </ul>	1) These coils are all turned OFF at the startup sequence. 2) When coils are solved in the first scan, they turn ON and OFF according to those results. A coil's status does not change until that coil is solved in the next scan. Subsequent operations are the same as above.
Enabled latch coils <ul style="list-style-type: none"> <li>• Coils (0xxxxx)</li> <li>• Link coils (D1xxxx) (D2xxxx)</li> <li>• MC coils (Y1xxxx) (Y2xxxx)</li> <li>• MC control coils (Q1xxxx) (Q2xxxx)</li> </ul>	1) Solving begins with the status from just before the power interruption. 2) When coils are solved in the first scan, they turn ON and OFF according to those results. A coil's status does not change until that coil is solved in the next scan. Subsequent operations are the same as above.
Stepping switches (2Yxxxx)	Solving begins with the status from just before the power interruption.
Disabled relays and coils	Solving begins with the status from just before the power interruption. The status is refreshed either from the Programming Panel (i.e., a forced ON/OFF operation) or at the end of the scan.
<ul style="list-style-type: none"> <li>• Holding registers (4xxxxx)</li> <li>• Link registers (R1xxxx) (R2xxxx)</li> </ul>	Solving begins with the status from just before the power interruption. When the program is scanned, the status is refreshed each time the contents of a register is changed by solving the ladder program.

- 6) I/O servicing between ladder programs and modules is performed via I/O references, link references, and motion references in the status memory. *Figure 2.24* shows how I/O



servicing is performed between the status memory and a modules at the beginning of the scan.

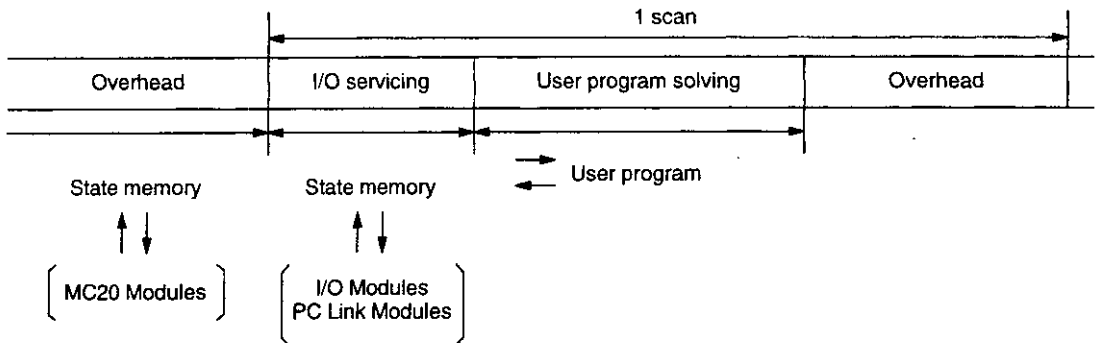
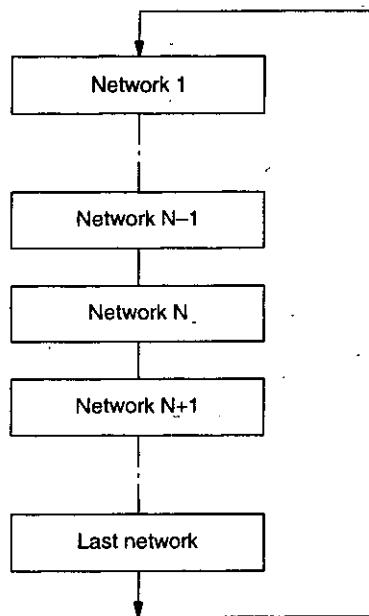


Figure 2.24 I/O Servicing and User Program

- 7) The ON/OFF status of elements (i.e., coils, link coils, MC coils and MC control coils) that are changed by solving the ladder logic instructions, as well as the contents of holding registers and link registers, receive new results when instructions are solved. Not only can those results be used immediately by subsequent networks, but they can even be reflected in subsequent elements within the same network.



The status of coils solved in network N is not changed from network N+1 through the last network, and then from network 1 through network N-1, for the period of one scan. In contrast, the contents of holding registers and link registers can be changed a number of times within a single scan. Any change is retained until there is a further change.

Figure 2.25 Order of Network Solving Within a Segment



When the SET BIT (SBIT) and RESET BIT (RBIT) instructions are used, the same coil can be turned ON and OFF multiple times within a single scan.

- 8) If the order of ladder logic program solving shown in *Figure 2.26* were written as reference numbers, it would appear as follows:  
 100001 → 100003 → 100002 → 100042 → 000101 (coil) → 000101 (contact) → 000050 (coil)

Any change that occurs in coil 000101 is reflected in the circuit in the second row immediately (in the same scan).

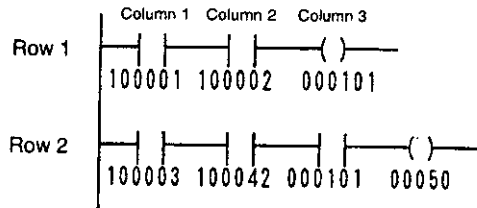


Figure 2.26 Order of Solving Within a Network

- 9) If another element is inserted between contact 100002 and coil 000101 in the first row of a network (as shown in the top diagram in *Figure 2.27*), the results of coil 000101 changes will not be reflected in the second row within that scan, but will be delayed until the next scan.

If this delay will present a problem, it can be avoided by deleting the second row from that network and writing it to the next network (as shown in the two lower diagrams in *Figure 2.27*). When it is done this way, the results will be reflected in the same scan.

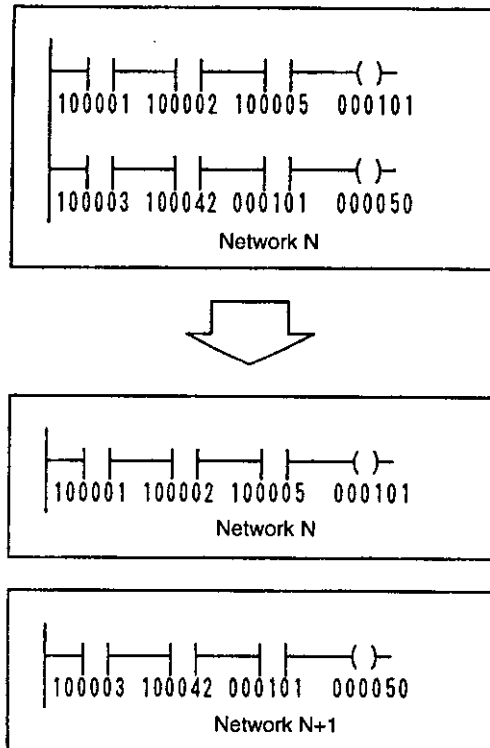


Figure 2.27 Separation into Two Networks

## 2.4.2 Scan Time

This section explains scan times for user programs with normal segments only, i.e., when the high-speed scan function is not used. Scan times when the high-speed scan is used are explained in section 2.5 *Segment Scheduler*.

### 1. Local I/O System Scan Times

- 1) An estimated scan time T (ms) for a local I/O system can be calculated by means of the following equation:

$$T = (\text{Basic time}) + (\text{Additional time}) + (\text{User program solving time}) + (\text{I/O servicing time})$$

#### A. Basic Time

The basic time is the time required for the self diagnosis that is carried out with every scan.

#### B. Additional Time

- a) The additional time is the time required for MEMOBUS command processing.
- b) MEMOBUS commands are communications commands sent from devices that work with MEMOBUS (e.g., Programming Panel, computers, etc.).
- c) There are several kinds of MEMOBUS commands: User program read commands, write commands, search commands, commands for reading and writing the status of coils, relays, register data, and so on.
- d) The additional time is the processing time that is required when there are communications between devices due to MEMOBUS commands. This time will be "0" if no devices such as a Programming Panel or computer are connected. The time is determined by the type and contents of MEMOBUS commands that are received.

#### C. User Program Solving Time

The user program solving time is the time required to solve the user program from the first network of normal segment 1 to the last network of the last segment. This time is determined by the types of instructions used in each network, and the number of instructions that are used. Refer to the appendices for instruction processing time.

Instruction processing times differ depending on whether the instruction is solved or not solved, the scan time normally fluctuates.

#### D. I/O Servicing Time

- a) I/O servicing time is the time required for inputting and outputting data between the CPU Module and the various other Modules.
- b) The Modules that require I/O servicing time are I/O Modules and Option Modules (Communications Modules, 4-axis Motion Modules, etc.). The time is determined by the type and number of Modules that are used.

2) The respective items are as follows:

- a) Basic time = 2 to 3 ms
- b) Additional time = 0.12 ms/MEMOBUS port
- c) User program solving time =  $(2 \mu\text{s} \times \text{Number of networks} + \Sigma \text{Instruction processing time})/1000$  (ms)
- d) I/O servicing time = (I/O Module I/O servicing time) + (Option Module I/O servicing time)
  - (1) If there is no I/O Module allocation, and if no Option Module is used, the I/O servicing time will be "0."
  - (2) I/O Module I/O servicing time = (Input overhead) + (Output overhead) + ( $\Sigma$  Input Module time) + ( $\Sigma$  Input byte time) + ( $\Sigma$  Output Module time) + ( $\Sigma$  Output byte time)

Item	GL120 ( $\mu\text{s}$ )	GL130 ( $\mu\text{s}$ )
Input overhead	67	100
Output overhead	56	91
Per Input Module	83 (96)	72 (88)
Per input byte	20 (24)	15 (19)
Per Output Module	91 (105)	75 (91)
Per output byte	10 (14)	8 (12)

**Note** The numbers inside the parentheses are the values when mounted to racks 2 through 4.

- (3) With the GL120 or the GL130, the I/O Module I/O servicing time when two 24-VDC 32-point Modules are used for both input and output on Rack 1 is as follows:

$$\text{I/O Module I/O servicing time} = 67 + 56 + 83 \times 2 + 20 \times 8 + 91 \times 2 + 10 \times 8 = 711 \text{ } (\mu\text{s})$$

- (4) The Option Module I/O servicing times are shown in *Table 2.12* below.

**Table 2.12 Option Module I/O Servicing Times**

Name	Model Name	Model No.	GL120 I/O Servicing Time ( $\mu\text{s}$ )	GL130 I/O Servicing Time ( $\mu\text{s}$ )
Coaxial Remote I/O Driver Module	RIOD-COAX	JAMSC-120CRD13100	363	310
RS-232C MEMOBUS Module	MEMOBUS-232	JAMSC-120NOM26100	2900 $\pm$ 300	3000 $\pm$ 330
RS-422 MEMOBUS Module	MEMOBUS-422	JAMSC-120NOM27100	2900 $\pm$ 300	3000 $\pm$ 330
PC Link Module*1	PCLINK-COAX	JAMSC-120NFB23100	7400 $\pm$ 1800 $+\alpha$	3000 $\pm$ 1000 $+\beta$
4-axis Motion Module	MC20	JAMSC-120MMB10400	1870	1497

\*1: With PC Link Modules, the following link data processing times are added.

Item	GL120 Data Processing Time (μs)	GL130 Data Processing Time (μs)
Link input processing time	$3,650 + 0.4 \times (\text{number of input words})$	$3,500 + 0.23 \times (\text{number of input words})$
Link output processing time	$4,260 + 1.42 \times (\text{number of output words})$	$140 + 1.95 \times (\text{number of output words})$
Link data processing time	$\alpha = \text{Inputs} + \text{outputs}$	$\beta = \text{Inputs} + \text{outputs}$

**IMPORTANT**

Instruction processing times differ greatly depending on whether instructions are solved or not solved. In addition, the basic time and I/O servicing time fluctuate, so the scan time T normally fluctuates.

**2. Processing Time When Remote I/O is Used**

When remote I/O is connected, the scan time will be the longer of the following: the remote I/O processing time or the single scan time T.

Figure 2.28 shows an example when the single scan time T is longer. Refer to the MEMOCON GL120, GL130 Coaxial Remote I/O System User's Manual (SIEZ-C825-70.8) for details.

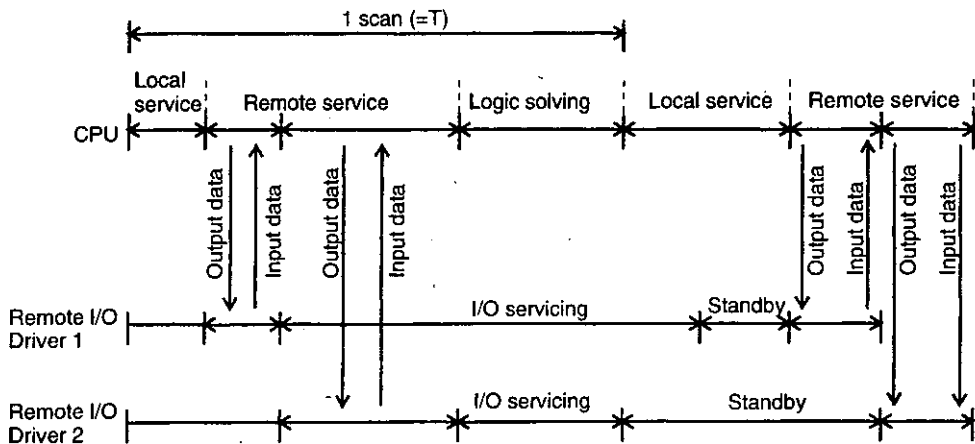


Figure 2.28 When Remote I/O is Shorter

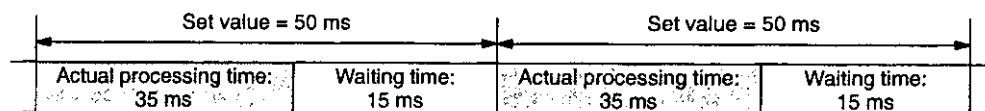
**2.4.3 Sweep Functions**

The GL120 and GL130 have two sweep functions: constant sweep (which regulates the scan time) and single sweep (which runs the user program for one scan only).

**1. Constant Sweep**

- 1) The constant sweep function causes the scan to be executed at fixed intervals, up to a maximum of 200 ms (in units of 10 ms). It is used for regulating the scan time.

- 2) The constant sweep function is set using the MEMOSOFT.
  - a) For offline operations, select the constant sweep (C sweep) and set the scan time using the segment scheduler menu under the system configuration menu.
  - b) For online operations, select the constant sweep (C sweep) under the sweep menu under the PLC operations (PLC Ops) menu.
- 3) The scan time set from the MEMOSOFT is stored in holding register 409998.
- 4) When the actual scan time is shorter than the set value, the CPU Module regulates the time by waiting for a period equal to the difference between them.
  - a) For example, *Figure 2.29* shows a case where the constant sweep set value is 50 ms, whereas the actual scan time is only 35 ms. The CPU Module brings the time up to 50 ms by waiting for 15 ms.
  - b) The actual processing time is stored in holding register 409999 in units of 10 ms. Thus in the case illustrated in *Figure 2.29*, "30" would be stored there.
- 5) If the actual processing time is longer than the set value the constant sweep (C sweep), the CPU Module begins the next scan immediately without waiting. Thus if the contents of 409999 are greater than those of 409998, the constant sweep function will not operate. Make the set value larger than the actual scan time to operate the constant sweep function.



**Figure 2.29 Constant Sweep Function**

## 2. Single Sweep

- 1) The single sweep function carries out I/O servicing and user program solving for one scan only, and it is useful for checking user program operation.
- 2) This function is enabled only when the CPU Module is in RUN mode.
- 3) The single sweep function is started up by means of the PLC operation menu under MEMOSOFT online operations. The CPU Module stops user program solving when either single sweep or high-speed single sweep is selected from the PLC operation's sweep menu.
- 4) When the single sweep function is started up, the CPU Module executes a single scan. *Figure 2.30* shows the changes in the contents of a register when a single sweep is carried out.

- 5) With single sweep, the user programming in normal segments is executed; with high-speed single sweep, the high-speed segment is executed.
- 6) There may be cases when the timer circuits do not time properly due to the combination of the minimum scan time set value (200 ms max., in units of 10 ms) and the timing unit of the timer.
- 7) Leaving the single sweep menu returns the CPU Module to normal scan processing.

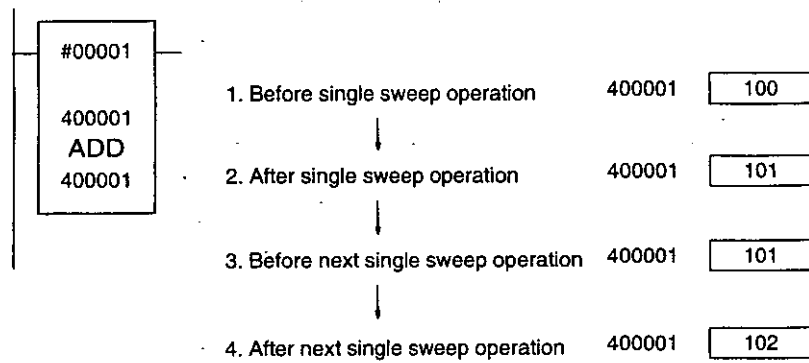
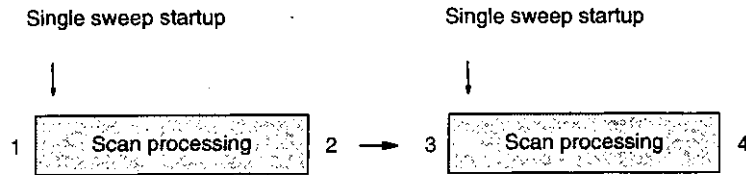


Figure 2.30 Single Sweep Operation Example

## 2.5 Segment Scheduler

▣ This section explains the segment scheduler based on the high-speed scan.

2.5.1	High-speed Scan .....	2-47
2.5.2	The Segment Scheduler .....	2-53

### 2.5.1 High-speed Scan

#### 1. High-speed Scan

The high-speed scan is used for high-speed segment execution and high-speed I/O servicing at fixed intervals within a range of 4 to 100 ms. As shown in *Figure 2.31*, the high-speed scan and the normal scan can be run in parallel.

The high-speed scan is processed in time  $T_{HS}$ .  $T_{HS}$  consists of the actual processing (hereafter referred to as “high-speed processing”) time,  $T_{HR}$ , and the remaining time,  $T_R$ . The CPU Module uses this remaining time for normal scan processing.

As described in the previous section, normal scans consist of normal segment execution, I/O servicing, and overhead such as self-diagnosis.



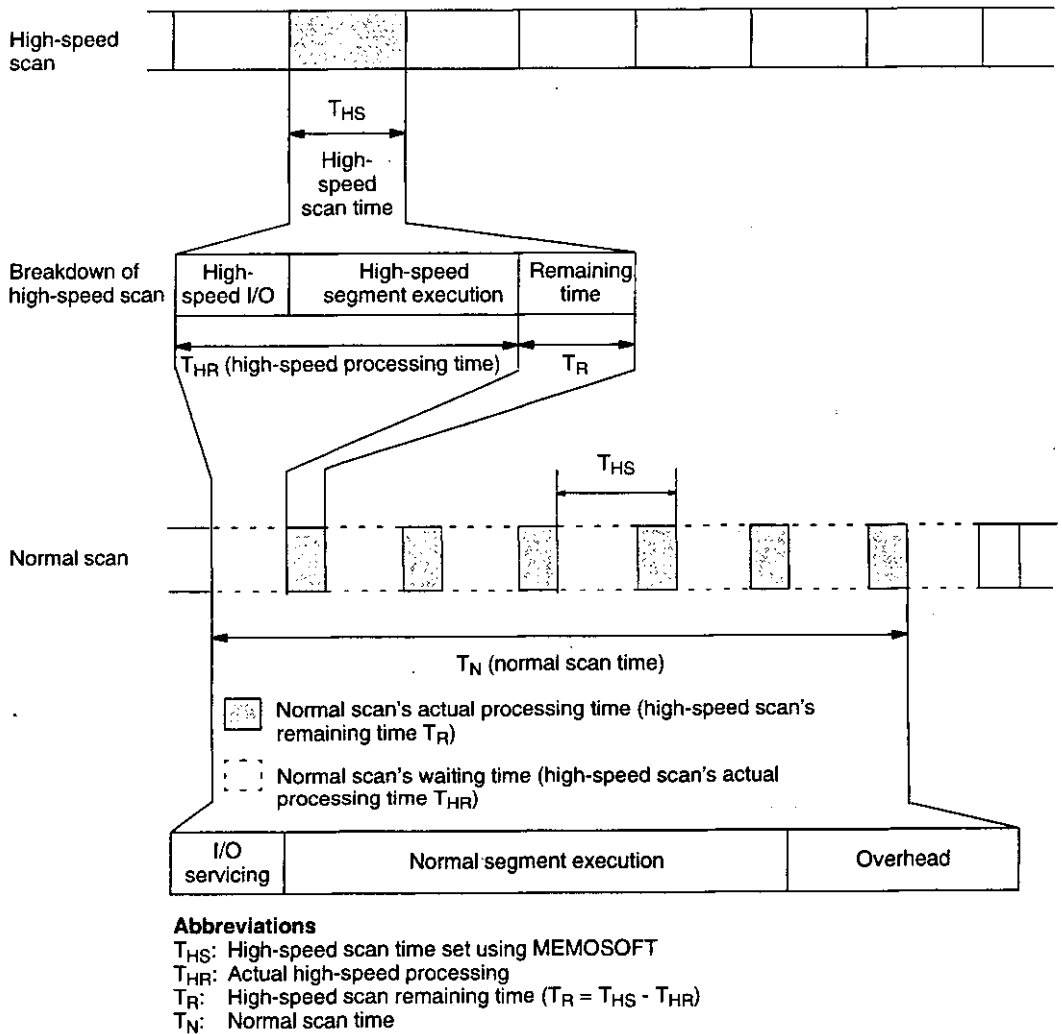


Figure 2.31 Relationship between High-speed Scans and the Normal Scan

## 2. High-speed Scan Processing

The items processed during the high-speed scan are high-speed segment solving and high-speed I/O servicing.

### A. High-speed Segment Execution

The high-speed segment is indicated by "H" on the MEMOSOFT segment status display. Like normal segments, it consists of networks, and it is executed in order of network numbers. The order of execution within networks is also the same as for normal segments.

If there is a JUMP SUBROUTINE (JSR) instruction in a network, the appropriate subroutine program in the subroutine segment will be executed. For details regarding subroutines, refer to 2.1.3 Subroutines.

The relationship between the user program and the high-speed segment is shown in Figure 2.32.

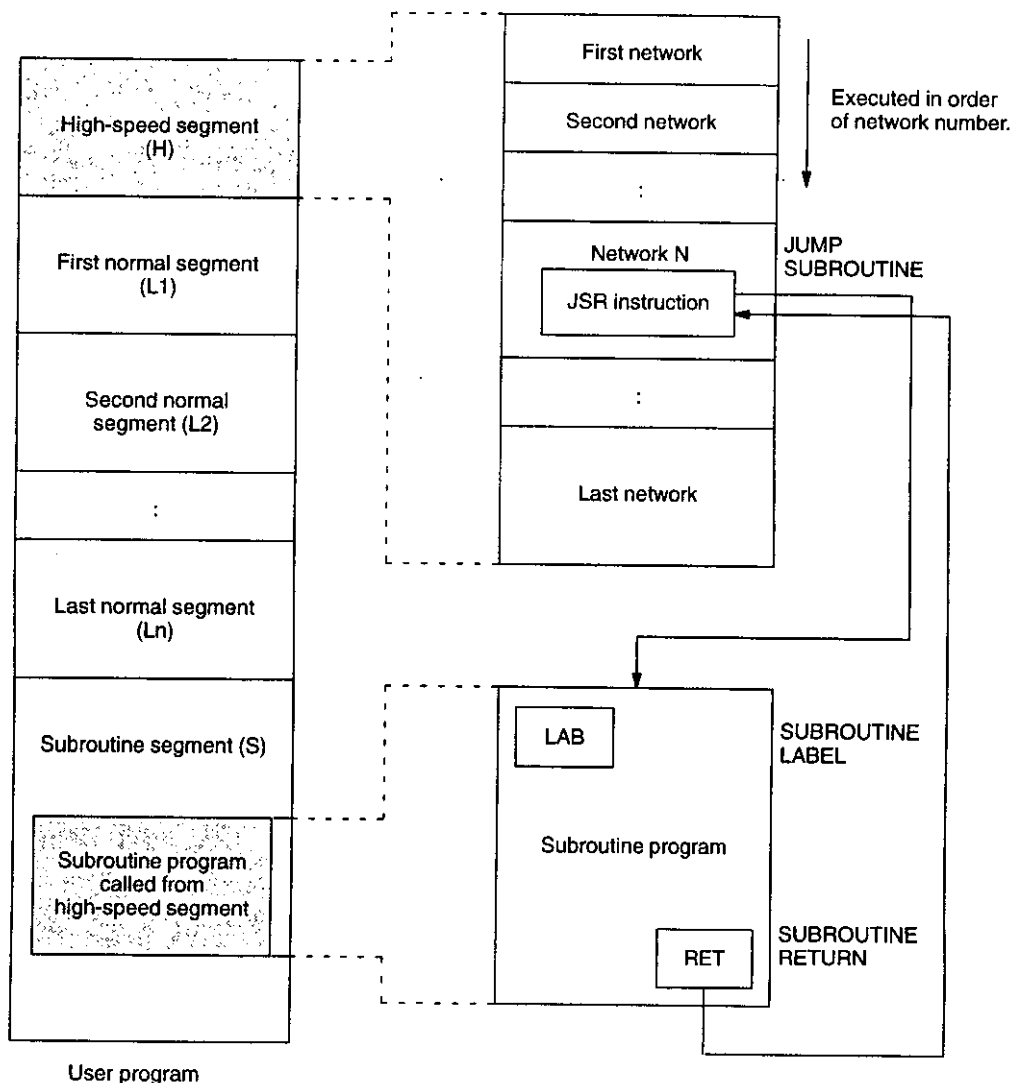


Figure 2.32 Relationship Between User Program and High-speed Segment

### B. High-speed I/O Servicing and Motion Module Servicing

High-speed I/O servicing is I/O servicing for Modules allocated as "high speed" for scan servicing during I/O allocation. Output servicing is executed first, followed by input servicing.

For local I/O, either high-speed servicing or normal servicing can be specified with respect to each Module. For remote I/O, either high-speed servicing or normal servicing can be specified by station. Only one station can be specified for high-speed servicing per channel.

For details regarding high-speed I/O service allocation, refer to section 3.4 *Remote Channel I/O Allocation*.

MC20 (4-axis Motion Module, JAMSC-120MMB10400) I/O servicing consists of exchanging motion reference data, motion commands set by ladder program motion instructions, and responses from the MC20.

There is no Motion Module service when the MC20 is not used.

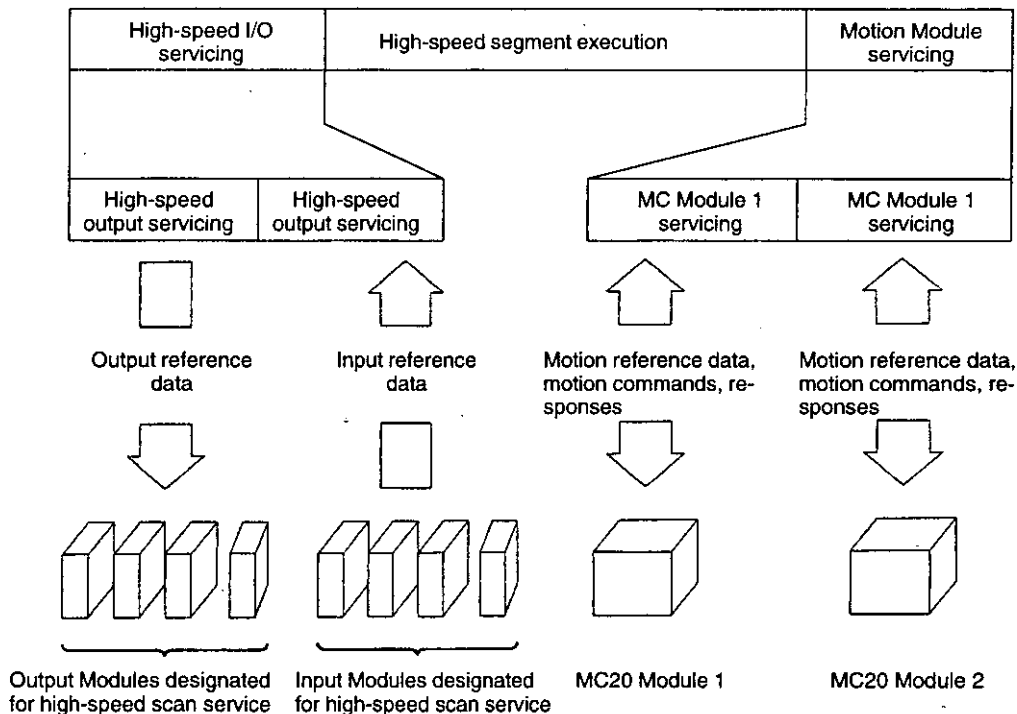


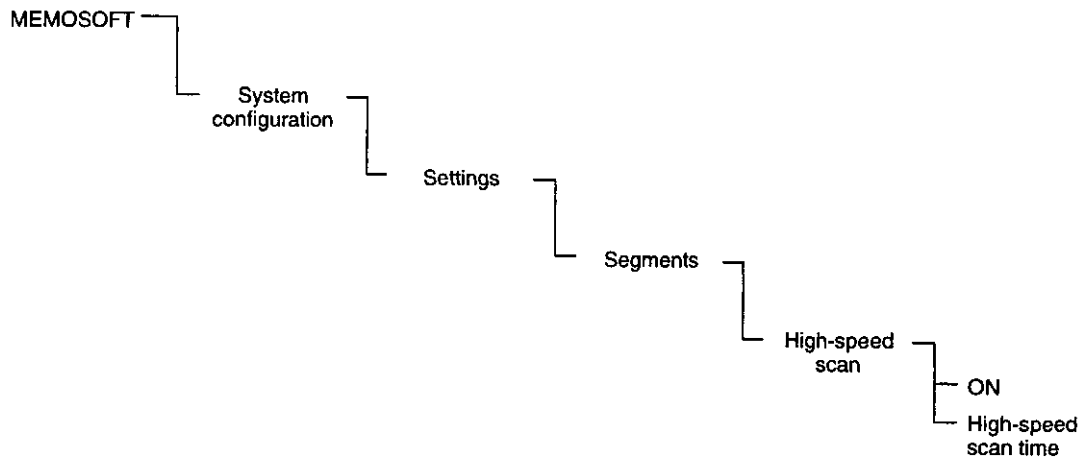
Figure 2.33 High-speed I/O Servicing and Motion Module Servicing

- Note**
- (1) If I/O references allocated for high-speed I/O or internal references used by the high-speed segment are referenced by a normal segment, there is a risk that the status or values may be changed even within the same normal scan. Take care to account for this in programming.
  - (2) When the high-speed scan is turned on, Motion Module servicing is carried out in a high-speed scan. When the high-speed scan is turned off, Motion Module servicing is carried out in a normal scan.

### 3. Setting High-speed Scan

The high-speed scan function is set using the MEMOSOFT. Within the MEMOSOFT system configuration, select "ON" for the high-speed scan under the segments menu under the settings menu. At this time the high-speed scan time can be set from 4 to 100 ms, in units of 1 ms.

For more detail on setting the high-speed scan, refer to the MEMOSOFT USER'S MANUAL.



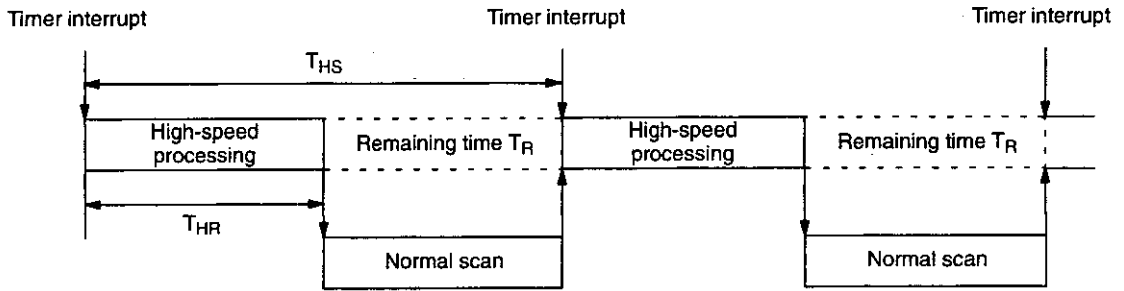
**Figure 2.34 Menu Configuration for Setting High-speed Scan**

**Note** Turn off the high-speed scan when not using it.

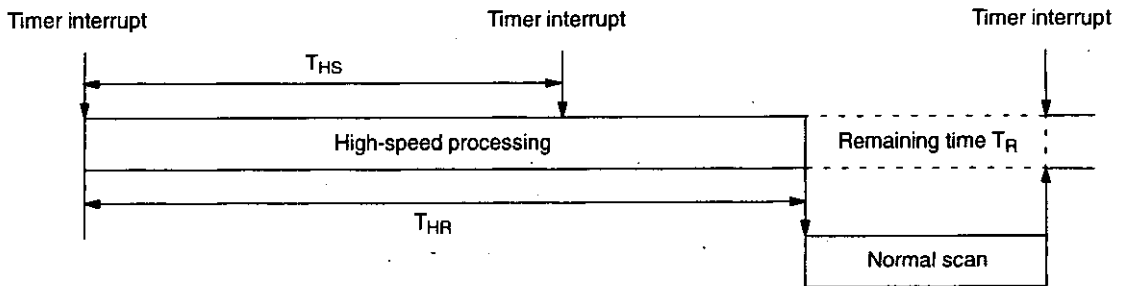
#### 4. High-speed Scan Time

As shown in *Figure 2.35*, a timer interrupt occurs at set intervals when the high-speed scan is set to "ON." The CPU Module begins scan processing with these timer interrupts.

When the actual high-speed processing time  $T_{HR}$  is longer than the set high-speed scan time  $T_{HS}$  (i.e.,  $T_{HS} < T_{HR}$ ), the CPU Module ignores the timer interrupts until the high-speed processing has been completed. It then begins the next high-speed scan with the timer interrupt following the completion of high-speed processing. Refer to *Figure 2.36*.



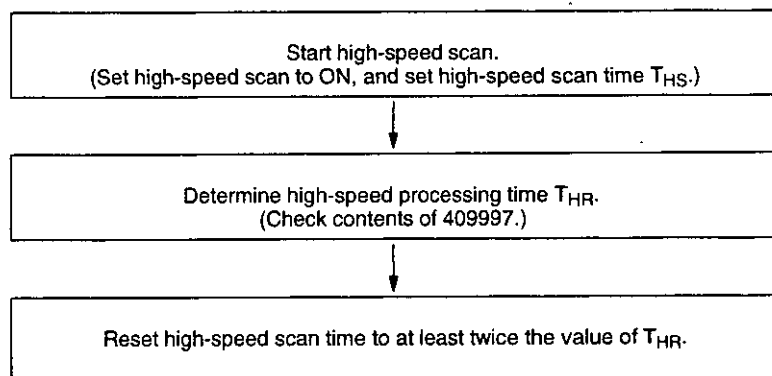
**Figure 2.35** Timer Interrupts and High-speed Scans (When  $T_{HS} > T_{HR}$ )



**Figure 2.36** Timer Interrupts and High-speed Scans (When  $T_{HS} < T_{HR}$ )

The high-speed processing time ( $T_{HR}$ ) that is actually required for the high-speed scan is stored in the high-speed scan time register (holding register 409997) in units of 1 ms.

This timer can be used to set the optimum high-speed scan time.



**Figure 2.37** Procedure for Setting High-speed Scan Time

It is recommended that the high-speed scan time,  $T_{HS}$ , be set to at least twice the time required for high-speed processing,  $T_{HR}$  ( $2 \times T_{HR} \leq T_{HS}$ ). For example, if the content of register 409997 is 5, set the high-speed scan time to at least 10.

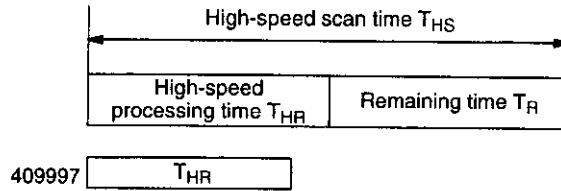


Figure 2.38 High-speed Scan Time and High-speed Scan Timer

**Note** As previously noted, the remaining time ( $T_R$ ) after the high-speed scan is used for normal scan processing. When there is little remaining time, therefore, the normal scan time will be longer and the scan will not be completed within the watchdog timer setting (240 ms) and operation will stop. Conversely, if the remaining time,  $T_R$ , is longer than the normal scan time, the normal scan will be processed more times than the high-speed scan.

## 2.5.2 The Segment Scheduler

### 1. What is the Segment Scheduler?

The segment scheduler allows you to set the conditions to be satisfied for execution of a maximum of 30 normal segments. The schedule can be used to shorten the scan time. In this section the term "segment" will mean "normal segment" unless otherwise indicated.

## 2. Allocating Control Specifications to Segments by Function

Organize control specifications such as manual operations or initial settings at startup as independent functional units according to their purposes. Clearly specify the conditions for starting each functional unit. For example, the startup signal can be used as a start condition for the initial settings segments. Use the start condition as the condition for the segment scheduler, and create a ladder logic treating that independent unit as a single segment. As explained earlier in this chapter 2.2, each segment consists of networks.

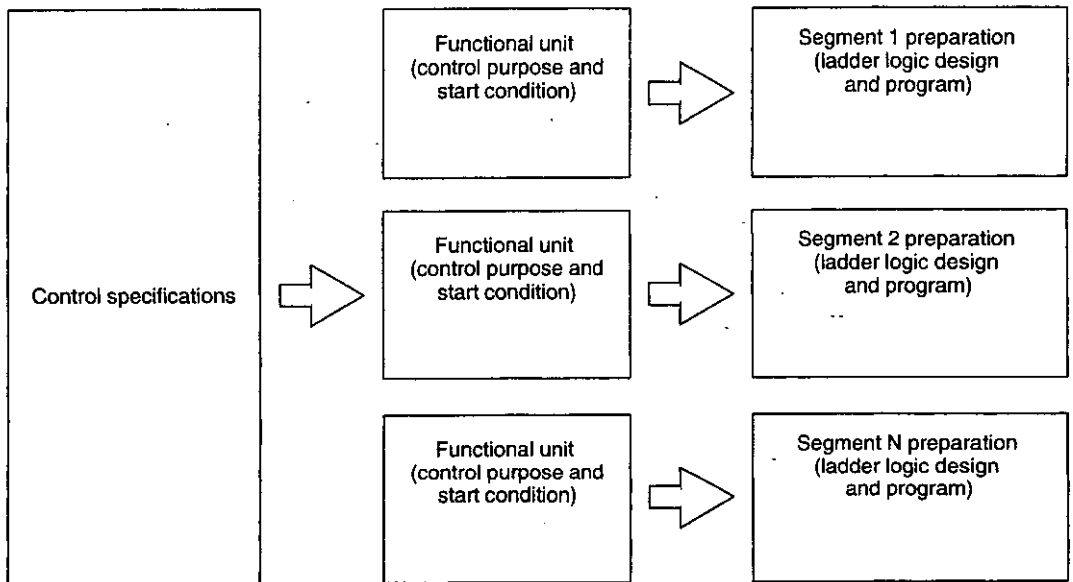
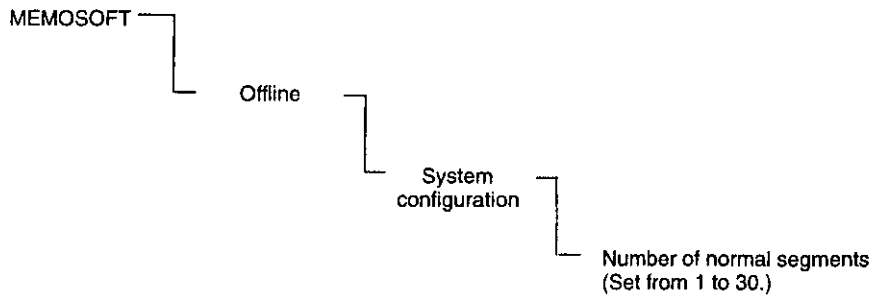


Figure 2.39 Allocation to Segments

## 3. Setting the Number of Segments

The factory setting for the number of normal segments is “1.” The MEMOSOFT can be used to change the setting to a maximum of 30 normal segments. This setting is made by means of the “segments” operation from the offline mode’s configuration menu. *Figure 2.40* shows the MEMOSOFT menu configuration for setting the number of segments. For more details, refer to the MEMOSOFT USER’S MANUAL.

A list of the segments in the user program and the number of networks programmed in each segment can be referenced by using the MEMOSOFT segment status display. *Figure 2.41* shows a display example. The segment status display is displayed by selecting it from the MEMOSOFT ladder edit menu.



**Figure 2.40 Menu Item for Setting the Number of Normal Segments**

	Segment	Networks
High-speed segment →	H	No program
Normal segment 1 →	L01	12
Normal segment 2 →	L02	23
Normal segment 3 →	L03	45
Subroutine segment →	S	No program

**Figure 2.41 Segment Status Example**

#### 4. Segment Scheduler Settings

The segment scheduler defines the order in which segments are executed in a scan. As shown in *Table 2.13*, the segment scheduler consists of schedule numbers, types, reference numbers, senses, and normal segment numbers.

**Table 2.13 Segment Scheduler Configuration**

Schedule No.	Type	Reference No.	Sense	Normal Segment No. (L Speed Segment)
1	Continuous			01
2	Controlled	100001	ON	02
3	Controlled	100002	OFF	03
:	:	:	:	:

##### A. Schedule Number

The schedule number is the number that indicates the order of execution in the scan.



**B. Type**

The type specifies whether the segment indicated by the schedule number will be executed with or without conditions. To carry out segments without conditions, specify "continuous". To carry out the segments under specified conditions, specify "controlled".

**C. Reference Number and Sense**

Reference numbers and senses are valid when "controlled" is specified as the type.

Use coil references 0xxxxx or input relay references 1xxxxx as reference numbers. Set the sense to either ON or OFF. When set to ON, the condition for executing the segment indicated by the schedule number is satisfied when the reference number signal is ON. When set to OFF, the condition for executing the segment indicated by the schedule number is satisfied when the reference number signal is OFF.

The segment scheduler setting is made with the MEMOSOFT system configuration. The segment scheduler setting screen is displayed by selecting segments under the settings menu. For details, refer to the MEMOSOFT USER'S MANUAL.

**5. Segment Schedule Example**

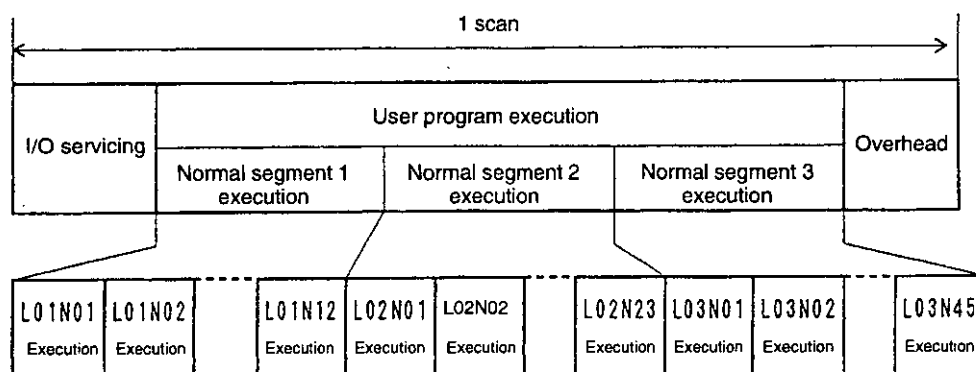
Figure 2.41 shows an example of a segment schedule.

**A. Continuous Scheduling Example**

If the segment scheduler is set with the contents shown in Table 2.14, the order of execution in one scan will be as shown in Figure 2.42.

**Table 2.14 Segment Schedule Example**

Schedule Number	Type	Reference Number	Sense	Normal Segment No. (L Speed Segment)
1	Continuous			L01
2	Continuous			L02
3	Continuous			L03



L□□: Normal segment number  
 N□□: Network number

**Figure 2.42 Continuous Scheduling Example**



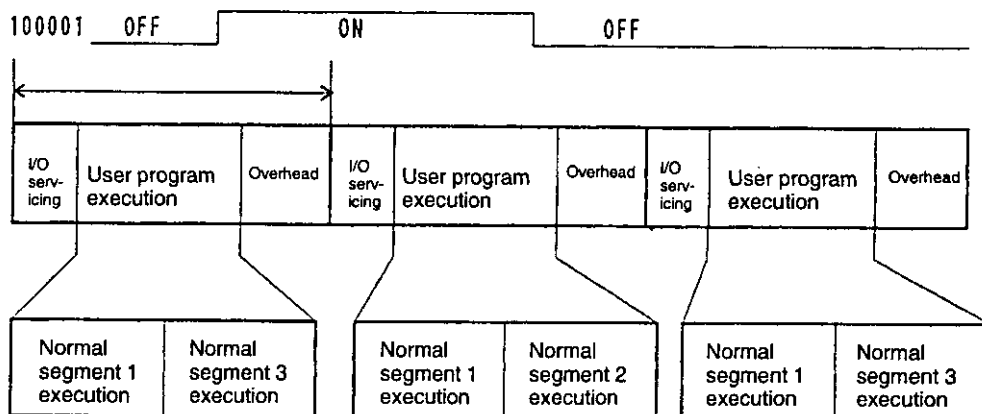
Continuous scheduling in which the number of segments is set to "1" and the entire ladder logic program is included in a single segment is equivalent to the normal scan method.

### B. Controlled Scheduling Example

If the segment scheduler is set with the contents shown in *Table 2.15*, the order of execution in one scan will be as shown in *Figure 2.43*.

**Table 2.15 Segment Schedule Example**

Schedule Number	Type	Reference Number	Sense	Normal Segment No.
1	Continuous			L01
2	Controlled	100001	ON	L02
3	Controlled	100001	OFF	L03



**Figure 2.43 Controlled Scheduling Example**

**Note** With controlled scheduling, the status of coils used in the segments where the specified conditions are not satisfied remains the same as the status in the last scan when the conditions were satisfied.

For example, when coil 000001 was ON in the last scan when the conditions were satisfied, the status of coil 000001 remains ON while the conditions are not satisfied. To change forcibly the ON/OFF status of coil used in the segment where the conditions are not satisfied, use SET BIT (SBIT) or RESET BIT (RBIT) instruction.

## 2.6 Start Mode

▣ This section explains the start modes for CPU Module.

2.6.1	Starting Mode Setting .....	2-58
2.6.2	Automatic RUN Mode .....	2-58
2.6.3	Normal Operating Mode .....	2-59

### 2.6.1 Starting Mode Setting

There are two kinds of starting modes for the CPU Module: “automatic RUN operating mode” and “normal operating mode”. The setting of Pin No.5 on the DIP switch of the CPU Module determines the starting mode.

Table 2.16 Start Mode Setting

Setting	Function
ON	Automatic RUN operating mode
OFF	Normal operating mode

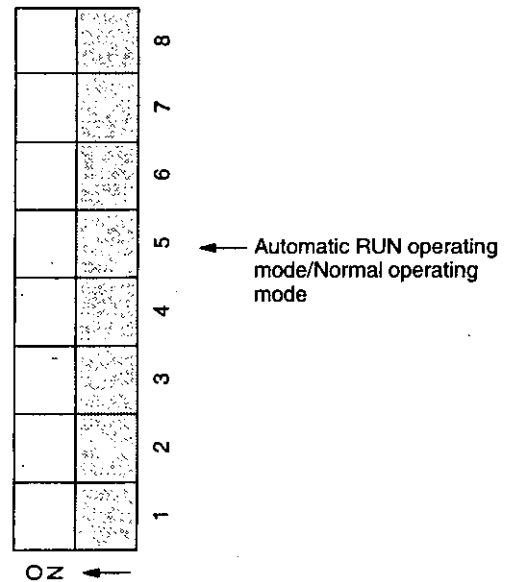


Figure 2.44 DIP Switch

**Note** Pin no. 5 of the DIP switch is effective only when the CPU Module is turned ON or reset.

### 2.6.2 Automatic RUN Operating Mode

When pin no. 5 of the CPU Module's DIP switch is turned ON, the Module goes into the automatic RUN operating mode. In this mode, the CPU Module will begin scanning in RUN status even if it was in STOP status just prior to the power interruption (as long as no errors are detected in the self-diagnosis during the startup sequence).

### 2.6.3 Normal Operating Mode

When pin no. 5 of the CPU Module's DIP switch is turned OFF, the Module goes into the normal operating mode. In this mode, the CPU Module will begin scanning in the status that was in effect just before the power interruption. For example, if the prior status was RUN, then the CPU Module will begin scanning in RUN status. Likewise, if the prior status was STOP, then the CPU Module will begin scanning in STOP status.

**Note** Even in the automatic RUN mode, or in the normal mode when the status just prior to the power interruption was RUN, the CPU Module will go to STOP status if an error is detected during the startup sequence.

## 2.7 Disable Operation

■ This section explains the disable operation.

2.7.1 Disable Operation ..... 2-60

### 2.7.1 Disable Operation

1) The disabling function can be used to disable any relays or coils and to force the disabled relays or coils on or off. This function is helpful for debugging ladder programs and making adjustments in trial operation.

2) Table 2.17 shows a list of references that can be used with the disable operation.

**Table 2.17 References Usable With Disable Operation**

Reference Name	Reference
Coils (including outputs and internal coils)	0xxxxx
Input relays	1xxxxx
Link coils	Dxxxxx
MC relays	Xxxxxx
MC control relays	Pxxxxx
MC coils	Yxxxxx
MC control coils	Qxxxxx
M code relays	Mxxxxx

**Note** u = 1 or 2

3) For example, the disabled input relay can be forced on or off regardless of the actual (on or off) status of the external input signal. Also, the disabled coil can be turned on or off independently of the ladder logic program.

4) For details regarding the disable operation, refer to the MEMOSOFT USER'S MANUAL.

5) Precautions on Disable Operation

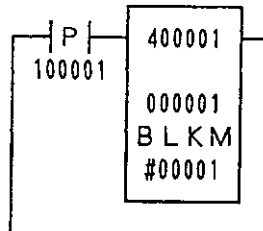
a) The disabled status is not cleared by turning the power supply to the Power Supply Module in the CPU rack on or off.

b) Unallocated input relays that were turned on when disabled will not turn off even when returned to normal status. To turn off these input relays, turn them off when they are disabled and then clear the disabled status.

c) When using the disable operation for coils (i.e., coils, link coils, MC coils, or MC control coils) that are used as destinations for data transfer instructions or matrix instruc-

tions, do not execute those instructions. If those data transfer instructions are executed, as shown in the following example, the disabled coils will be turned ON and OFF according to the instruction results.

Moreover, the disabled status will not be cleared.



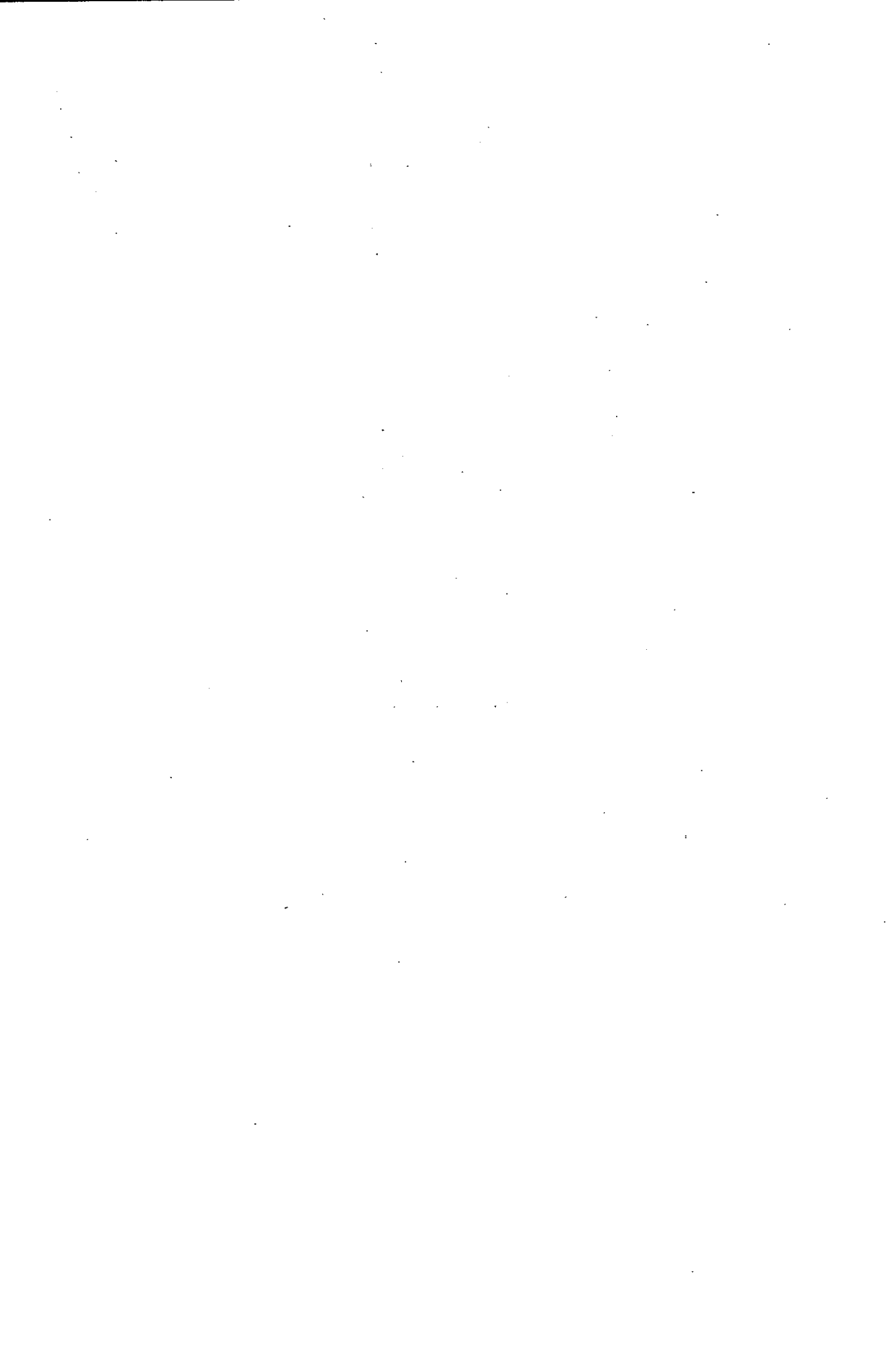
- (1) Suppose that the status of holding register 400001 and coils 000001 through 000016 is as follows:

400001 

1111	1111	1111	1111
------	------	------	------

Coils 000001 through 000016: All disable OFF

- (2) If input relay 100001 is turned from OFF to ON, coils 000001 through 000016 will all go to disabled ON status.



This chapter explains the I/O allocation required for the GL120 and GL130 to input signals from input devices and output signals to output devices.

<b>3.1</b>	<b>I/O Allocation Basics</b> .....	<b>3-2</b>
3.1.1	What is I/O Allocation? .....	3-2
3.1.2	I/O Section Configuration .....	3-4
3.1.3	Number of I/O Allocation Points .....	3-6
3.1.4	I/O Module Arrangement Example .....	3-7
3.1.5	I/O Reference Numbers .....	3-8
<b>3.2</b>	<b>Local Channel I/O Allocation</b> .....	<b>3-9</b>
3.2.1	I/O Reference Allocation .....	3-9
3.2.2	Module Type .....	3-11
3.2.3	I/O Data Format (Zoom Operation) .....	3-12
3.2.4	Local Channel Allocation Example .....	3-16
<b>3.3</b>	<b>High-speed Segment I/O Allocation</b> .....	<b>3-18</b>
3.3.1	Local Channel High-speed Segment I/O Allocation ...	3-18
3.3.2	Remote Channel High-speed Segment I/O Allocation .	3-18
<b>3.4</b>	<b>Remote Channel I/O Allocation</b> .....	<b>3-19</b>
3.4.1	I/O References .....	3-19
3.4.2	Module Type .....	3-19
3.4.3	I/O Data Format .....	3-19
3.4.4	Remote Channel Allocation Example .....	3-20



## 3.1 I/O Allocation Basics

■ This section explains the basics of I/O allocation.

3.1.1	What is I/O Allocation? .....	3-2
3.1.2	I/O Section Configuration .....	3-4
3.1.3	Number of I/O Allocation Points .....	3-6
3.1.4	I/O Module Arrangement Example .....	3-7
3.1.5	I/O Reference Numbers .....	3-8

### 3.1.1 What is I/O Allocation?

In order for the GL120 and GL130 to be able to input signals from input devices and to output signals to output devices, the I/O allocation must define the relationships between I/O signals and I/O references. The I/O allocation is executed using the MEMOSOFT, and the results are stored in the CPU Module memory as an I/O allocation table.

#### 1. Input Signals

Some input signals, such as limit switches, pushbutton switches, and proximity sensors, are expressed as ON/OFF data. Others, such as temperature sensors and digital switches, are expressed as numeric data.

With the GL120 and GL130, ON/OFF data is handled through digital inputs, and numeric data is handled through register inputs. Digital input data is input using input relays, while register input data is input using input registers.

#### 2. Output Signals

Some output signals, such as signals to electromagnetic contactors or solenoid valves, are expressed as ON/OFF data. Others, such as signals to heaters or numeric displays, are expressed as numeric data.

With the GL120 and GL130, the ON/OFF data is handled through digital outputs, and numeric data is handled through register outputs. Digital output data is output using output coils, while register output data is output using output registers.

#### 3. Locations

With the GL120 and GL130, the places where I/O Modules are mounted are called their locations. Locations are comprised of channels, stations, racks, and slots. Locations are explained in more detail in *3.1.2 I/O Section Configuration*.

GL120 and GL130 I/O use a free location method, so any I/O Module can be mounted at any location.

## 4. I/O Allocation Data

I/O allocation data consists of the following items. For details, refer to *3.2 Local Channel I/O Allocation*.

### A. Module Types

The Module type specifies the type of I/O Module being used. A part of the I/O Module model number is used for this specification.

### B. I/O Reference Numbers

I/O reference numbers specify the reference numbers used for exchanging I/O data with I/O Modules.

### C. I/O Data Format (Set With Zoom Operation)

#### 1) I/O Data Type

The I/O data type specifies the format for exchanging I/O data and I/O references. For example, either BCD or BIN (binary) data can be specified for inputting numeric data.

#### 2) Bit Order

The bit order specifies the most significant (MSB) and least significant bits (LSB) for I/O purposes.

#### 3) Service Scan

The service scan specifies the high-speed or normal scans for local station Modules.

#### 4) Timeout Outputs

Specifies output data when the CPU stops.

#### 5) Timeout Output Data

Sets the contents of output data used when the CPU stops.

## 5. I/O Allocation Operations

I/O allocation operations are carried out while ladder program solving and I/O servicing of CPU Module are stopped. They are done independently for each location, so the I/O allocation for a given location can be changed without affecting other locations. All that is necessary when mounting additional I/O Modules is to add the allocation for the new Modules to the existing I/O allocation.

For more detail regarding I/O allocation, refer to the MEMOSOFT user's manual.

## 3.1.2 I/O Section Configuration

As shown in *Figure 3.1* the I/O section of the GL120 or GL130 consists of a local channel, remote channel 1, and remote channel 2, for a total of three channels. An I/O section configuration can consist of the local channel only. *Figure 3.2* shows the GL120/GL130 location configuration.

### 1. Local Channel

All I/O Modules mounted to the CPU Module's Mounting Base (JRMSI-120XBP0□□00, with □□ representing a number indicating the number of slots) or to Mounting Bases connected by Rack Cables (JZMSZ-120W0100-□□, with □□ representing a number indicating the cable length) via I/O Expander Modules (JAMSC-120CBE37000) on the right side of the CPU Module's Mounting Base make up what is called the local channel for I/O allocation purposes.

### 2. Remote Channels

All I/O connected by remote cable from Remote I/O Driver Modules (JAMSC-120CRD13100) make up what is called the remote channel.

A maximum of two Remote I/O Driver Modules can be mounted as part of a GL120 or GL130, and they are separated as remote channels 1 and 2. The remote channel number is set by a switch on the Remote I/O Driver Module.

### 3. Remote Stations

All I/O Modules mounted to the Mounting Base of a Remote I/O Receiver Module (JAMSC-120CRR13100) or to Mounting Bases connected by Rack Cables via I/O Expander Modules on the right side of the Remote I/O Receiver Module's Mounting Base make up what is called a remote station for I/O allocation purposes.

A station address from 1 to 15 is assigned to each remote station for identification. The same station address cannot be used more than once in the same remote channel. The station address is set by a rotary switch on the Remote I/O Receiver Module.

Set the rotary switch to the station address.

### 4. Racks

All of the I/O Modules mounted to a single Mounting Base are called a rack for I/O allocation purposes. Racks are numbered from 1 to 4 (for identification).

For the local channel, the rack with the CPU Module becomes rack 1. For the remote channels, the rack with the Remote I/O Receiver Module becomes rack 1. The same rack number cannot be used more than once in either the same local channel or the same remote channel.

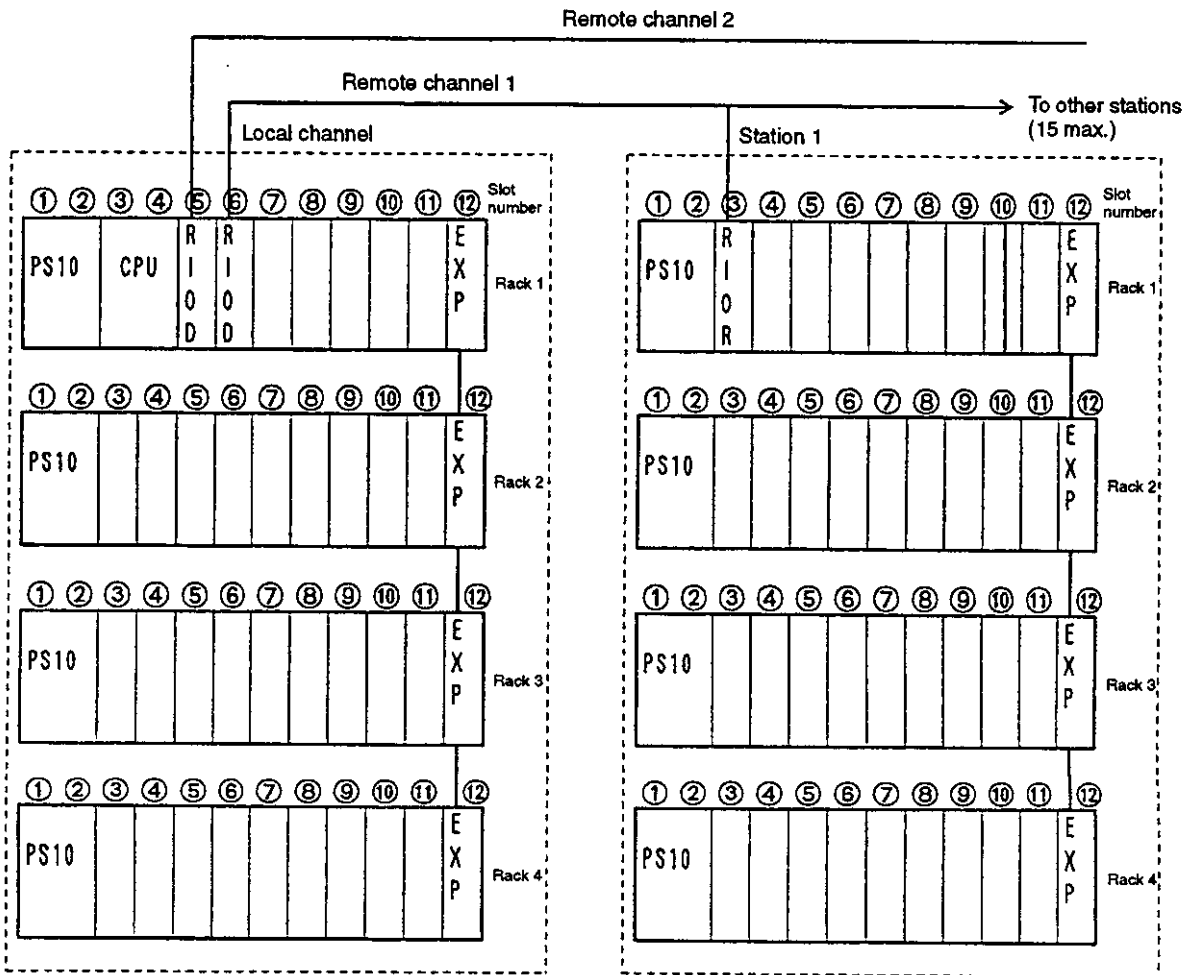
The rack number is set by a rotary switch on the I/O Expander Module (JAMSC-120CBE37000). Set the rotary switch to a number that is one less than the rack number. For example, set "0" for rack 1, "1" for rack 2, and so on.

Racks can be numbered in any order desired, but from the standpoint of maintenance considerations it is recommended that they be arranged in order (i.e., rack 1, rack 2, rack 3, etc.).

### 5. Slots

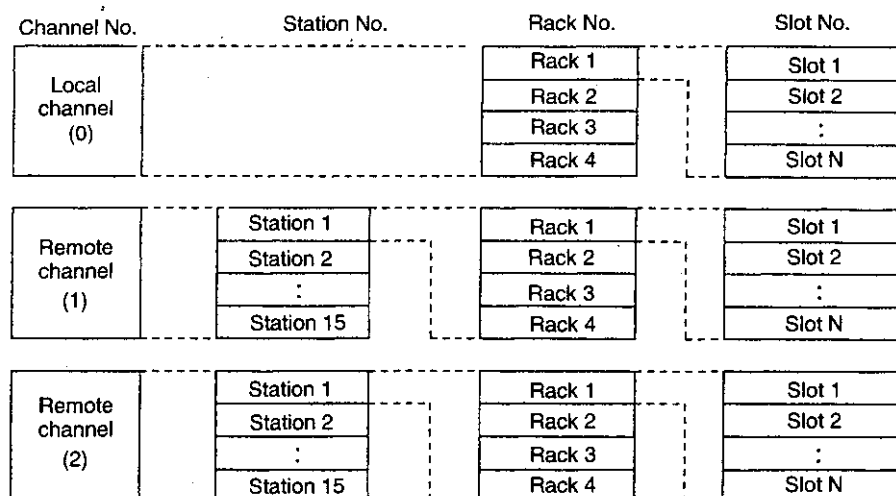
The mounting locations for I/O Modules on racks are called slots for I/O allocation purposes. To identify the locations, slot numbers are assigned in order from left to right on each rack.

Unlike station and rack numbers, there are no switches to set slot numbers.



- CPU: CPU Module (DDSCR-120CPU34100, DDSCR-120CPU54100)
- PS10: Power Supply Module (JRMSP-120CPS11300)
- EXP: Expander Module (JAMSC-120CBE37000)
- RIOD: Remote I/O Driver Module (JAMSC-120CRD13100)
- RIOR: Remote I/O Receiver Module (JAMSC-120CRR13100)

Figure 3.1 I/O Section Configuration



**Note** "N" represents the number of slots on the particular Mounting Base that is used. (Refer to Table 3.2.)

**Figure 3.2 Location Configuration**

### 3.1.3 Number of I/O Allocation Points

Table 3.1 shows the numbers of I/O points that can be allocated for the GL120 and GL130. Table 3.2 shows the number of slots available on each of the five Mounting Bases.

**Table 3.1 Numbers of I/O Points Available**

Item		GL120	GL130
Number of I/O points	Digital I/O	1,024 points max.	4,096 points max.
	Register I/O	512 registers max. (1 register = 16 bits)	
Local channels	Number of channels	1	
	Number of racks	4/channel max.	
	Number of slots	16/rack max. (See note.)	
	Number of I/O Modules that can be mounted	54 max.	
Remote channels	Number of channels	2	
	Number of stations	15/channel max.	
	Number of racks	4/station max.	
	Number of slots	16/rack max. (See note.)	
	I/O points per station	(Digital input points ÷ 8) + (Register input points x 2) ≤ 512 bytes (Digital output points ÷ 8) + (Register output points x 2) ≤ 512 bytes	

**Note** The maximum number of slots is determined by the particular Mounting Base that is used. The values in this table assume that a JRMSI-120XBP01600 Mounting Base (16-slot type) is used.

Table 3.2 Mounting Base Slots

Mounting Base	Number of Slots
JRMSI-120XBP00600	6
JRMSI-120XBP00800	8
JRMSI-120XBP01000	10
JRMSI-120XBP01200	12
JRMSI-120XBP01600	16

### 3.1.4 I/O Module Arrangement Example

As long as the number of I/O points is within the permissible range, I/O Modules can be mounted in any desired arrangement on the remote channels, and it is also acceptable to leave open slots between them. From the standpoint of maintenance and wiring, however, it is recommended that they be arranged in related groups (e.g., input and output, power supply, applications, etc.).

Figure 3.3 shows an example of an I/O Module arrangement.

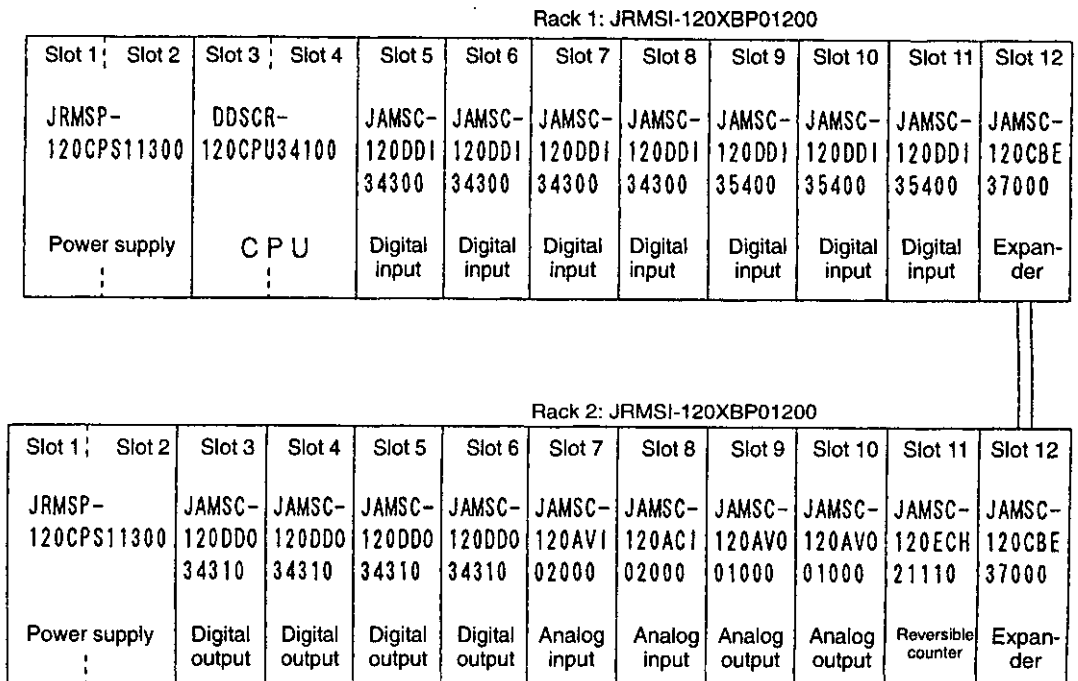


Figure 3.3 I/O Module Arrangement Example

### 3.1.5 I/O Reference Numbers

Table 3.3 shows an example of I/O reference numbers used for I/O allocation.

**Table 3.3 I/O Reference Number Example**

I/O Type	Examples of References Used for I/O Allocation		
		GL120	GL130
Digital input	Input relays	100001 to 101024	100001 to 104096
Register input	Input registers	300001 to 300512	
Digital output	Output coils	000001 to 001024	000001 to 004096
Register output	Output registers	400001 to 400512	

The above table shows the ranges of I/O numbers that can be allocated, but in actual practice the following control conditions apply.

#### 1) GL120

- Digital input + Digital output  $\leq$  1,024 points
- Register input + Register output  $\leq$  512 registers

#### 2) GL130

- Digital input + Digital output  $\leq$  4,096 points
- Register input + Register output  $\leq$  512 registers
- When input relays up to 104096 are used, the system configuration reference range of input relays must be changed from 101024 to 104096.

## 3.2 Local Channel I/O Allocation

In order for the GL120 and GL130 to input signals from input devices and output signals to output devices, the local channel I/O allocation specifies the types of I/O Modules, I/O references, and I/O data format for each location (i.e., racks and slots).

3.2.1	I/O Reference Allocation .....	3-9
3.2.2	Module Type .....	3-11
3.2.3	I/O Data Format (Zoom Operation) .....	3-12
3.2.4	Local Channel Allocation Example .....	3-16

### 3.2.1 I/O Reference Allocation

- 1) To allocate I/O references, the beginning number of the I/O references to be used by an I/O Module is designated. For example, as shown in *Figure 3.4*, if input relay 100001 is designated as the beginning number for a 24-VDC, 16-point Input Module (JAMSC-120DDI34300), the input references will be allocated in order beginning with the Module's leading input. That is, 100001 will be allocated to input signal 1, 100002 to input signal 2, and so on. This method is used for the "MSB" (most significant bit) setting.

As an option, the process can be reversed, so that the beginning number will be allocated to the largest input number. With this method, 10001 will be allocated to input 16, 10002 to input 15, and so on. This method is used for the "LSB" (least significant bit) setting. For details, refer to section 3.2.3 *I/O Data Format*. The MEMOSOFT initial setting is for the MSB setting.

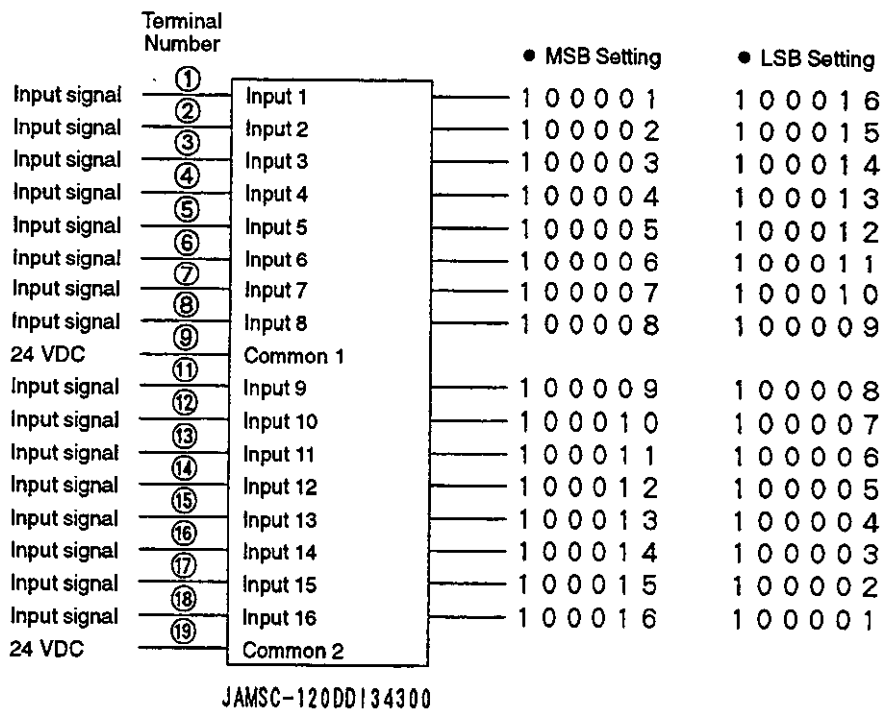


Figure 3.4 I/O Reference Allocation Example



2) When the I/O allocation is digital, the leading I/O reference number must be as follows:

- Leading number for digital input =  $100001 + 16n$   
Where GL120:  $n = 0$  to 63; GL130:  $n = 0$  to 255
- Leading number for digital output =  $000001 + 16n$   
Where GL120:  $n = 0$  to 511; GL130:  $n = 0$  to 511)

Thus 100001 and 000001 can be used as leading numbers, but 100002 and 000002 cannot.

3) Registers can also be allocated to I/O references for digital-type I/O Modules. If, for example, 300001 rather than 100001 was allocated as the beginning number for the I/O Module in Figure 3.4, the ON/OFF data for inputs 1 through 16 would be entered in 300001 as shown in Figure 3.5. This is the LSB setting method. The turning ON and OFF of input signals is expressed by the respective "1" or "0" status of individual bits in 300001. For example, the status of bit number 6 indicates the ON/OFF status of input 10. If the bit order were reversed, then the MSB setting would be used.

The initial MEMOSOFT setting for the input and output register bit order is "LSB."

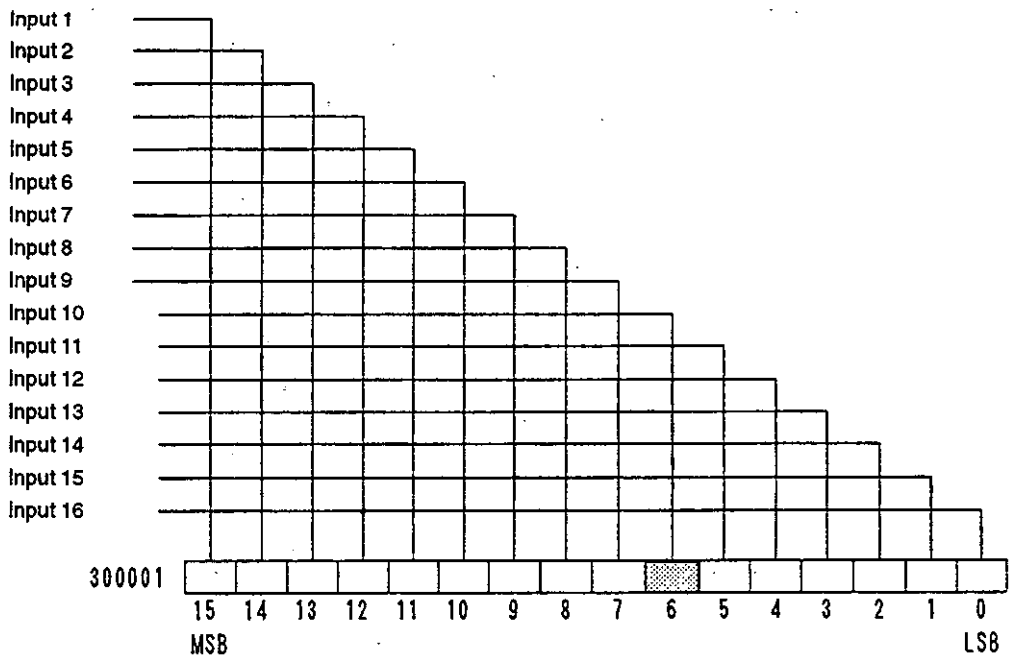


Figure 3.5 Input Register Allocation (LSB Setting) for Digital Input Module

### 3.2.2 Module Type

- 1) For the "Module type" the rightmost 11 digits of the model number are used to indicate the type of I/O Module mounted in a slot. *Table 3.4* shows the I/O Modules and the Module types.
- 2) The "reference number" column in the table shows the types of I/O references that can be used with each Module. "1x" indicates input relay references (1xxxxx), "3x" indicates input register references (3xxxxx), "0x" indicates output coil references (0xxxxx), and "4x" indicates output register references (4xxxxx).
- 3) "1x or 3x" in the table indicates that either input relay references or input register references can be used as references for I/O allocation. When there are multiple lines of references, as with single-axis Motion Module, it indicates that all specified references must be allocated.
- 4) For single-axis Motion Modules and Counter Modules, 1x, 3x, 0x, and 4x must all be allocated. Modules which allocate several kinds of references to a single I/O Module in this way are called "Compound Modules." The "number of points" column in the table shows the number of I/O points that those I/O Modules have. For register I/O, one register is equivalent to 16 bits.

## I/O Allocation

### 3.2.3 I/O Data Format (Zoom Operation)

**Table 3.4 I/O Modules and Module Types**

Model Number	Specifications	Module Type	Reference Numbers	Number of Points	Remarks
JAMSC-120DAI54300	100 VAC, 16-pt input	120DAI54300	1x or 3x	16 pts or 1 register	
JAMSC-120DAI74300	200 VAC, 16-pt input	120DAI74300	1x or 3x	16 pts or 1 register	
JAMSC-120DDI34300	12 to 24 VDC, 16-pt input	120DDI34300	1x or 3x	16 pts or 1 register	
JAMSC-120DDI35400	12 to 24 VDC, 32-pt input	120DDI35400	1x or 3x	32 pts or 2 registers	
JAMSC-120DAO84300	100 to 200 VAC, 16-pt output	120DAO84300	0x or 4x	16 pts or 1 register	
JAMSC-120DDO34310	12 to 24 VDC, 16-pt sinking output	120DDO34310	0x or 4x	16 pts or 1 register	
JAMSC-120DDO35410	12 to 24 VDC, 32-pt sinking output	120DDO35410	0x or 4x	32 pts or 2 registers	
JAMSC-120DRA84300	Relay contacts, 16-pt output	120DRA84300	0x or 4x	16 pts or 1 register	
JAMSC-120ACI02000	Analog current input, 4 channels	120ACI02000	3x	5 registers	
JAMSC-120AVI02000	Analog voltage input, 4 channels	120AVI02000	3x	5 registers	
JAMSC-120ACO01000	Analog current output, 2 channels	120ACO01000	4x	2 registers	
JAMSC-120AVO01000	Analog voltage output, 2 channels	120AVO01000	4x	2 registers	
JAMSC-120EHC21110	Counter Module, 1 channel	120EHC21110	1x 3x 0x 4x	16 pts 4 registers 16 pts 4 registers	
JAMSC-120MMB10100	1-axis Motion Module	120MMB10110	1x 3x 0x 4x	32 pts 14 registers 32 pts 14 registers	
JAMSC-120CRD21110	Uniwire interface	120CRD21110	1x or 3x 0x or 4x	272 pts or 17 registers	See note.

0x: Output coil references (00xxxx)

1x: Input relay references (1xxxxx)

3x: Input register references (3xxxxx)

4x: Output register references (4xxxxx)

**Note** includes 16 input points or 1 register for Module monitoring.

- 5) In the actual I/O allocation operation, the number of I/O points is shown in the form of the allocated I/O reference range display when the Module type and I/O reference are allocated. If, for example, the Module type is "120DAI54300" and the I/O reference is "100001," "100001 – 100016" will be displayed on the actual I/O screen to show the 16 input points that are allocated.

### 3.2.3 I/O Data Format (Zoom Operation)

- 1) The I/O data format specifies the format in which I/O data will be handled for allocated I/O references. In the actual I/O allocation operation, this is done by means of the zoom operation. Some Modules do not need the zoom operation.

## 2) Setting the Bit Order

- a) There are two ways in which the bit order can be set: MSB (most significant bit) and LSB (least significant bit). With the MSB setting, the leading I/O reference number is allocated to the smallest input number. With the LSB setting, the order is reversed so that the leading number is allocated to the largest input number.

The MEMOSOFT's factory setting is "MSB" for digital I/O and "LSB" for register I/O.

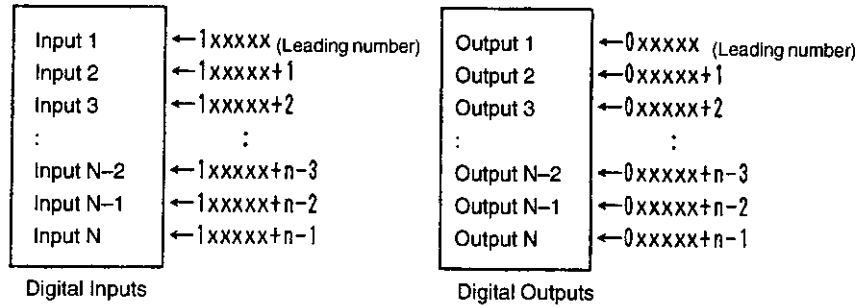


Figure 3.6 Digital I/O Data Format (Bit Order Setting = MSB)

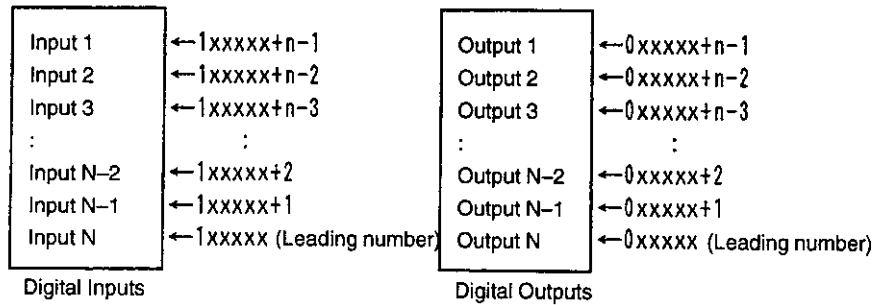
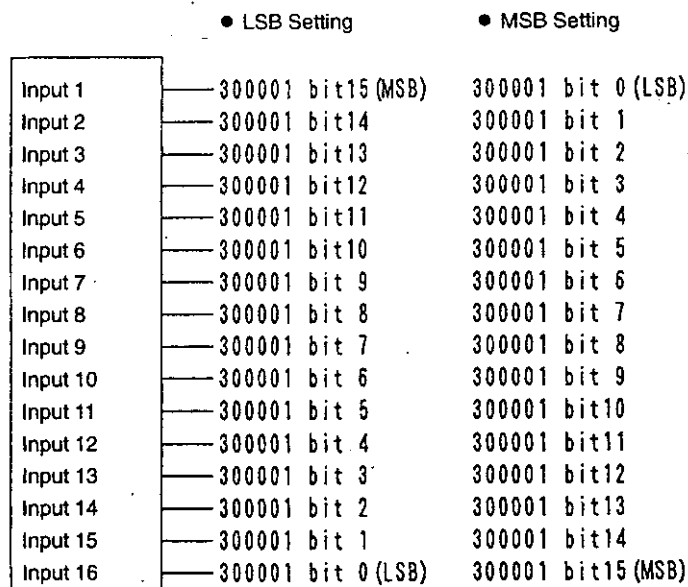


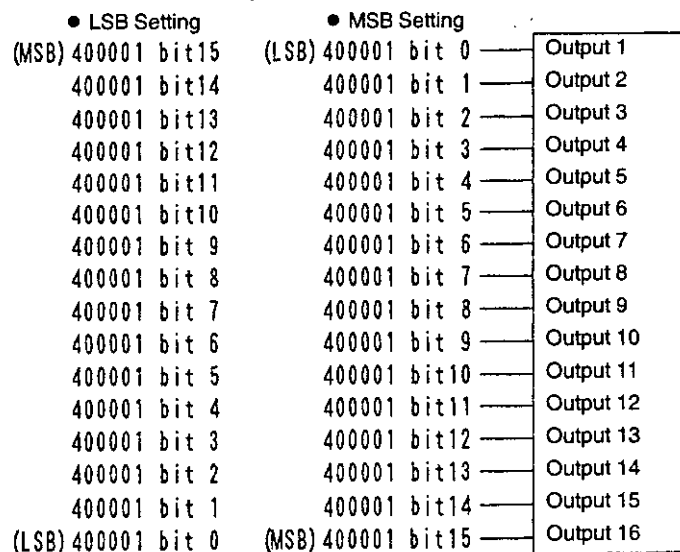
Figure 3.7 Digital I/O Data Format (Bit Order Setting = LSB)

b) In the example shown in *Figure 3.8*, input 16 enters in bit 0 (LSB) of 300001 when the input register setting is "LSB."



**Figure 3.8 Input Register Allocation Beginning with 300001 (16 Bits)**

c) In the example shown in *Figure 3.9*, bit 0 (LSB) of 400001 is output to output 16 when the output register setting is "LSB."



**Figure 3.9 Output Register Allocation Beginning with 400001 (16 Bits)**

**Note** When using the default setting, LSB, for register I/Os, the order of output signals to the I/O Modules is the reverse of that for the GL 60 series. To put them in the same order as the GL 60 series, select MSB.

### 3) Setting Binary or BCD as the I/O Data Type

This setting specifies whether data I/O will be handled as binary or BCD data for register I/O references.

The factory setting is for binary data.

Fig. 3-10 shows an example for BCD data.

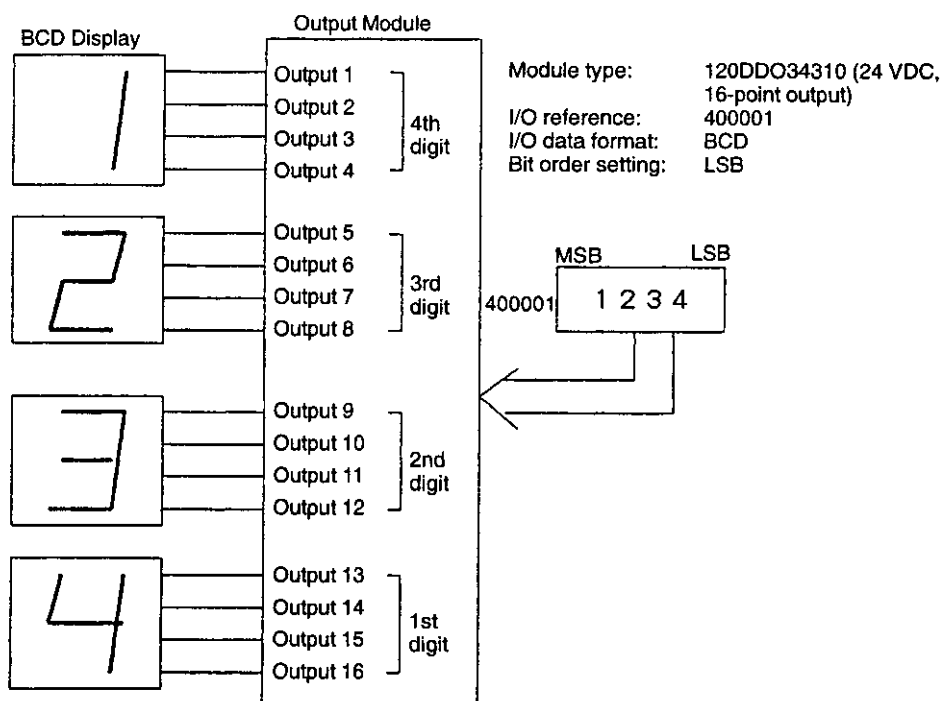


Figure 3.10 BCD Output

### 4) Setting the Service Scan

There are two kinds of service scans: normal and high-speed. The service scan designation has meaning only when the high-speed segment is used. When the high-speed scan is designated, the specified I/O Module is serviced before solving the high-speed segment of the ladder logic program. High-speed servicing of I/O Modules combining with the high-speed scan is called high-speed segment I/O. For details regarding the high-speed segment, refer to chapter 2 *Operating Principles*.

The factory setting is for the normal scan.

### 5) Specifying the Timeout Output

The output data can be selected for when the CPU Module is stopped after having been in RUN status. Either the status from directly before the CPU Module was stopped, or a preset status can be output.

The factory setting is for all OFF.

## 6) Specifying the Timeout Output Data

If preset data is specified for the timeout output, the data output when the CPU Module stops is set as the timeout output data. The data set here is the PLC reference data image, so it is converted and output according to the bit order that is set.

The factory setting for the set value is "0."

## 3.2.4 Local Channel Allocation Example

The I/O example from *Figure 3.3* is shown in *Table 3.5*. The default values are used for the I/O data format, so for 1x and 0x the data type is "binary" and the bit order is "MSB"; for 3x and 4x the data type is "binary" and the bit order is "LSB." The service scan is "normal."

Table 3.5 Local I/O Allocation Example

Location		Module Type	I/O Reference No.								
Rack No.	Slot No.		Digital Input		Register Input		Digital Output		Register Output		
			Reference	No. of Pts	Reference	No. of Pts	Reference	No. of Pts	Reference	No. of Pts	
1	1	Power Supply Module									
	2										
	3	CPU Module									
	4										
	5	120DDI34300	100001	16							
	6	120DDI34300	100017	16							
	7	120DDI34300	100033	16							
	8	120DDI34300	100049	16							
	9	120DDI35400	100065	32							
	10	120DDI35400	100097	32							
	11	120DDI35400	100129	32							
	12	Expander									
2	1	Power Supply Module									
	2										
	3	120DDO34310					000001	16			
	4	120DDO34310					000017	16			
	5	120DDO34310					000033	16			
	6	120DDO35410					000049	32			
	7	120AVI02000			300001	5					
	8	120ACI02000			300006	5					
	9	120AVO01000								400001	2
	10	120ACO01000								400003	2
	11	120EHC21110	100161	16	300011	4	000081	16		400005	4
	12	Expander									

**Note** No allocation is required for Power Supply Modules, CPU Modules, Expander Modules, or empty slots.

**Note** (1) Do not allocate I/O for Modules that will be inputting and outputting data by means of the DIRECT IN (DIN) and DIRECT OUT (DOU) instructions. These instructions input

and output data while solving the ladder program with no relation to I/O servicing. If I/O is allocated for those Modules, the I/O data may be changed even during normal I/O servicing, thereby disrupting inputs and outputs.

- (2) When either DIN or DOUT is solved, an error will be generated if I/O has been allocated for any of the Modules involved.
- (3) DIN and DOUT can be used only with the local channel.



### 3.3 High-speed Segment I/O Allocation

High-speed segment I/O refers to the I/O Modules serviced with the high-speed scan. For details regarding the high-speed scan, refer to chapter 2 *Operating Principles*. This section explains high-speed segment I/O allocation.

3.3.1 Local Channel High-speed Segment I/O Allocation ..... 3-18  
 3.3.2 Remote Channel High-speed Segment I/O Allocation ..... 3-18

#### 3.3.1 Local Channel High-speed Segment I/O Allocation

As explained in subsection 3.2.3 *I/O Data Format*, high-speed segment I/O allocation for the local channel is carried out by Module. I/O Modules designated "high-speed" for service scans are allocated as high-speed segment I/O.

If, for example, I/O Modules in slot 5 of rack 1 and slot 3 of rack 2 from *Table 3.5* are allocated for high-speed servicing, input relays 100001 through 100016 and output coils 000001 through 000016 will be serviced before the high-speed segment is solved.

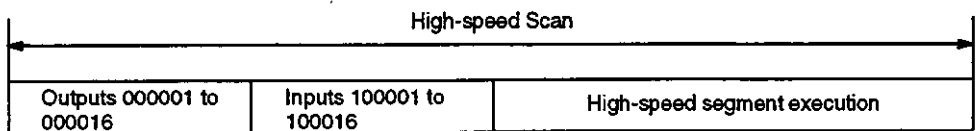


Figure 3.11 Local Channel High-speed Segment I/O Service

#### 3.3.2 Remote Channel High-speed Segment I/O Allocation

High-speed segment I/O allocation for the remote channels is carried out by remote station. One station per remote channel can be allocated as high-speed segment I/O. Therefore, when two remote channels are used, high-speed segment I/O can be allocated for a maximum of two stations.

When the station 2 is allocated for high-speed servicing, input relays 100177 through 100368 will be serviced before the high-speed segment is solved. (Refer to *Table 3.6*.)

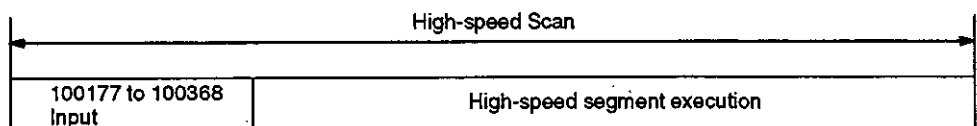


Figure 3.12 Remote Channel High-speed Segment I/O Service

## 3.4 Remote Channel I/O Allocation

I/O allocation for remote channels is basically the same as for the local channel, except that, in the case of remote channels, remote channel numbers and station numbers are added to define the location. (Refer to *Figure 3.2*) It is therefore necessary to specify the channel and station numbers when allocating I/O for remote channels. Within each station, the considerations are the same as for the local channel. Moreover, just like the local channel I/O allocation, the remote channel I/O allocation consists of I/O reference numbers, Module types, and I/O data formats.

3.4.1	I/O References .....	3-19
3.4.2	Module Type .....	3-19
3.4.3	I/O Data Format .....	3-19
3.4.4	Remote Channel Allocation Example .....	3-20

### 3.4.1 I/O References

I/O references are set in the same way as for the local channel. For details, refer to section 3.2.1 *I/O Reference Allocation*.

### 3.4.2 Module Type

The Module type is set in the same way as for the local channel, except that for the remote channels the number of I/O points allocated per station is restricted as follows:

Number of input bytes =  
 $(\text{Total digital input points}) \div 8 + (\text{Total register input points}) \times 2 \leq 512 \text{ bytes}$

Number of output bytes =  
 $(\text{Total digital output points}) \div 8 + (\text{Total register output points}) \times 2 \leq 512 \text{ bytes}$

For details, refer to 3.2.2 *Module Type*.

### 3.4.3 I/O Data Format

Except for the points without a service scan designation, the format setting is the same as for the local channel. For details, refer to 3.2.3 *I/O Data Format*. The high-speed scan for the remote channels is designated in station units.

For details, refer to 3.3 *High-speed Segment I/O Allocation*.

### 3.4.4 Remote Channel Allocation Example

Figure 3.13 shows an example of remote channel I/O Module arrangement, and Table 3.6 shows an I/O allocation example.

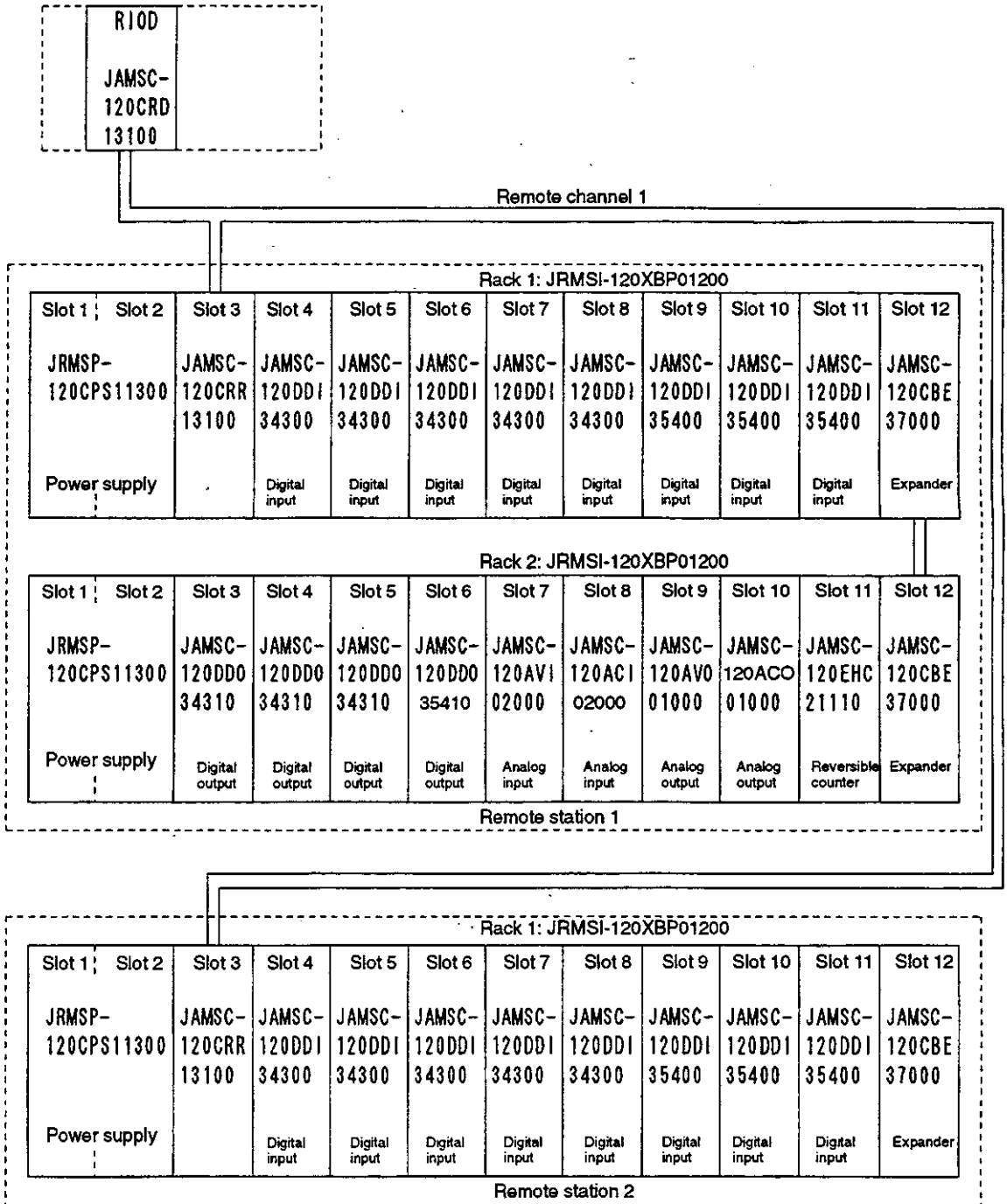


Figure 3.13 I/O Module Arrangement Example

Table 3.6 Remote I/O Allocation Example

## Station 1

Remote Channel	I/O Allocation			Module Type	I/O Reference No.										
	Station Address	Rack No.	Slot No.		Digital Input		Register Input		Digital Output		Register Output				
					Ref.	No. of Points	Ref.	No. of Points	Ref.	No. of Points	Ref.	No. of Points			
1	1	1	1	Power Supply Module											
			2												
			3	Remote I/O Receiver											
			4	120DDI34300	100001	16									
			5	120DDI34300	100017	16									
			6	120DDI34300	100033	16									
			7	120DDI34300	100049	16									
			8	120DDI34300	100065	16									
			9	120DDI35400	100081	32									
			10	120DDI35400	100113	32									
			11	120DDI35400	100145	32									
			12	Expander Module											
		2	2	1	Power Supply Module										
				2											
				3	120DDO34310					000001	16				
				4	120DDO34310					000017	16				
				5	120DDO34310					000033	16				
				6	120DDO35410					000049	32				
				7	120AVI02000			300001	5						
				8	120ACI02000			300006	5						
				9	120AVO01000								400001	2	
				10	120ACO01000								400003	2	
				11	120EHC21110	100177	16	300011	4	000081	16	400005	4		
				12	Expander Module										

**Note** No allocation is required for Power Supply Modules, CPU Modules, Remote I/O Receiver Modules, I/O Expander Modules, or empty slots.

**I/O Allocation**

**3.4.4 Remote Channel Allocation Example cont.**

**Station 2**

Remote Channel	I/O Allocation			Module Type	I/O Reference No.								
	Station Address	Rack No.	Slot No.		Digital Input		Register Input		Digital Output		Register Output		
					Ref.	No. of Points	Ref.	No. of Points	Ref.	No. of Points	Ref.	No. of Points	
1	2	1	1	Power Supply Module									
			2										
			3	Remote I/O Receiver									
			4	120DDI34300	100193	16							
			5	120DDI34300	100209	16							
			6	120DDI34300	100225	16							
			7	120DDI34300	100241	16							
			8	120DDI34300	100257	16							
			9	120DDI35400	100273	32							
			10	120DDI35400	100305	32							
			11	120DDI35400	100337	32							
			12	Expander Module									

**Note** No allocation is required for Power Supply Modules, CPU Modules, Remote I/O Receiver Modules, I/O Expander Modules, or empty slots.

# Overview of Instructions

# 4

This chapter provides an overview of the GL120/GL130 instructions.

<b>4.1</b>	<b>Instructions .....</b>	<b>4-2</b>
<b>4.2</b>	<b>Instruction Outlines .....</b>	<b>4-8</b>
4.2.1	Basic Instructions .....	4-9
4.2.2	Math Instructions .....	4-14
4.2.3	Data Transfer Instructions .....	4-41
4.2.4	Indexed Block Transfer Instructions .....	4-55
4.2.5	Matrix Instructions .....	4-60
4.2.6	Bit Manipulation Instructions .....	4-72
4.2.7	Data Conversion Instructions .....	4-76
4.2.8	Other Data Manipulation Instructions .....	4-81
4.2.9	System Status Monitoring Instruction .....	4-91
4.2.10	Sequencers .....	4-92
4.2.11	Program Control Instructions .....	4-94
4.2.12	I/O Control Instructions .....	4-101

## 4.1 Instructions

■ This section consists of tables of basic information for GL120/GL130 instructions.

**Table 4.1 Instructions**

Class	Subclass	Name	Symbol	Outline	Details
Basic Instructions (16 instructions)	Relays	1) Normally Open (N.O.) Contact		Refer to 4.2.1	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 1
		2) Normally Closed (N.C.) Contact			
		3) Positive Transitional Contacts			
		4) Negative Transitional Contacts			
		5) Horizontal Short			
		6) Vertical Short			
		7) Coil	• Normal		
		8) Link Coil			
		9) MC Coil	• Latched		
		10) MC Control Coil			
	Timer	1) 1-SECOND TIMER	T1.0		
		2) 0.1-SECOND TIMER	T0.1		
		3) 0.01-SECOND TIMER	T.01		
		4) 0.001-SECOND TIMER	T1MS		
Counters	1) UP COUNTER	UCTR			
	2) DOWN COUNTER	DCTR			
Math Instructions (25 instructions)	Unsigned, 4-digit, Decimal Arithmetic Instructions	1) UNSIGNED SINGLE PRECISION DECIMAL ADDITION	ADD	Refer to 4.2.2	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 2
		2) UNSIGNED SINGLE PRECISION DECIMAL SUBTRACTION	SUB		
		3) UNSIGNED SINGLE PRECISION DECIMAL MULTIPLICATION	MUL		
		4) UNSIGNED SINGLE PRECISION DECIMAL DIVISION	DIV		

Class	Subclass	Name	Symbol	Outline	Details			
Math Instructions (25 instructions)	Unsigned, 8-digit, Decimal Arithmetic Instructions	1) UNSIGNED DOUBLE PRECISION DECIMAL ADDITION	DADD	Refer to 4.2.2	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 2			
		2) UNSIGNED DOUBLE PRECISION DECIMAL SUBTRACTION	DSUB					
		3) UNSIGNED DOUBLE PRECISION DECIMAL MULTIPLICATION	DMUL					
		4) UNSIGNED DOUBLE PRECISION DECIMAL DIVISION	DDIV					
	Signed, 4-digit, Decimal Arithmetic Instructions	1) SIGNED SINGLE PRECISION DECIMAL ADDITION	SADD					
		2) SIGNED SINGLE PRECISION DECIMAL SUBTRACTION	SSUB					
		3) SIGNED SINGLE PRECISION DECIMAL MULTIPLICATION	SMUL					
		4) SIGNED SINGLE PRECISION DECIMAL DIVISION	SDIV					
	Signed, 8-digit, Decimal Arithmetic Instructions	1) SIGNED DOUBLE PRECISION DECIMAL ADDITION	SDAD					
		2) SIGNED DOUBLE PRECISION DECIMAL SUBTRACTION	SDSB					
	Sixteen-bit Arithmetic Instructions	1) 16-BIT ADDITION	AD16					
		2) 16-BIT SUBTRACTION	SU16					
		3) 16-BIT MULTIPLICATION	MU16					
		4) 16-BIT DIVISION	DV16					
	Thirty-two-bit Arithmetic Instructions	1) 32-BIT ADDITION	AD32					
		2) 32-BIT SUBTRACTION	SU32					
		3) 32-BIT COMPARE	TEST					
	Decimal Square Root Instructions	1) SINGLE PRECISION DECIMAL SQUARE ROOT	SQRT					
		2) DOUBLE PRECISION DECIMAL SQUARE ROOT	DSQR					
	Decimal Trigonometric Instructions	1) DECIMAL SINE	SIN					
		2) DECIMAL COSINE	COS					
	Data Manipulation Instructions (48 instructions)	Data Transfer	1) REGISTER-TO-TABLE MOVE			R→T	Refer to 4.2.3	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 3
			2) TABLE-TO-REGISTER MOVE			T→R		
			3) TABLE-TO-TABLE MOVE			T→T		
			4) FIRST IN			FIN		
5) FIRST OUT			FOUT					
6) TABLE SEARCH			SRCH					
7) TABLE SET			TSET					
8) BLOCK MOVE			BLKM					
9) BLOCK-TO-TABLE MOVE			BLKT					
10) TABLE-TO-BLOCK MOVE			TBLK					
11) INDIRECT BLOCK WRITE			IBKW					
12) INDIRECT BLOCK READ			IBKR					



Overview of Instructions

Class	Subclass	Name	Symbol	Outline	Details
Data Manipulation Instructions (48 instructions)	Indexed Block Transfer Instructions	1) DESTINATION INDEXED BLOCK TRANSFER 1	DIBT	Refer to 4.2.4	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 4
		2) DESTINATION INDEXED BLOCK TRANSFER 2	DIBR		
		3) SOURCE INDEXED BLOCK TRANSFER 1	SIBT		
		4) SOURCE INDEXED BLOCK TRANSFER 2	SIBR		
	Matrix	1) LOGICAL AND	AND	Refer to 4.2.5	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 5
		2) LOGICAL OR	OR		
		3) LOGICAL EXCLUSIVE OR	XOR		
		4) LOGICAL COMPLEMENT	COMP		
		5) LOGICAL COMPARE	CMPR		
		6) LOGICAL BIT MODIFY	MBIT		
		7) LOGICAL SENSE	SENS		
		8) LOGICAL BIT ROTATE	BROT		
		9) LOGICAL MULTI-BIT ROTATE	MROT		
		10) LOGICAL BIT COUNT	BCNT		
	Bit Manipulation Instructions	1) NORMALLY OPEN BIT	NOBT	Refer to 4.2.6	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 6
		2) NORMALLY CLOSED BIT	NCBT		
		3) NORMAL BIT	NBIT		
		4) SET BIT	SBIT		
		5) RESET BIT	RBIT		
	Data Conversion Instructions	1) BCD-TO-BINARY CONVERSION	BIN	Refer to 4.2.7	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 7
		2) BINARY-TO-BCD CONVERSION	BCD		
		3) ASCII-TO-BINARY CONVERSION	ATOB		
		4) BINARY-TO-ASCII CONVERSION	BTOA		
		5) 16-BIT CONVERSION	CAST		
		6) 32-BIT CONVERSION	DCST		
	Other Data Manipulation Instructions	1) SET WORD DATA	SDAT	Refer to 4.2.8	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 8
		2) SET DOUBLE WORD DATA	SDDT		
		3) LOGICAL BYTE REARRANGEMENT	TWST		
		4) SWAP	SWAP		
		5) SORT	SORT		
		6) BYTE SPLIT	BYSL		
		7) BYTE COMPOSITION	BYCM		
8) NIBBLE SPLIT		NBSL			
9) NIBBLE COMPOSITION		NBCM			
10) BLOCK ADD		BADD			

Class	Subclass	Name	Symbol	Outline	Details
Diagnostic Instructions (1 instruction)		SYSTEM STATUS MONITORING	STAT	Refer to 4.2.9	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 9
Sequence Control Instructions (2 instructions)		1) Stepping Sequencers	<ul style="list-style-type: none"> <li>• N.O. Contacts —  S  —</li> <li>• N.C. Contacts —  \$  —</li> </ul>	Refer to 4.2.10	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 10
		2) SEQUENCE CONTROL INTERFACE	SCIF		
Program Control Instructions (7 instructions)	Skip Node Instructions	1) SKIP CONSTANT	SKPC	Refer to 4.2.11	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 11
		2) SKIP REGISTER	SKPR		
	Subroutine Instructions	3) SUBROUTINE JUMP	JSR		
		4) SUBROUTINE LABEL	LAB		
		5) SUBROUTINE RETURN	RET		
	Master Control Instructions	6) MASTER CONTROL ON	MSON		
		7) MASTER CONTROL OFF	MSOF		
I/O Control Instructions (2 instructions)		1) DIRECT IN	DIN	Refer to 4.2.12	MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 12
		2) DIRECT OUT	DOUT		
Communications Instructions (10 instructions)	COM	1) COMM	COMM		MEMOCON GL120, GL130 COM Instructions User's Manual
		2) COMR	COMR		
		3) CHECKSUM	CKSM		MEMOCON GL120, GL130 Software User's Manual, Volume 2 Chapter 8
		4) MEMOBUS PLUS COMM	MSTR		MEMOCON GL120, GL130 MEMOBUS PLUS User's Manual
	FBUS	5) FBUS MEMOBUS MESSAGE SEND	MSND		MEMOCON GL120, GL130 PC Link Module User's Manual
		6) FBUS MEMOBUS MESSAGE RECEIVE	MRCV		
		7) FBUS MESSAGE SEND	SEND		
		8) FBUS MESSAGE RECEIVE	RECV		
	ASCII	9) ASCII WRITE	WRIT		(Under development.)
		10) ASCII READ	READ		

Overview of Instructions

Class	Subclass	Name	Symbol	Outline	Details
Motion Control Instructions (22 instructions)		1) MODE SET	MOD		MEMOCON GL120, GL130 Motion Module MC20 Software User's Manual
		2) SERVO ON	SVN		
		3) PROGRAM RUN	MVL		
		4) A AXIS OPERATION	MVA		
		5) B AXIS OPERATION	MVB		
		6) C AXIS OPERATION	MVC		
		7) D AXIS OPERATION	MVD		
		8) HOME RETURN	ZRN		
		9) JOG	JOG		
		10) STEP	STP		
		11) SINGLE BLOCK MODE	SMD		
		12) MACHINE LOCK MODE	MLK		
		13) MODULE RESET	MRS		
		14) MACHINE RESET	RST		
		15) EMERGENCY STOP NOTIFICATION	ESP		
		16) ALARM RESET	ARS		
		17) MONITOR	MON		
		18) COORDINATE SETTING	POS		
		19) PARAMETER SETTING	PRM		
		20) H VARIABLE SETTING	VAR		
		21) POINT TABLE SETTING	PTBL		
		22) HOME POSITION SETTING	ZST		
Expansion Math Instructions (32 instructions)	Logarithmic Calculations	1) LOG	EMTH LOG		MEMOCON GL120, GL130 Software User's Manual, Volume 3
		2) ANTILOG	EMTH ANLOG		
	Floating-point Calculations	3) INTEGER-TO-FLOATING POINT CONVERSION	EMTH CNVIF		
		4) INTEGER PLUS FLOATING POINT	EMTH ADDIF		
		5) INTEGER MINUS FLOATING POINT	EMTH SUBIF		
		6) INTEGER TIMES FLOATING POINT	EMTH MULIF		
		7) INTEGER DIVIDED BY FLOATING POINT	EMTH DIVIF		
		8) FLOATING POINT MINUS INTEGER	EMTH SUBFI		
		9) FLOATING DIVIDED BY MINUS INTEGER	EMTH DIVFI		
		10) INTEGER-FLOATING COMPARISON	EMTH CMPIF		

Class	Subclass	Name	Symbol	Outline	Details				
Expansion Math Instructions (32 instructions)	Floating-point Calculations	11) FLOATING POINT-TO-INTEGGER CONVERSION	EMTH CNVFI		MEMOCON GL120, GL130 Software User's Manual, Volume 3				
		12) FLOATING POINT ADDITIONS	EMTH ADDFP						
		13) FLOATING POINT SUBTRACTION	EMTH SUBFP						
		14) FLOATING POINT MULTIPLICATION	EMTH MULFP						
		15) FLOATING POINT DIVISION	EMTH DIVFP						
		16) FLOATING POINT COMPARISON	EMTH CMPFP						
		17) FLOATING POINT SQUARE ROOT	EMTH SQRFP						
		18) FLOATING POINT CHANGE SIGN	EMTH CHSIN						
		19) LOAD FLOATING POINT $\pi$	EMTH PI						
		20) FLOATING POINT SINE (RADIAL)	EMTH SINE						
		21) FLOATING POINT COSINE (RADIAL)	EMTH COS						
		22) FLOATING POINT TANGENT (RADIAL)	EMTH TAN						
		23) FLOATING POINT ARC SINE (RADIAL)	EMTH ARSIN						
		24) FLOATING POINT ARC COSINE (RADIAL)	EMTH ARCOS						
		25) FLOATING POINT ARC TANGENT (RADIAL)	EMTH ARTAN						
		26) RADIANS-TO-DEGREES CONVERSION	EMTH CONVRD						
		27) DEGREE-TO-RADIANS CONVERSION	EMTH CONVDR						
		28) FLOATING POINT POWER	EMTH POW						
		29) FLOATING POINT EXPONENT	EMTH EXP						
		30) FLOATING POINT NATURAL LOGARITHM	EMTH LNFP						
		31) FLOATING POINT COMMON LOGARITHM	EMTH LOGFP						
		32) FLOATING POINT CALCULATION ERROR READ	EMTH ERLOG						
		Process Control Instructions (1 instruction)				PID2	PID2		MEMOCON GL120, GL130 Software User's Manual

## 4.2 Instruction Outlines

■ This section provides outlines of GL120/GL130 instructions.

4.2.1	Basic Instructions .....	4-9
4.2.2	Math Instructions .....	4-14
4.2.3	Data Transfer Instructions .....	4-41
4.2.4	Indexed Block Transfer Instructions .....	4-55
4.2.5	Matrix Instructions .....	4-60
4.2.6	Bit Manipulation Instructions .....	4-72
4.2.7	Data Conversion Instructions .....	4-76
4.2.8	Other Data Manipulation Instructions .....	4-81
4.2.9	System Status Monitoring Instruction .....	4-91
4.2.10	Sequencers .....	4-92
4.2.11	Program Control Instructions .....	4-94
4.2.12	I/O Control Instructions .....	4-101

### • Reference Numbers

The ranges of reference numbers that can be used with each instruction are provided in this manual under the heading *Structural Elements*, as shown in the following example. These reference numbers are specified as follows:

- 1) The ranges listed in the tables are for initial values.
  
- 2) Two systems are used for reference numbers for coils, input relays, input registers, holding registers, and constant registers: Reference numbers beginning with numbers are called numeric reference numbers and those beginning with letters are called lettered reference numbers. In the *Structural Elements* tables, the lettered reference numbers are given in parentheses.



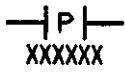
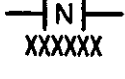
## Structural Elements of REGISTER-TO-TABLE MOVE (R→T)

Element	Meaning	Settings
Top (S)	Source reference number	Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (P)	Pointer reference number	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 or R20001 to R21023
Bottom (Z)	Destination table size	Constant: #00001 to #00999

## 4.2.1 Basic Instructions

## 1. Relays

Table 4.2 Relay Elements

Name	Structure	Function	Reference No.
Normally Open (N.O.) Contact		Power is passed from left to right while the corresponding reference is ON.	Reference numbers for one of the following types of relay are specified for "XXXXXX".
Normally Closed (N.C.) Contact		Power is passed from left to right while the corresponding reference is OFF.	1) Coils 2) Link coils 3) MC coils
Positive Transitional Contacts		Power is passed from left to right for one scan each time the corresponding reference goes from OFF to ON.	4) MC control coils 5) Input relays
Negative Transitional Contacts		Power is passed from left to right for one scan each time the corresponding reference goes from ON to OFF.	6) MC relays 7) MC control relays 8) M code relays

Name	Structure	Function	Reference No.	
Horizontal Short	—————	Shorts adjacent columns and passes power from left to right.	None (not necessary)	
Vertical Short		Shorts adjacent rows and passes power from top to bottom or from bottom to top.		
Coil	1) Normal — ( ) — XXXXXX	1) Output coils send the ON/OFF status to Digital Output Modules.  2) Internal coils are used only to build logic in the ladder logic program.	000001 to 008192 (O00001 to O08192)	
	2) Latched — (L) — XXXXXX	3) Battery monitor coils are used to monitor the output voltage of the memory backup battery built into the CPU Module.		
	Link Coil	Transfers ON/OFF data to/from another PLC in a PC link.		D10001 to D11024 D20001 to D21024
	MC Coil	Outputs ON/OFF data to a Four-axis Motion Module (MC20).		Y10001 to Y10256 Y20001 to Y20256
MC Control Coil		Outputs overrides or MFIN signals to a Four-axis Motion Module (MC20).	Q10001 to Q10160 Q20001 to Q20160	

**Note** "XXXXXX" represents the reference number of a coil or relay.

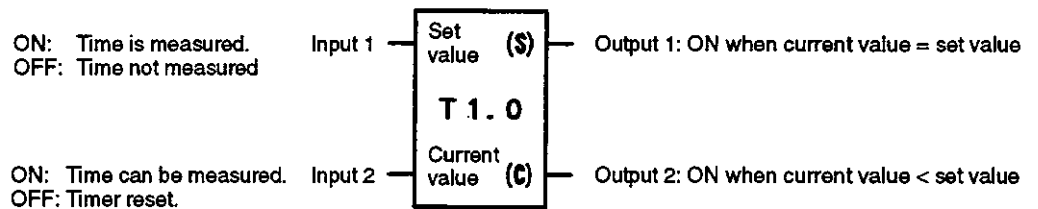
## 2. Timers and Counters

### A. 1-SECOND TIMER (T1.0)

#### 1) Function

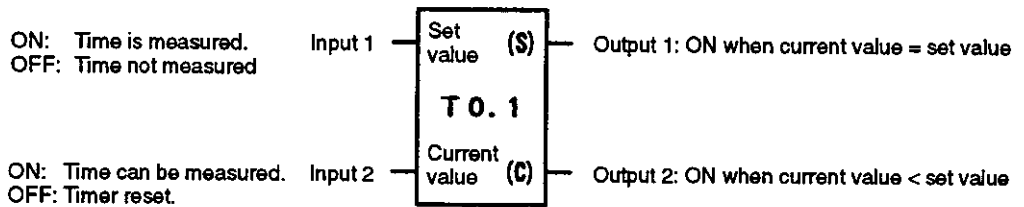
1-SECOND TIMER is used to measure time in 1-s intervals between 1 and 65,535 s.

#### 2) Structure

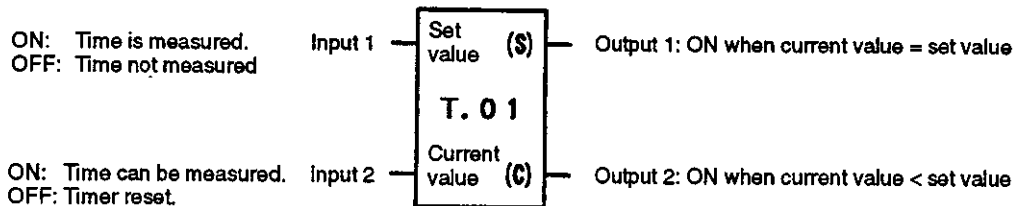


**B. 0.1-SECOND TIMER (T0.1)****1) Function**

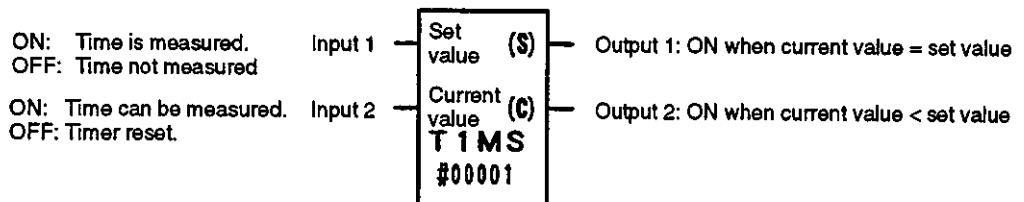
0.1-SECOND TIMER is used to measure time in 0.1-s intervals between 0.1 and 6,553.5 s.

**2) Structure****C. 0.01-SECOND TIMER (T.01)****1) Function**

0.01-SECOND TIMER is used to measure time in 0.01-s intervals between 0.01 and 655.35 s.

**2) Structure****D. 0.001-SECOND TIMER (T1MS)****1) Function**

0.001-SECOND TIMER is used to measure time in 0.001-s intervals between 0.001 and 65.535 s.

**2) Structure**

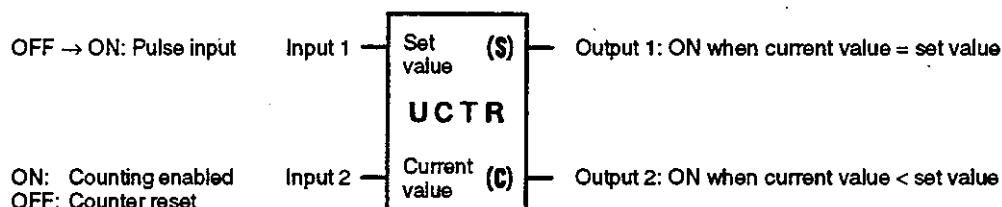


**E. UP COUNTER (UCTR)**

**1) Function**

UP COUNTER counts pulses and increments the current value one at a time. The counting range is between 1 and 65,535.

**2) Structure**

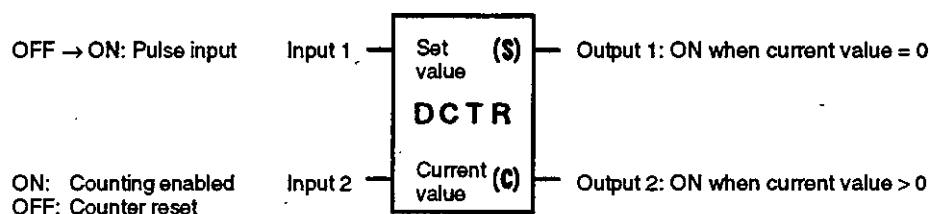


**F. DOWN COUNTER (DCTR)**

**1) Function**

DOWN COUNTER counts pulses and decrements the current value one at a time. The counting range is between 1 and 65,535.

**2) Structure**



**3) Structural Elements**

**Table 4.3 Structural Elements of T1.0, T0.1 and T.01**

Element	Meaning	Possible Settings
Top (S)	The value of the constant or the contents of the register with the specified reference number will be used as the set value (S) of the timer.  The value of S must be between 0 and 65,535.	Constant: #00000 to #65535 Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Bottom (C)	The current value of the timer (C) will be stored in the specified register.  The value of C will be between 0 and the set value.	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 R20001 to R21024

Table 4.4 Structural Elements of T1MS

Element	Meaning	Possible Settings
Top (S)	The value of the constant or the contents of the register with the specified reference number will be used as the set value (S) of the timer.  The value of S must be between 0 and 65,535.	Constant: #00000 to #65535 Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Middle (C)	The current value of the timer (C) will be stored in the specified register.  The value of C will be between 0 and the set value.	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 R20001 to R21024
Bottom	—	Constant: #00001

Table 4.5 Structural Elements of UCTR and DCTR

Element	Meaning	Possible Settings
Top (S)	The value of the constant or the contents of the register with the specified reference number will be used as the set value (S) of the counter.  The value of S must be between 0 and 65,535.	Constant: #00000 to #65535 Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Bottom (C)	The current value of the counter (C) will be stored in the specified register.  The value of C will be between 0 and the set value.	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 R20001 to R21024

## 4.2.2 Math Instructions

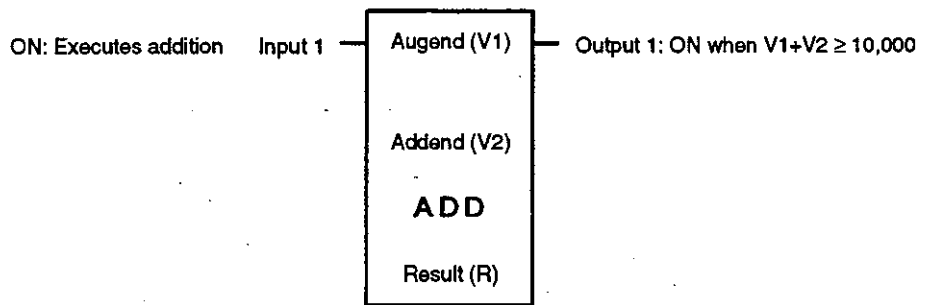
### 1. Unsigned, Four-digit, Decimal Arithmetic Instructions

#### A. UNSIGNED SINGLE PRECISION DECIMAL ADDITION (ADD)

##### 1) Function

Unsigned addition is performed between two 4-digit decimal numbers, V1 and V2.

##### 2) Structure



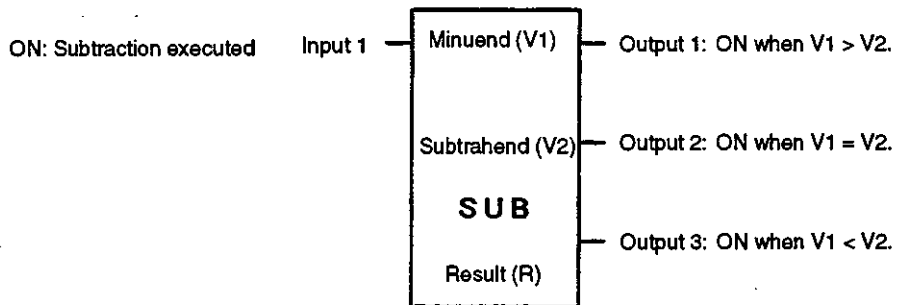
#### B. UNSIGNED SINGLE PRECISION DECIMAL SUBTRACTION (SUB)

##### 1) Function

a) Unsigned subtraction is performed between two 4-digit decimal numbers, V1 and V2.

b) The sizes of V1 and V2 are compared.

##### 2) Structure

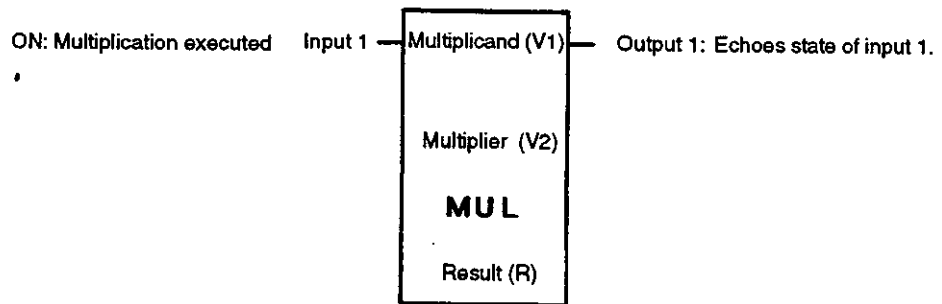


### C. UNSIGNED SINGLE PRECISION DECIMAL MULTIPLICATION (MUL)

#### 1) Function

Unsigned multiplication is performed between two 4-digit decimal numbers, V1 and V2.

#### 2) Structure



**Table 4.6 Structural Elements of ADD and SUB**

Element	Meaning	Possible Settings
Top (V1)	Either the value of the constant or the contents of the register is used as the augend, V1. V1 must be between 0 and 9,999.	Constant: #00000 to #09999 Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999)
Middle (V2)	Either the value of the constant or the contents of the register is used as the addend, V2. V2 must be between 0 and 9,999.	Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Bottom (R)	The result is stored in the register. The result must be between 0 and 9,999.	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 R20001 to R21024

Table 4.7 Structural Elements of MUL

Element	Meaning	Possible Settings				
Top (V1)	Either the value of the constant or the contents of the register is used as the multiplicand, V1. V1 must be between 0 and 9,999.	Constant: #00000 to #09999 Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999)				
Middle (V2)	Either the value of the constant or the contents of the register is used as the multiplier, V2. V2 must be between 0 and 9,999.	Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024				
Bottom (R)	The result is stored in registers as shown below. The result must be between 0 and 99,980,001. <table border="1" style="margin-left: 40px;"> <tr> <td>R</td> <td>Upper 4 digits of result</td> </tr> <tr> <td>R+1</td> <td>Lower 4 digits of result</td> </tr> </table>	R	Upper 4 digits of result	R+1	Lower 4 digits of result	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 R20001 to R21023
R	Upper 4 digits of result					
R+1	Lower 4 digits of result					

**D. UNSIGNED SINGLE PRECISION DECIMAL DIVISION (DIV)**

**1) Function**

Unsigned division is performed between a 4-digit or 8-digit decimal number V1 and a 4-digit decimal number V2 (V1 ÷ V2).

**2) Structure**

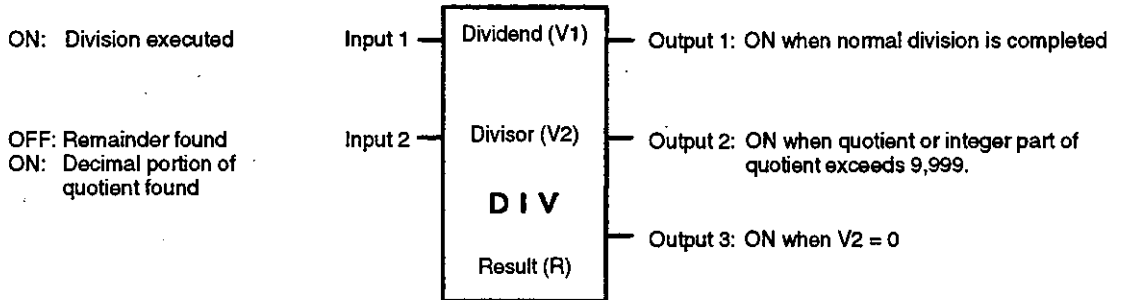


Table 4.8 Structural Elements of DIV

Element	Meaning	Possible Settings								
Top (V1)	<p>1) If a constant is specified, its value is used as the dividend, V1. In this case, the value of V1 must be between 0 and 9,999.</p> <p>2) If a reference number is specified, the contents of two consecutive registers is used as the dividend, V1, as shown in the following example. The value of V1 must be between 0 and 99,989,999.</p> <p>In the example, "400001" was specified for the top element.</p> <table border="1" style="margin-left: 40px;"> <tr> <td style="padding-right: 10px;">400001</td> <td>Upper 4 digits</td> </tr> <tr> <td style="padding-right: 10px;">400002</td> <td>Lower 4 digits</td> </tr> </table>	400001	Upper 4 digits	400002	Lower 4 digits	<p>Constant: #00000 to #09999</p> <p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>				
400001	Upper 4 digits									
400002	Lower 4 digits									
Middle (V2)	<p>1) If a constant is specified, its value is used as the divisor, V2. In this case, the value of V2 must be between 0 and 9,999.</p> <p>2) If a reference number is specified, the contents of the register is used as the divisor, V2. In this case, the value of V2 must also be between 0 and 9,999.</p>	<p>Constant: #00000 to #09999</p> <p>Input register: 300001 to 300512 (Z00001 to Z00512)</p> <p>Holding register: 400001 to 409999 (W00001 to W09999)</p> <p>Constant register: 700001 to 704096 (K00001 to K04096)</p> <p>Link register: R10001 to R11024 R20001 to R21024</p>								
Bottom (R)	<p>The result is stored in registers as shown below. The result must be between 0 and 9,999.</p> <p>1) <b>Input 2 OFF</b></p> <table border="1" style="margin-left: 40px;"> <tr> <td style="padding-right: 10px;">R</td> <td>Quotient</td> </tr> <tr> <td style="padding-right: 10px;">R+1</td> <td>Remainder</td> </tr> </table> <p>2) <b>Input 2 ON</b></p> <table border="1" style="margin-left: 40px;"> <tr> <td style="padding-right: 10px;">R</td> <td>Integer portion of quotient</td> </tr> <tr> <td style="padding-right: 10px;">R+1</td> <td>Decimal portion of quotient</td> </tr> </table>	R	Quotient	R+1	Remainder	R	Integer portion of quotient	R+1	Decimal portion of quotient	<p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
R	Quotient									
R+1	Remainder									
R	Integer portion of quotient									
R+1	Decimal portion of quotient									

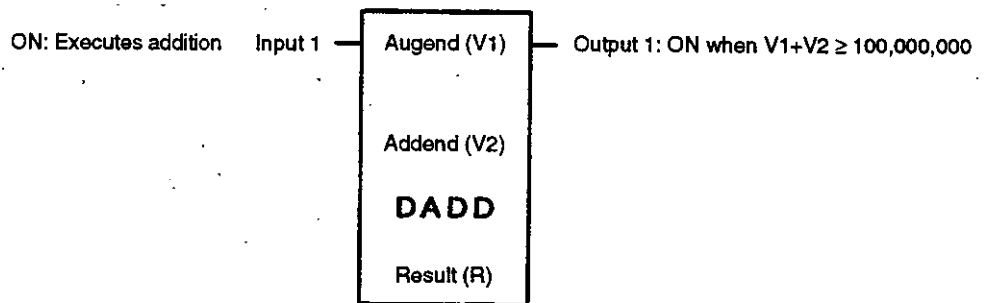
## 2. Unsigned, Eight-digit, Decimal Arithmetic Instructions

### A. UNSIGNED DOUBLE PRECISION DECIMAL ADDITION (DADD)

#### 1) Function

Unsigned addition is performed between two 8-digit decimal numbers, V1 and V2.

#### 2) Structure



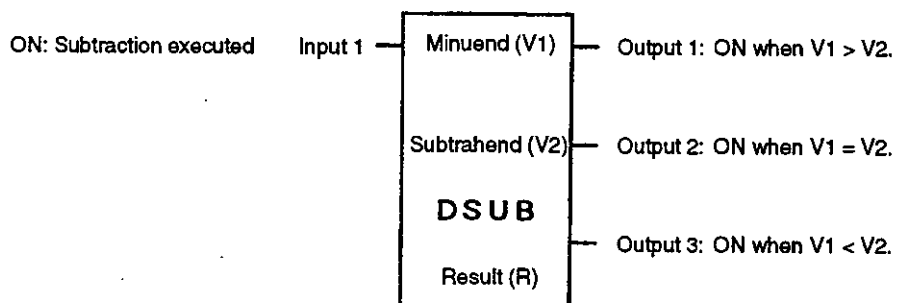
### B. UNSIGNED DOUBLE PRECISION DECIMAL SUBTRACTION (DSUB)

#### 1) Function

a) Unsigned subtraction is performed between two 8-digit decimal numbers, V1 and V2.

b) The sizes of V1 and V2 are compared.

#### 2) Structure

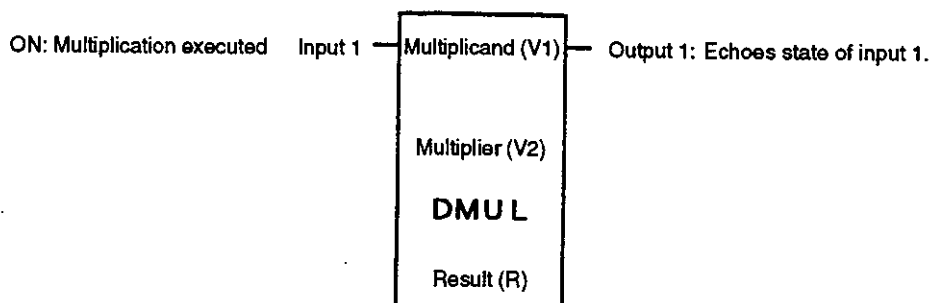


### C. UNSIGNED DOUBLE PRECISION DECIMAL MULTIPLICATION (DMUL)

#### 1) Function

Unsigned multiplication is performed between two 8-digit decimal numbers, V1 and V2.

#### 2) Structure



**Table 4.9 Structural Elements of DADD and DSUB**

Element	Meaning	Possible Settings				
Top (V1)	<p>The contents of two consecutive registers is used as the augend, V1, as shown in the following example. The value of V1 must be between 0 and 99,999,999.</p> <p>In the example, "400001" was specified for the top element.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: center;">400001</td> <td style="text-align: center;">Upper 4 digits</td> </tr> <tr> <td style="text-align: center;">400002</td> <td style="text-align: center;">Lower 4 digits</td> </tr> </table>	400001	Upper 4 digits	400002	Lower 4 digits	<p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400001	Upper 4 digits					
400002	Lower 4 digits					
Middle (V2)	<p>The contents of two consecutive registers is used as the addend, V2 as shown in the following example. The value of V2 must be between 0 and 99,999,999.</p> <p>In the example, "400003" was specified for the middle element.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: center;">400003</td> <td style="text-align: center;">Upper 4 digits</td> </tr> <tr> <td style="text-align: center;">400004</td> <td style="text-align: center;">Lower 4 digits</td> </tr> </table>	400003	Upper 4 digits	400004	Lower 4 digits	
400003	Upper 4 digits					
400004	Lower 4 digits					
Bottom (R)	<p>The result is stored in registers as shown below. The result must be between 0 and 99,999,999.</p> <p>In the example, "400005" was specified for the bottom element.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: center;">400005</td> <td style="text-align: center;">Upper 4 digits</td> </tr> <tr> <td style="text-align: center;">400006</td> <td style="text-align: center;">Lower 4 digits</td> </tr> </table>	400005	Upper 4 digits	400006	Lower 4 digits	<p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400005	Upper 4 digits					
400006	Lower 4 digits					

**Table 4.10 Structural Elements of DMUL**



Element	Meaning	Possible Settings								
Top (V1)	<p>The contents of two consecutive registers is used as the multiplicand, V1, as shown in the following example. The value of V1 must be between 0 and 99,999,999.</p> <p>In the example, "400001" was specified for the top element.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: right;">400001</td> <td>Upper 4 digits</td> </tr> <tr> <td style="text-align: right;">400002</td> <td>Lower 4 digits</td> </tr> </table>	400001	Upper 4 digits	400002	Lower 4 digits	<p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>				
400001	Upper 4 digits									
400002	Lower 4 digits									
Middle (V2)	<p>The contents of two consecutive registers is used as the multiplier, V2, as shown in the following example. The value of V2 must be between 0 and 99,999,999.</p> <p>In the example, "400003" was specified for the middle element.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: right;">400003</td> <td>Upper 4 digits</td> </tr> <tr> <td style="text-align: right;">400004</td> <td>Lower 4 digits</td> </tr> </table>	400003	Upper 4 digits	400004	Lower 4 digits					
400003	Upper 4 digits									
400004	Lower 4 digits									
Bottom (R)	<p>The result is stored in registers as shown below. The result must be between 0 and 9,999,999,800,000,001.</p> <p>In the example, "400005" was specified for the bottom element.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: right;">400005</td> <td>Most upper 4 digits</td> </tr> <tr> <td style="text-align: right;">400006</td> <td>Upper 4 digits</td> </tr> <tr> <td style="text-align: right;">400007</td> <td>Lower 4 digits</td> </tr> <tr> <td style="text-align: right;">400008</td> <td>Most lower 4 digits</td> </tr> </table>	400005	Most upper 4 digits	400006	Upper 4 digits	400007	Lower 4 digits	400008	Most lower 4 digits	<p>Holding register: 400001 to 409996 (W00001 to W09996)</p> <p>Link register: R10001 to R11021 R20001 to R21021</p>
400005	Most upper 4 digits									
400006	Upper 4 digits									
400007	Lower 4 digits									
400008	Most lower 4 digits									

**D. UNSIGNED DOUBLE PRECISION DECIMAL DIVISION (DDIV)**

**1) Function**

Unsigned division is performed between a 16-digit decimal number V1 and a 8-digit decimal number V2 ( $V1 \div V2$ ).

**2) Structure**

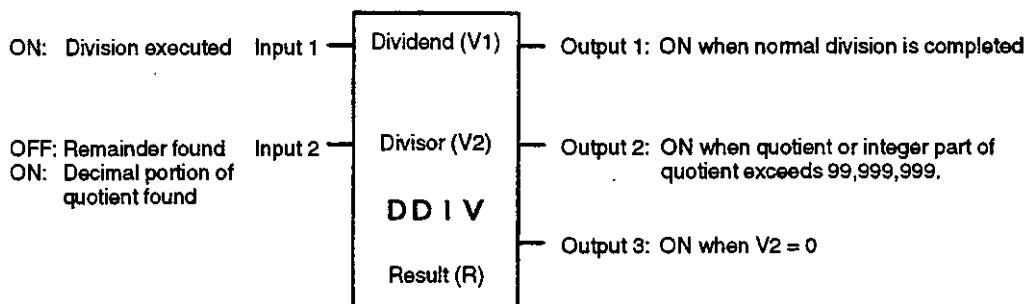


Table 4.11 Structural Elements of DDIV

Element	Meaning	Possible Settings																
Top (V1)	<p>The contents of four consecutive registers is used as the dividend, V1, as shown in the following example. The value of V1 must be between 0 and 9,999,999,899,999,999.</p> <p>In the example, "400001" was specified for the top element.</p> <table border="1"> <tr> <td>400001</td> <td>Most upper 4 digits</td> </tr> <tr> <td>400002</td> <td>Upper 4 digits</td> </tr> <tr> <td>400003</td> <td>Lower 4 digits</td> </tr> <tr> <td>400004</td> <td>Most lower 4 digits</td> </tr> </table>	400001	Most upper 4 digits	400002	Upper 4 digits	400003	Lower 4 digits	400004	Most lower 4 digits	<p>Input register: 300001 to 300509 (Z00001 to Z00509)</p> <p>Holding register: 400001 to 409996 (W00001 to W09996)</p> <p>Constant register: 700001 to 704093 (K00001 to K04093)</p> <p>Link register: R10001 to R11021 R20001 to R21021</p>								
400001	Most upper 4 digits																	
400002	Upper 4 digits																	
400003	Lower 4 digits																	
400004	Most lower 4 digits																	
Middle (V2)	<p>The contents of two consecutive registers is used as the divisor, V2, as shown in the following example. The value of V2 must be between 0 and 99,999,999.</p> <p>In the example, "400005" was specified for the middle element.</p> <table border="1"> <tr> <td>400005</td> <td>Upper 4 digits</td> </tr> <tr> <td>400006</td> <td>Lower 4 digits</td> </tr> </table>	400005	Upper 4 digits	400006	Lower 4 digits	<p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>												
400005	Upper 4 digits																	
400006	Lower 4 digits																	
Bottom (R)	<p>The result is stored in registers as shown below. The result must be between 0 and 99,999,999.</p> <p>In the example, "400007" was specified for the bottom element.</p> <p><b>Input 2 OFF</b></p> <table border="1"> <tr> <td>400007</td> <td>Upper 4 digits of quotient</td> </tr> <tr> <td>400008</td> <td>Lower 4 digits of quotient</td> </tr> <tr> <td>400009</td> <td>Upper 4 digits of remainder</td> </tr> <tr> <td>400010</td> <td>Lower 4 digits of remainder</td> </tr> </table> <p><b>Input 2 ON</b></p> <table border="1"> <tr> <td>400007</td> <td>Upper 4 digits of integer portion</td> </tr> <tr> <td>400008</td> <td>Lower 4 digits of integer portion</td> </tr> <tr> <td>400009</td> <td>Upper 4 digits of decimal portion</td> </tr> <tr> <td>400010</td> <td>Lower 4 digits of decimal portion</td> </tr> </table>	400007	Upper 4 digits of quotient	400008	Lower 4 digits of quotient	400009	Upper 4 digits of remainder	400010	Lower 4 digits of remainder	400007	Upper 4 digits of integer portion	400008	Lower 4 digits of integer portion	400009	Upper 4 digits of decimal portion	400010	Lower 4 digits of decimal portion	<p>Holding register: 400001 to 409996 (W00001 to W09996)</p> <p>Link register: R10001 to R11021 R20001 to R21021</p>
400007	Upper 4 digits of quotient																	
400008	Lower 4 digits of quotient																	
400009	Upper 4 digits of remainder																	
400010	Lower 4 digits of remainder																	
400007	Upper 4 digits of integer portion																	
400008	Lower 4 digits of integer portion																	
400009	Upper 4 digits of decimal portion																	
400010	Lower 4 digits of decimal portion																	

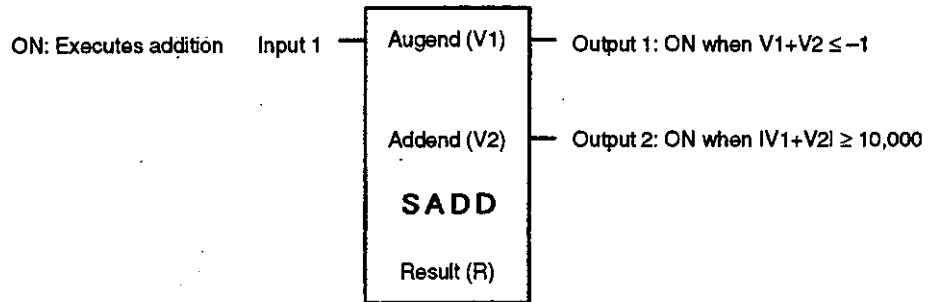
### 3. Signed, Four-digit, Decimal Arithmetic Instructions

#### A. SIGNED SINGLE PRECISION DECIMAL ADDITION (SADD)

##### 1) Function

Signed addition is performed between two 4-digit decimal numbers, V1 and V2.

##### 2) Structure

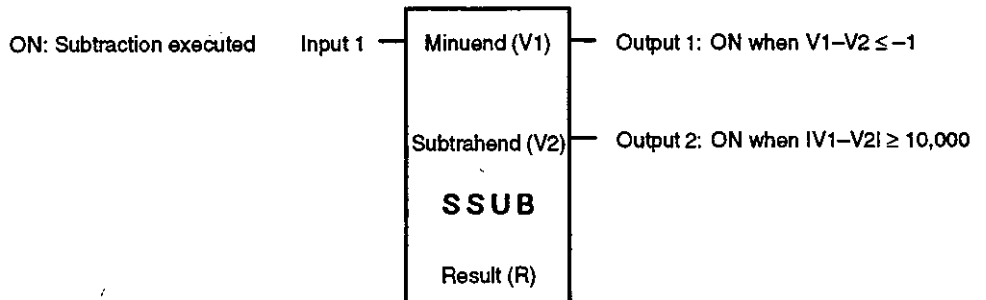


#### B. SIGNED SINGLE PRECISION DECIMAL SUBTRACTION (SSUB)

##### 1) Function

Signed subtraction is performed between two 4-digit decimal numbers, V1 and V2.

##### 2) Structure

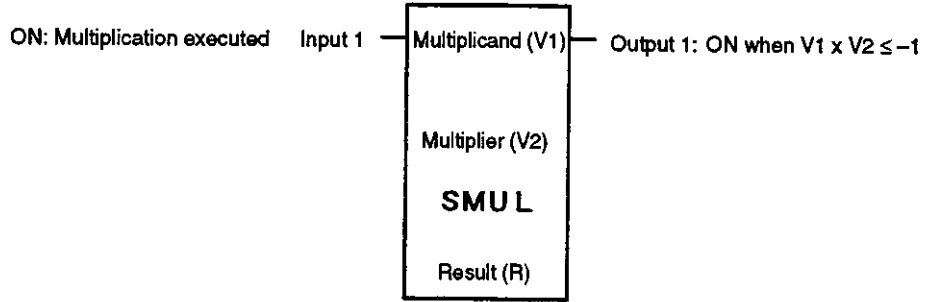


### C. SIGNED SINGLE PRECISION DECIMAL MULTIPLICATION (SMUL)

#### 1) Function

Signed multiplication is performed between two 4-digit decimal numbers, V1 and V2.

#### 2) Structure



**Table 4.12 Structural Elements of SADD and SSUB**

Element	Meaning	Possible Settings
Top (V1)	<ol style="list-style-type: none"> <li>1) The contents of the specified register is used as the augend, V1.</li> <li>2) V1 must be between <math>-9,999</math> and <math>9,999</math>.</li> <li>3) Set the MSB of the register to 1 if <math>V1 &lt; 0</math>.</li> </ol>	Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096)
Middle (V2)	<ol style="list-style-type: none"> <li>1) The contents of the specified register is used as the addend, V2.</li> <li>2) V2 must be between <math>-9,999</math> and <math>9,999</math>.</li> <li>3) Set the MSB of the register to 1 if <math>V2 &lt; 0</math>.</li> </ol>	Link register: R10001 to R11024 R20001 to R21024
Bottom (R)	<ol style="list-style-type: none"> <li>1) The result is stored in the specified register.</li> <li>2) The result must be between <math>-9,999</math> and <math>9,999</math>.</li> <li>3) The MSB of the register will be set to 1 if the results <math>&lt; 0</math>.</li> </ol>	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 R20001 to R21024

Table 4.13 Structural Elements of SMUL

Element	Meaning	Possible Settings				
Top (V1)	1) The contents of the register is used as the multiplicand, V1. 2) V1 must be between -9,999 and 9,999. 3) Set the MSB of the register to 1 if $V1 < 0$ .	Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096)				
Middle (V2)	1) The contents of the register is used as the multiplier, V2. 2) V2 must be between -9,999 and 9,999. 3) Set the MSB of the register to 1 if $V2 < 0$ .	Link register: R10001 to R11024 R20001 to R21024				
Bottom (R)	1) The result is stored in the specified register and the next register as shown below. 2) The result must be between -99,980,001 and 99,980,001. 3) The MSB of the register will be set to 1 if the result $< 0$ . 4) In the example, "400003" was specified for the bottom element. The MSB of 400003 will be set to 1 if the result is negative.	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 R20001 to R21023				
	<table border="1" style="margin-left: 20px;"> <tr> <td style="padding: 2px;">400003</td> <td style="padding: 2px;">Upper 4 digits of result</td> </tr> <tr> <td style="padding: 2px;">400004</td> <td style="padding: 2px;">Lower 4 digits of result</td> </tr> </table>	400003	Upper 4 digits of result	400004	Lower 4 digits of result	
400003	Upper 4 digits of result					
400004	Lower 4 digits of result					

**D. SIGNED SINGLE PRECISION DECIMAL DIVISION (SDIV)**

**1) Function**

Signed division is performed between a 8-digit decimal number V1 and a 4-digit decimal number V2 ( $V1 \div V2$ ).

**2) Structure**

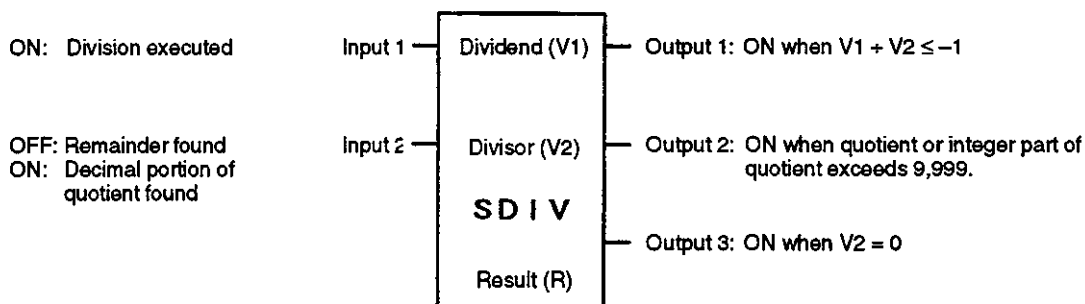


Table 4.14 Structural Elements of SDIV

Element	Meaning	Possible Settings								
Top (V1)	<p>1) The contents of two consecutive registers is used as the dividend, V1, as shown in the following example.</p> <p>2) V1 must be between -99,989,999 and 99,989,999.</p> <p>3) Set the MSB of the registers to 1 if <math>V1 &lt; 0</math>.</p> <p>4) In the example, "400001" was specified for the top element.</p> <p>400001 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Upper 4 digits</td></tr><tr><td>Lower 4 digits</td></tr></table> Negative: Set MSB of 400001 to 1.</p>	Upper 4 digits	Lower 4 digits	<p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>						
Upper 4 digits										
Lower 4 digits										
Middle (V2)	<p>1) The contents of the register is used as the divisor, V2.</p> <p>2) V2 must be between -9,999 and 9,999.</p> <p>3) Set the MSB of the register to 1 if <math>V2 &lt; 0</math>.</p>	<p>Input register: 300001 to 300512 (Z00001 to Z00512)</p> <p>Holding register: 400001 to 409999 (W00001 to W09999)</p> <p>Constant register: 700001 to 704096 (K00001 to K04096)</p> <p>Link register: R10001 to R11024 R20001 to R21024</p>								
Bottom (R)	<p>1) The result is stored in the specified register and the next register as shown below.</p> <p>2) The quotient, remainder, and integer portion of the quotient must be between -9,999 and 9,999. The decimal portion of the quotient must be between 0 and 9,999.</p> <p>R <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Quotient</td></tr></table></p> <p>R+1 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Remainder</td></tr></table></p> <p>or</p> <p>R <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Integer portion of quotient</td></tr></table></p> <p>R+1 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Decimal portion of quotient</td></tr></table></p> <p>3) The MSB of R will be set to 1 if the results <math>&lt; 0</math>.</p> <p>4) In the example, "400004" was specified for the bottom element.</p> <p>400004 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Quotient</td></tr><tr><td>Remainder</td></tr></table> Negative: MSB of 400004 and 400005 set to 1</p> <p>400005</p> <p>or</p> <p>400004 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Integer portion of quotient</td></tr><tr><td>Decimal portion of quotient</td></tr></table> Negative: MSB of 400004 set to 1</p> <p>400005</p>	Quotient	Remainder	Integer portion of quotient	Decimal portion of quotient	Quotient	Remainder	Integer portion of quotient	Decimal portion of quotient	<p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
Quotient										
Remainder										
Integer portion of quotient										
Decimal portion of quotient										
Quotient										
Remainder										
Integer portion of quotient										
Decimal portion of quotient										

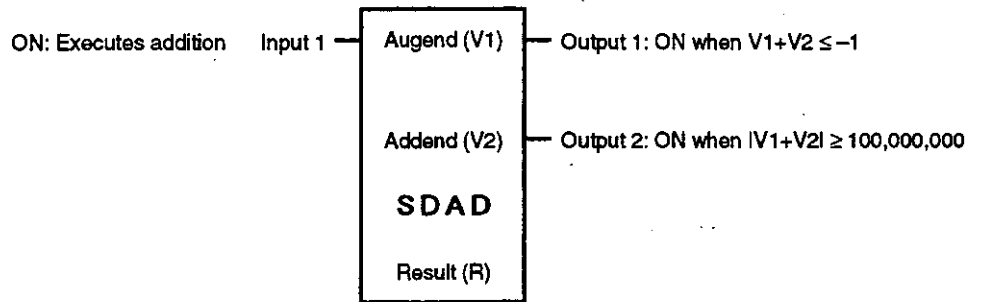
#### 4. Signed, Eight-digit, Decimal Arithmetic Instructions

##### A. SIGNED DOUBLE PRECISION DECIMAL ADDITION (SDAD)

###### 1) Function

Signed addition is performed between two 8-digit decimal numbers, V1 and V2.

###### 2) Structure



##### B. SIGNED DOUBLE PRECISION DECIMAL SUBTRACTION (SDSB)

###### 1) Function

Signed subtraction is performed between two 8-digit decimal numbers, V1 and V2.

###### 2) Structure

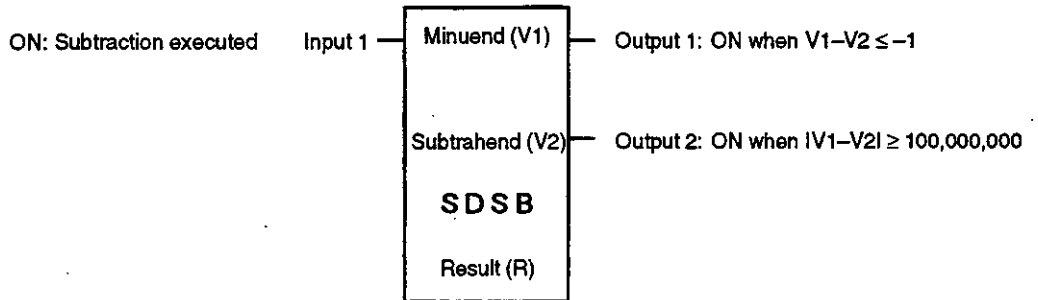


Table 4.15 Structural Elements of SDAD and SDSB

Element	Meaning	Possible Settings				
Top (V1)	<p>1) The contents of two consecutive registers is used as the minuend, V1, as shown in the following example.</p> <p>2) The value of V1 must be between -99,999,999 and 99,999,999.</p> <p>3) If <math>V1 &lt; 0</math>, set the MSB to 1.</p> <p>4) In the example, "400001" was specified for the top element.</p> <p>400001 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Upper 4 digits</td></tr><tr><td>Lower 4 digits</td></tr></table> Negative: Set MSB of 400001 to 1.</p> <p>400002 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Upper 4 digits</td></tr><tr><td>Lower 4 digits</td></tr></table></p>	Upper 4 digits	Lower 4 digits	Upper 4 digits	Lower 4 digits	<p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
Upper 4 digits						
Lower 4 digits						
Upper 4 digits						
Lower 4 digits						
Middle (V2)	<p>1) The contents of two consecutive registers is used as the subtrahend, V2, as shown in the following example.</p> <p>2) The value of V2 must be between -99,999,999 and 99,999,999.</p> <p>3) If <math>V2 &lt; 0</math>, set the MSB to 1.</p> <p>4) In the example, "400003" was specified for the middle element.</p> <p>400003 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Upper 4 digits</td></tr><tr><td>Lower 4 digits</td></tr></table> Negative: Set MSB of 400003 to 1.</p> <p>400004 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Upper 4 digits</td></tr><tr><td>Lower 4 digits</td></tr></table></p>	Upper 4 digits	Lower 4 digits	Upper 4 digits	Lower 4 digits	
Upper 4 digits						
Lower 4 digits						
Upper 4 digits						
Lower 4 digits						
Bottom (R)	<p>1) The result is stored in two consecutive registers, as shown in the following example.</p> <p>2) The value of the result must be between -99,999,999 and 99,999,999.</p> <p>3) If <math>result &lt; 0</math>, the MSB of R is set to 1.</p> <p>4) In the example, "400005" was specified for the bottom element.</p> <p>400005 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Upper 4 digits</td></tr><tr><td>Lower 4 digits</td></tr></table> Negative: MSB of 400005 set to 1.</p> <p>400006 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Upper 4 digits</td></tr><tr><td>Lower 4 digits</td></tr></table></p>	Upper 4 digits	Lower 4 digits	Upper 4 digits	Lower 4 digits	<p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
Upper 4 digits						
Lower 4 digits						
Upper 4 digits						
Lower 4 digits						



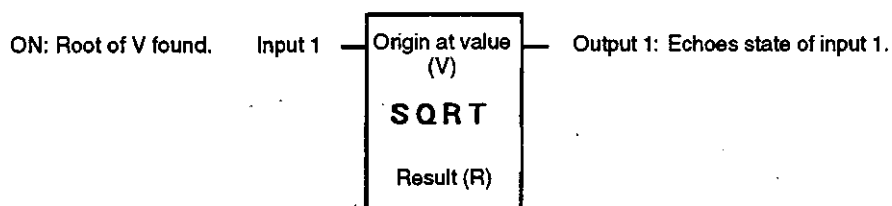
## 5. Decimal Square Root Instructions

### A. SINGLE PRECISION DECIMAL SQUARE ROOT (SQRT)

#### 1) Function

The square root of a 4-digit decimal value, V, is found.

#### 2) Structure



**Table 4.16 Structural Elements of SQRT**

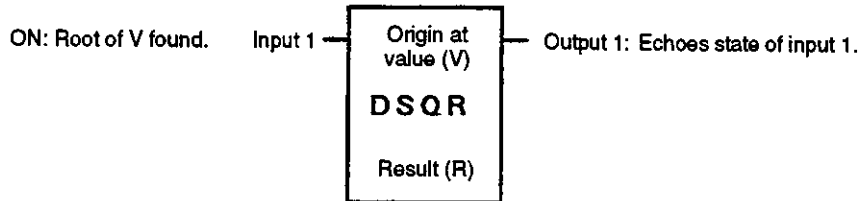
Element	Meaning	Possible Settings
Top (V)	The contents of the specified register is used as V. V must be between 0 and 9,999.	Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Bottom (R)	The square root of V is stored in registers as shown below. The decimal portion is truncated after the 4th decimal place. In the example, "400002" was specified for the bottom element.  400002    Integer portion of result 400003    Decimal portion of result	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 R20001 to R21023

**B. DOUBLE PRECISION DECIMAL SQUARE ROOT (DSQR)**

**1) Function**

The square root of an 8-digit decimal value, V, is found.

**2) Structure**



**Table 4.17 Structural Elements of DSQR**

Element	Meaning	Possible Settings				
Top (V)	<p>The contents of the specified register and the next register is used as V as shown below.</p> <p>V must be between 0 and 99,999,999.</p> <p>In the example, "400001" was specified for the top element.</p> <table border="1" style="margin-left: 20px;"> <tr> <td>400001</td> <td>Upper 4 digits</td> </tr> <tr> <td>400002</td> <td>Lower 4 digits</td> </tr> </table>	400001	Upper 4 digits	400002	Lower 4 digits	<p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400001	Upper 4 digits					
400002	Lower 4 digits					
Bottom (R)	<p>The square root of V is stored in registers as shown below. The decimal portion is truncated after the 4th decimal place.</p> <p>In the example, "400003" was specified for the bottom element.</p> <table border="1" style="margin-left: 20px;"> <tr> <td>400003</td> <td>Integer portion of result</td> </tr> <tr> <td>400004</td> <td>Decimal portion of result</td> </tr> </table>	400003	Integer portion of result	400004	Decimal portion of result	<p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400003	Integer portion of result					
400004	Decimal portion of result					

## 6. Decimal Trigonometric Instruction

### A. DECIMAL SINE (SIN)

#### 1) Function

- The sine of an angle,  $\theta$ , between  $0^\circ$  and  $360^\circ$  is found.

#### 2) Structure

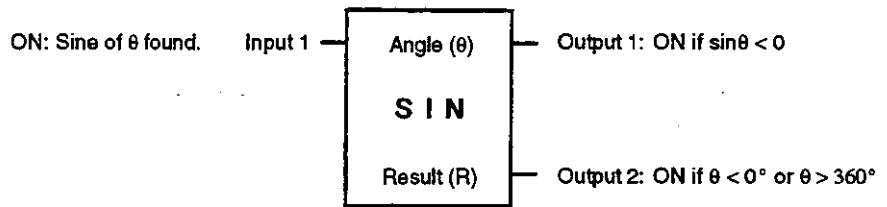
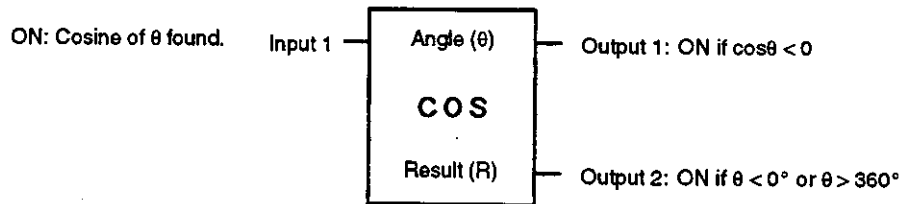


Table 4.18 Structural Elements of SIN

Element	Meaning	Possible Settings				
Top ( $\theta$ )	<p>The contents of the specified register and the next register is used as <math>\theta</math> as shown below.</p> <p><math>\theta</math> must be between <math>0.0000^\circ</math> and <math>360.0000^\circ</math>.</p> <p>In the example, "400001" was specified for the top element.</p> <table border="1" style="margin-left: 40px;"> <tr> <td>400001</td> <td>Integer portion of <math>\theta</math></td> </tr> <tr> <td>400002</td> <td>Decimal portion of <math>\theta</math></td> </tr> </table>	400001	Integer portion of $\theta$	400002	Decimal portion of $\theta$	<p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400001	Integer portion of $\theta$					
400002	Decimal portion of $\theta$					
Bottom (R)	<p>The result is stored in registers as shown below. The decimal portion is truncated after the 4th decimal place.</p> <p>In the example, "400003" was specified for the bottom element.</p> <table border="1" style="margin-left: 40px;"> <tr> <td>400003</td> <td>Integer portion of <math> \sin\theta </math></td> </tr> <tr> <td>400004</td> <td>Decimal portion of <math> \sin\theta </math></td> </tr> </table>	400003	Integer portion of $ \sin\theta $	400004	Decimal portion of $ \sin\theta $	<p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400003	Integer portion of $ \sin\theta $					
400004	Decimal portion of $ \sin\theta $					

**B. DECIMAL COSINE (COS)****1) Function**

The cosine of an angle,  $\theta$ , between  $0^\circ$  and  $360^\circ$  is found.

**2) Structure****Table 4.19 Structural Elements of COS**

Element	Meaning	Possible Settings				
Top ( $\theta$ )	<p>The contents of the specified register and the next register is used as <math>\theta</math> as shown below.</p> <p><math>\theta</math> must be between <math>0.0000^\circ</math> and <math>360.0000^\circ</math>.</p> <p>In the example, "400001" was specified for the top element.</p> <table border="1" style="margin-left: 40px;"> <tr> <td>400001</td> <td>Integer portion of <math>\theta</math></td> </tr> <tr> <td>400002</td> <td>Decimal portion of <math>\theta</math></td> </tr> </table>	400001	Integer portion of $\theta$	400002	Decimal portion of $\theta$	<p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400001	Integer portion of $\theta$					
400002	Decimal portion of $\theta$					
Bottom (R)	<p>The result is stored in registers as shown below. The decimal portion is truncated after the 4th decimal place.</p> <p>In the example, "400003" was specified for the bottom element.</p> <table border="1" style="margin-left: 40px;"> <tr> <td>400003</td> <td>Integer portion of <math> \cos\theta </math></td> </tr> <tr> <td>400004</td> <td>Decimal portion of <math> \cos\theta </math></td> </tr> </table>	400003	Integer portion of $ \cos\theta $	400004	Decimal portion of $ \cos\theta $	<p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400003	Integer portion of $ \cos\theta $					
400004	Decimal portion of $ \cos\theta $					

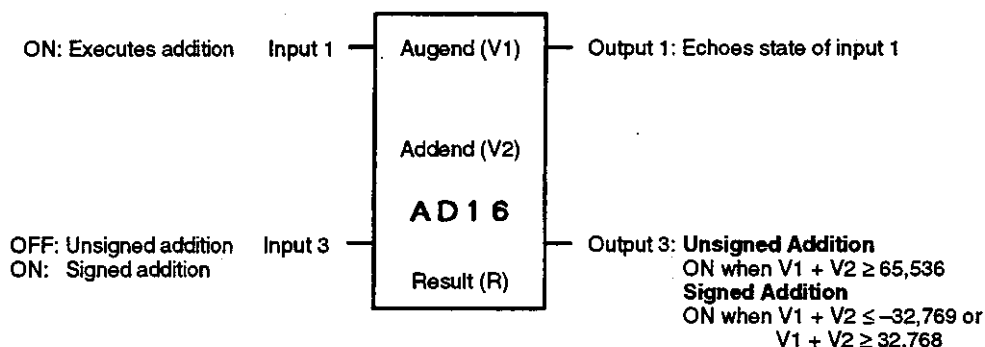
## 7. Sixteen-bit Arithmetic Instructions

### A. 16-BIT ADDITION (AD16)

#### 1) Function

Unsigned or signed addition is performed between two 16-bit binary numbers, V1 and V2. A negative number is treated as its two's complement.

#### 2) Structure



### B. 16-BIT SUBTRACTION (SU16)

#### 1) Function

Unsigned or signed subtraction is performed between two 16-bit binary numbers, V1 and V2. A negative number is treated as its two's complement.

#### 2) Structure

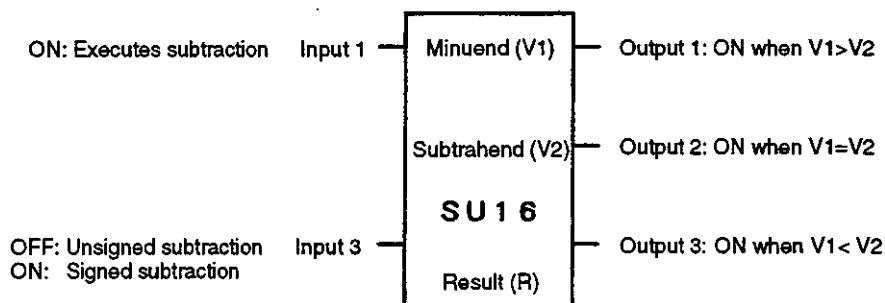


Table 4.20 Structural Elements of AD16 and SU16

Element	Meaning	Possible Settings
Top (V1)	1) If a constant is specified, its value is used as the minuend, V1. If a reference number is specified, the contents of the register is used.  2) V1 must be between the following values: <b>Unsigned Addition/Subtraction</b> Between 0 and 65,535 <b>Signed Addition/Subtraction</b> Between -32,768 and 32,767	Constant: #00000 to #65535  Input register: 300001 to 300512 (Z00001 to Z00512)  Holding register: 400001 to 409999 (W00001 to W09999)  Constant register: 700001 to 704096 (K00001 to K04096)  Link register: R10001 to R11024 R20001 to R21024
Middle (V2)	1) If a constant is specified, its value is used as the subtrahend, V2. If a reference number is specified, the contents of the register is used.  2) V2 must be within the same ranges as V1.	
Bottom (R)	1) The result is stored in the register.  2) The result must be within the same ranges as V1.	Holding register: 400001 to 409999 (W00001 to W09999)  Link register: R10001 to R11024 R20001 to R21024

### C. 16-BIT MULTIPLICATION (MU16)

#### 1) Function

Unsigned or signed multiplication is performed between two 16-bit binary numbers, V1 and V2. A negative number is treated as its two's complement.

#### 2) Structure

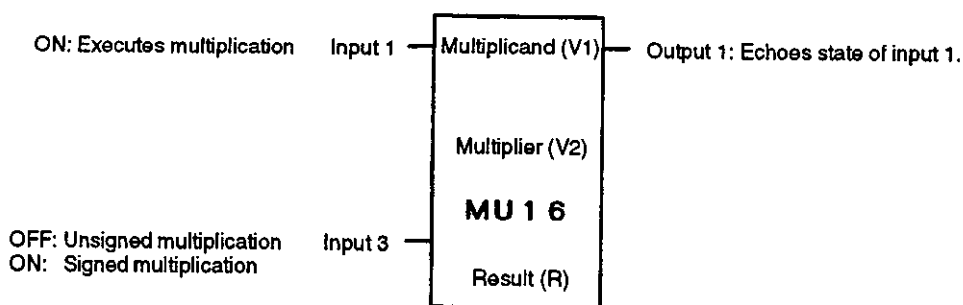


Table 4.21 Structural Elements of MU16

Element	Meaning	Possible Settings				
Top (V1)	<p>1) If a constant is specified, its value is used as the multiplicand, V1. If a reference number is specified, the contents of the register is used.</p> <p>2) V1 must be between the following values:  <b>Unsigned Multiplication</b>                      Between 0 and 65,535  <b>Signed Multiplication</b>                      Between -32,768 and 32,767</p>	<p>Constant: #00000 to #65535</p> <p>Input register: 300001 to 300512 (Z00001 to Z00512)</p> <p>Holding register: 400001 to 409999 (W00001 to W09999)</p> <p>Constant register: 700001 to 704096 (K00001 to K04096)</p> <p>Link register: R10001 to R11024 R20001 to R21024</p>				
Middle (V2)	<p>1) If a constant is specified, its value is used as the multiplier, V2. If a reference number is specified, the contents of the register is used.</p> <p>2) V2 must be within the same ranges as V1.</p>					
Bottom (R)	<p>1) The result is stored in two consecutive registers, as shown in the following example.</p> <p>2) The range of the result is the range that can be expressed with a 32-bit binary number. The value of the result must thus be between 0 and 4,294,967,295 for unsigned multiplication and between -2,147,483,648 and 2,147,483,647 for signed multiplication.</p> <p>3) In the example, "400001" was specified for the bottom element.</p> <table border="1" style="margin-left: 20px;"> <tr> <td>400002</td> <td>Upper 16 bits</td> </tr> <tr> <td>400001</td> <td>Lower 16 bits</td> </tr> </table>	400002	Upper 16 bits	400001	Lower 16 bits	<p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400002	Upper 16 bits					
400001	Lower 16 bits					

**D. 16-BIT DIVISION (DV16)**

**1) Function**

Unsigned or signed division is performed between two 16-bit binary numbers, V1 and V2. A negative number is treated as its two's complement. The remainder is also found.

**2) Structure**

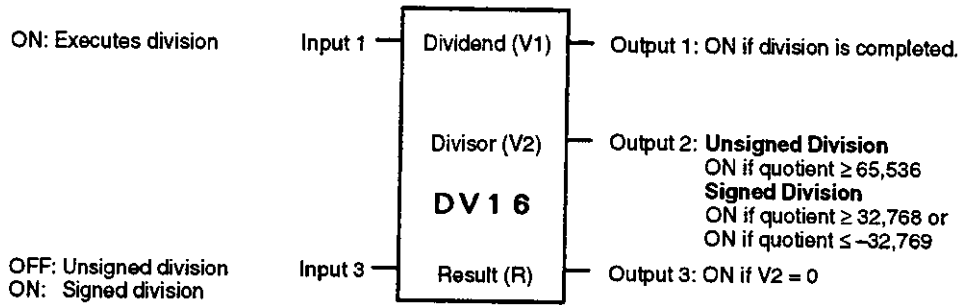


Table 4.22 Structural Elements of DV16

Element	Meaning	Possible Settings				
Top (V1)	<p>1) If a constant is specified, its value is used as the dividend, V1. The range of V1 is the range that can be expressed with a 16-bit binary number, i.e., between 0 and 65,535 for unsigned division and between -32,768 and 32,767 for signed division.</p> <p>2) If a reference number is specified, the contents of the specified register and the next reference is used. The range of V1 is the range that can be expressed with a 32-bit binary number, i.e., between 0 and 4,294,967,295 for unsigned division and between -2,147,483,648 and 2,147,483,647 for signed division.</p> <p>3) In the example, "400001" was specified for the top element,</p> <table border="1" style="margin-left: 40px;"> <tr> <td style="padding: 2px;">400002</td> <td style="padding: 2px;">Upper 16 bits</td> </tr> <tr> <td style="padding: 2px;">400001</td> <td style="padding: 2px;">Lower 16 bits</td> </tr> </table>	400002	Upper 16 bits	400001	Lower 16 bits	<p>Constant: #00000 to #65535</p> <p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400002	Upper 16 bits					
400001	Lower 16 bits					
Middle (V2)	<p>1) If a constant is specified, its value is used as the divisor, V2.</p> <p>2) If a reference number is specified, the contents of the register is used.</p> <p>3) The range of V2 is the range that can be expressed with a 16-bit binary number, i.e., between 0 and 65,535 for unsigned division and between -32,768 and 32,767 for signed division.</p>	<p>Input register: 300001 to 300512 (Z00001 to Z00512)</p> <p>Holding register: 400001 to 409999 (W00001 to W09999)</p> <p>Constant register: 700001 to 704096 (K00001 to K04096)</p> <p>Link register: R10001 to R11024 R20001 to R21024</p>				



Element	Meaning	Possible Settings
Bottom (R)	1) The result is stored in two consecutive registers, as shown in the following example. 2) The range of the result is the same as that of V2. 3) In the example, "400004" was specified for the bottom element.  400005    Quotient 400004    Remainder	Holding register: 400001 to 409998 (W00001 to W09998)  Link register: R10001 to R11023 R20001 to R21023

## 8. Thirty-two-bit Arithmetic Instructions

### A. 32-BIT ADDITION (AD32)

#### 1) Function

Unsigned or signed addition is performed between two 32-bit binary numbers, V1 and V2. A negative number is treated as its two's complement.

#### 2) Structure

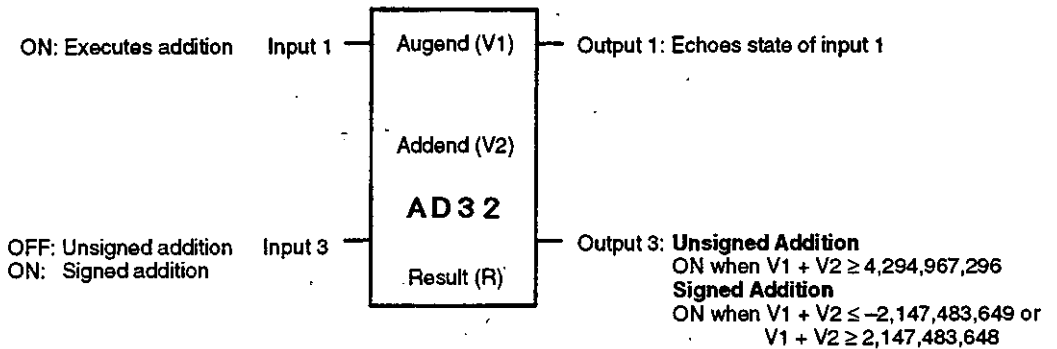


Table 4.23 Structural Elements of AD32

Element	Meaning	Possible Settings				
Top (V1)	<p>1) The contents of the specified register and the next register is used as the augend, V1.</p> <p>2) V1 must be within the range that can be expressed with a 32-bit binary number, i.e., the following ranges:  <b>Unsigned Addition</b>  Between 0 and 4,294,967,295  <b>Signed Addition</b>  Between -2,147,483,648 and 2,147,483,647</p> <p>3) In the example, "400001" was specified for the top element.</p> <table border="1" style="margin-left: 40px;"> <tr> <td>400002</td> <td>Upper 16 bits</td> </tr> <tr> <td>400001</td> <td>Lower 16 bits</td> </tr> </table>	400002	Upper 16 bits	400001	Lower 16 bits	<p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400002	Upper 16 bits					
400001	Lower 16 bits					
Middle (V2)	<p>1) The contents of the specified register and the next register is used as the addend, V2.</p> <p>2) V2 must be within the same ranges as V1.</p> <p>3) In the example, "400003" was specified for the middle element.</p> <table border="1" style="margin-left: 40px;"> <tr> <td>400004</td> <td>Upper 16 bits</td> </tr> <tr> <td>400003</td> <td>Lower 16 bits</td> </tr> </table>	400004	Upper 16 bits	400003	Lower 16 bits	
400004	Upper 16 bits					
400003	Lower 16 bits					
Bottom (R)	<p>1) The result is stored in the specified register and the next register.</p> <p>2) R must be within the same ranges as V1.</p> <p>3) In the example, "400005" was specified for the bottom element.</p> <table border="1" style="margin-left: 40px;"> <tr> <td>400006</td> <td>Upper 16 bits</td> </tr> <tr> <td>400005</td> <td>Lower 16 bits</td> </tr> </table>	400006	Upper 16 bits	400005	Lower 16 bits	<p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400006	Upper 16 bits					
400005	Lower 16 bits					

## B. 32-BIT SUBTRACTION (SU32)

### 1) Function

Unsigned or signed subtraction is performed between two 32-bit binary numbers, V1 and V2. A negative number is treated as its two's complement.

2) Structure

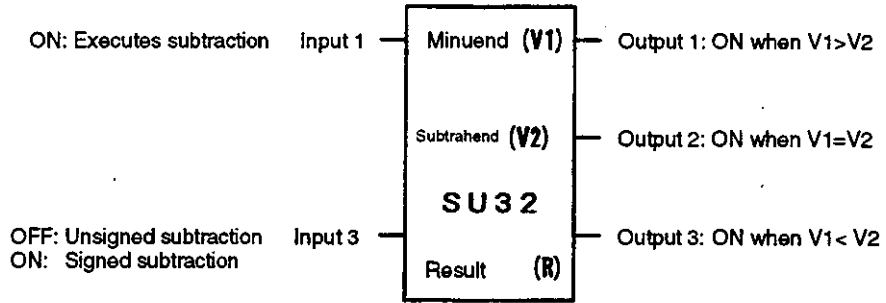


Table 4.24 Structural Elements of SU32

Element	Meaning	Possible Settings				
Top (V1)	<p>1) The contents of the specified register and the next register is used as the minuend, V1.</p> <p>2) V1 must be within the range that can be expressed with a 32-bit binary number, i.e., the following ranges:  <b>Unsigned Subtraction</b>                      Between 0 and 4,294,967,295  <b>Signed Subtraction</b>                      Between -2,147,483,648 and 2,147,483,647</p> <p>3) In the example, "400001" was specified for the top element.</p> <table border="1" style="margin-left: 40px;"> <tr> <td>400002</td> <td>Upper 16 bits</td> </tr> <tr> <td>400001</td> <td>Lower 16 bits</td> </tr> </table>	400002	Upper 16 bits	400001	Lower 16 bits	<p>Input register: 300001 to 300511 (Z00001 to Z00511)</p> <p>Holding register: 400001 to 409998 (W00001 to W09998)</p> <p>Constant register: 700001 to 704095 (K00001 to K04095)</p> <p>Link register: R10001 to R11023 R20001 to R21023</p>
400002	Upper 16 bits					
400001	Lower 16 bits					
Middle (V2)	<p>1) The contents of the specified register and the next register is used as the subtrahend, V2.</p> <p>2) V2 must be within the same ranges as V1.</p> <p>3) In the example, "400003" was specified for the middle element.</p> <table border="1" style="margin-left: 40px;"> <tr> <td>400004</td> <td>Upper 16 bits</td> </tr> <tr> <td>400003</td> <td>Lower 16 bits</td> </tr> </table>	400004	Upper 16 bits	400003	Lower 16 bits	
400004	Upper 16 bits					
400003	Lower 16 bits					

Element	Meaning	Possible Settings		
Bottom (R)	1) The result is stored in the specified register and the next register. 2) R must be within the same ranges as V1. 3) In the example, "400005" was specified for the bottom element.	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 R20001 to R21023		
	400006 <table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>Upper 16 bits</td></tr> <tr><td>Lower 16 bits</td></tr> </table>	Upper 16 bits	Lower 16 bits	
Upper 16 bits				
Lower 16 bits				
	400005			

### C. 32-BIT COMPARE (TEST)

#### 1) Function

##### a) Sixteen-bit Comparison

Unsigned or signed comparison is performed between two 16-bit binary numbers, V1 and V2. A negative number is treated as its two's complement.

##### b) Thirty-two-bit Comparison

Unsigned or signed comparison is performed between two 32-bit binary numbers, V1 and V2. A negative number is treated as its two's complement.

#### 2) Structure

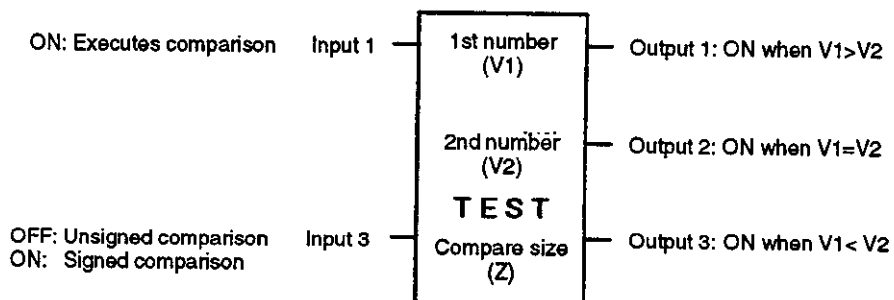


Table 4.25 Structural Elements of TEST for 16-bit Comparison

Element	Meaning	Possible Settings
Top (V1)	1) The contents of the specified register is used as V1.  2) V1 must be within the range that can be expressed with a 16-bit binary number, i.e., the following ranges: <b>Unsigned Comparison</b> Between 0 and 65,535 <b>Signed Comparison</b> Between -32,768 and 32,767	Input register: 300001 to 300512 (Z00001 to Z00512)  Holding register: 400001 to 409999 (W00001 to W09999)  Constant register: 700001 to 704096 (K00001 to K04096)  Link register: R10001 to R11024 R20001 to R21024
Middle (V2)	1) The contents of the specified register is used as V2.  2) V2 must be within the same range as V1.	
Bottom (Z)	Comparison size	Constant: #00001

Table 4.26 Structural Elements of TEST for 32-bit Comparison

Element	Meaning	Possible Settings		
Top (V1)	1) The contents of the specified register and the next register is used as V1.  2) V1 must be within the range that can be expressed with a 32-bit binary number, i.e., the following ranges: <b>Unsigned Comparison</b> Between 0 and 4,294,967,295 <b>Signed Comparison</b> Between -2,147,483,648 and 2,147,483,647  3) In the example, "400001" was specified for the top element.  400002 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Upper 16 bits</td></tr><tr><td>Lower 16 bits</td></tr></table> 400001	Upper 16 bits	Lower 16 bits	Input register: 300001 to 300511 (Z00001 to Z00511)  Holding register: 400001 to 409998 (W00001 to W09998)  Constant register: 700001 to 704095 (K00001 to K04095)  Link register: R10001 to R11023 R20001 to R21023
Upper 16 bits				
Lower 16 bits				
Middle (V2)	1) The contents of the specified register and the next register is used as V2.  2) V2 must be within the same ranges as V1.  3) In the example, "400003" was specified for the middle element.  400004 <table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>Upper 16 bits</td></tr><tr><td>Lower 16 bits</td></tr></table> 400003	Upper 16 bits	Lower 16 bits	
Upper 16 bits				
Lower 16 bits				
Bottom (Z)	Comparison size	Constant: #00002		

## 4.2.3 Data Transfer Instructions

### 1. REGISTER-TO-TABLE MOVE (R→T)

#### A. Function

- 1) Data from a single register is transferred to a data table. The pointer value determines which register in the table is the destination register.
- 2) The word of data in the source is copied to the register in the destination table specified by the pointer value.

#### B. Structure

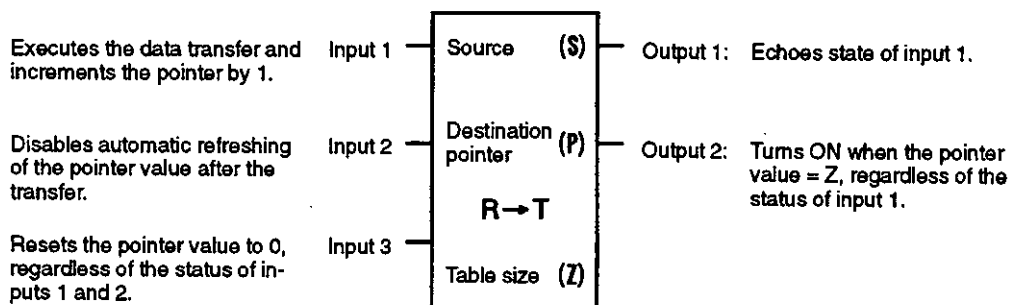


Table 4.27 Structural Elements of R→T

Element	Meaning	Possible Settings
Top (S)	Reference number of the source	Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (P)	Reference number of the pointer	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 or R20001 to R21023
Bottom (Z)	Size of the destination table	Constant: #00001 to #00999

## 2. TABLE-TO-REGISTER MOVE (T→R)

### A. Function

- 1) Data is transferred from a data table to a single register. The pointer is just before the destination register and the pointer value determines which register in the source table is the source register.
- 2) Any register in the source table can be specified by changing the pointer value. The data in the specified register is copied to the destination register.

### B. Structure

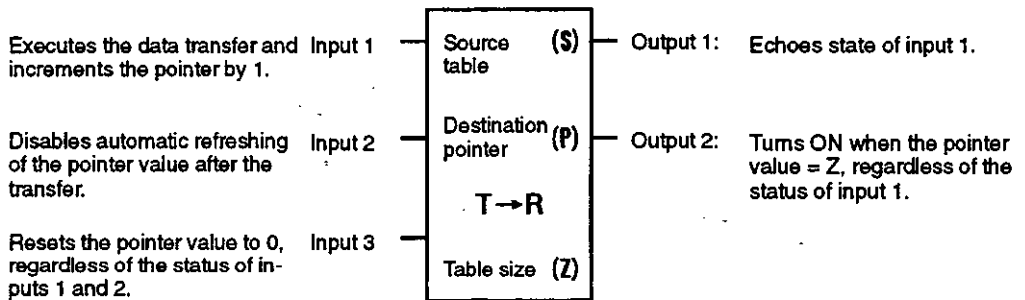


Table 4.28 Structural Elements of T→R

Element	Meaning	Possible Settings
Top (S)	Reference number of the leading word in the source table	Coil: 00001 to 008177 (O0001 to O08177)
		Input relay: 10001 to 101009 (I0001 to I01009)
		Input register: 30001 to 300512 (Z0001 to Z00512)
		Holding register: 40001 to 409999 (W0001 to W09999)
		Constant register: 70001 to 704096 (K0001 to K04096)
		Link coil: D1001 to D11009 or D2001 to D21009
		Link register: R1001 to R11024 or R2001 to R21024
		MC coil: Y1001 to Y10241 or Y2001 to Y20241
		MC control coil: Q1001 to Q10145 or Q2001 to Q20145
		MC relay: X1001 to X10241 or X2001 to X20241
		MC control relay: P1001 to P10241 or P2001 to P20241
M code relay: M1001 to M10081 or M2001 to M20081		
Middle (P)	Reference number of the pointer	Holding register: 40001 to 409998 (W0001 to W09998)
		Link register: R1001 to R11023 or R2001 to R21023

Element	Meaning	Possible Settings
Bottom (Z)	Size of the source table	Specify the constant. The maximum value of the constant differs with specified reference type. Coil: #00001 to #00512 Input relay: #00001 to #00064 Input register: #00001 to #00512 Holding register or constant register: #00001 to #00999 Link coil: #00001 to #00064 Link register: #00001 to #00999 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

### 3. TABLE-TO-TABLE MOVE (T→T)

#### A. Function

- 1) Data is transferred from a data table to another data table of the same size. The pointer is just before the destination table and the pointer value determines which register in the table is copied.
- 2) The data from the specified register in the source table is copied to the corresponding register in the destination table according to the pointer value.

#### B. Structure

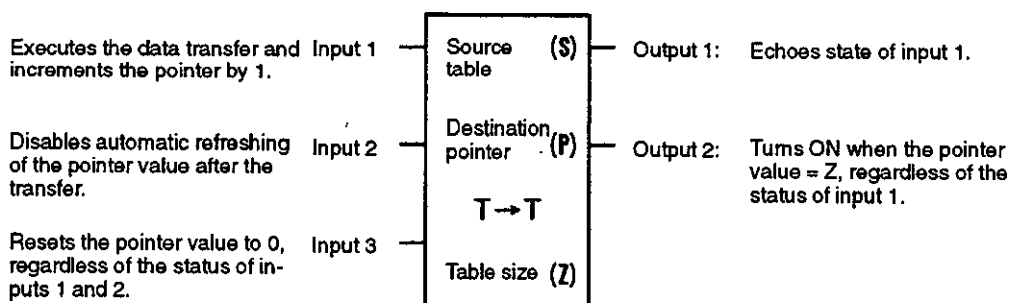




Table 4.29 Structural Elements of T→T

Element	Meaning	Possible Settings
Top (S)	Reference number of the leading word in the source table	Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (P)	Reference number of the pointer	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 or R20001 to R21023
Bottom (Z)	Size of the source and destination tables	Specify the constant. The maximum value of the constant differs with specified reference type. Coil: #00001 to #00512 Input relay: #00001 to #00064 Input register: #00001 to #00512 Holding register or constant register: #00001 to #00999 Link coil: #00001 to #00064 Link register: #00001 to #00999 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

#### 4. FIRST IN (FIN)

##### A. Function

- 1) Data is transferred from a single register to a data table. The pointer value indicates how many words of data have been stored in the table by the FIRST IN instruction.
- 2) When there is an "empty register" in the destination table, all of the data is shifted down by one word to empty the leading register in the table and the contents of the source word

are copied to that empty register.

- 3) The registers farther down in the table contain older data (data that was transferred earlier).
- 4) The FIRST IN instruction is usually used in conjunction with the FIRST OUT instruction. The FIRST OUT instruction transfers the oldest data from the table to a specified destination register. Refer to *item 5. FIRST OUT (FOUT)* for details.

## B. Structure

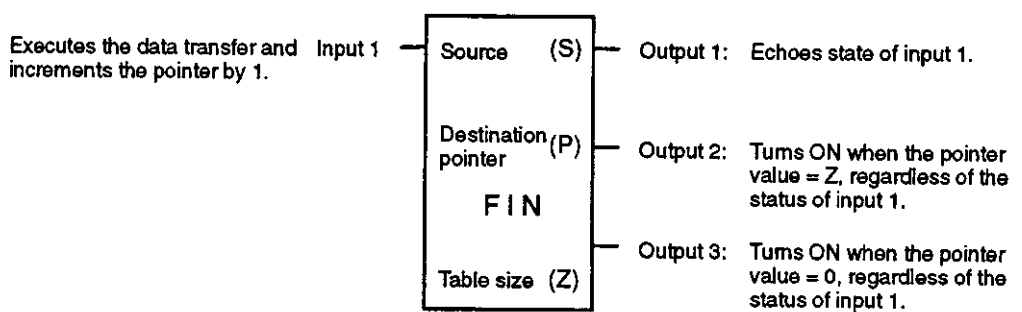


Table 4.30 Structural Elements of FIN

Element	Meaning	Possible Settings
Top (S)	Reference number of the source	Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (P)	Reference number of the pointer	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 or R20001 to R21023
Bottom (Z)	Size of the destination table	Constant: #00001 to #00100

### 5. FIRST OUT (FOUT)

#### A. Function

- 1) Data is transferred from a data table to a single register. The pointer is located just before the source table and it indicates how many words of data can be read by the FIRST OUT instruction.
- 2) The data that was transferred to the table first (the oldest data) is transferred to the destination.
- 3) The FIRST OUT instruction is usually used to read data entered into the source table with the FIRST IN instruction.

#### B. Structure

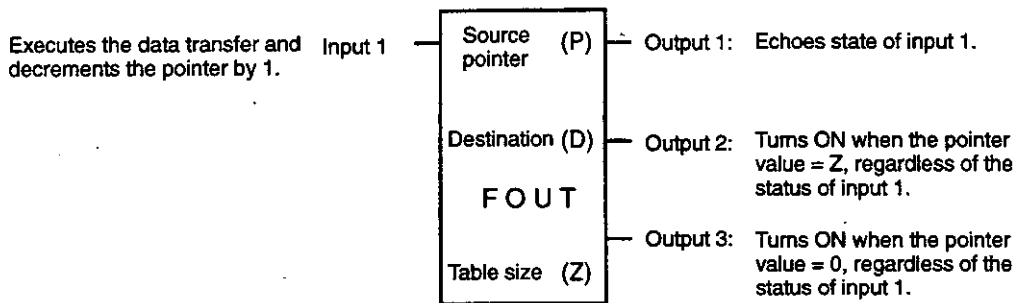


Table 4.31 Structural Elements of FOUT

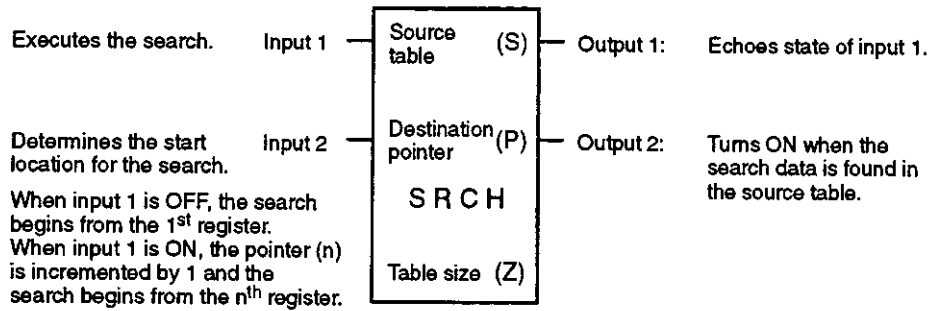
Element	Meaning	Possible Settings
Top (P)	Reference number of the source pointer	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 or R21001 to R21023
Middle (D)	Reference number of the destination	Coil: 000001 to 008161 (O00001 to O08161) Holding register: 400001 to 409999 (W00001 to W09999) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11023 or R21001 to R21023 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145
Bottom (Z)	Size of the source table	Constant: #00001 to #00100

### 6. TABLE SEARCH (SRCH)

#### A. Function

- 1) The source is a data table and the destination is a single register (the next register after the pointer). When the specified search data is found in the source table, that table position is stored in the pointer.
- 2) Searches the source table for the search data and writes that table position in the pointer.

**B. Structure**



**Table 4.32 Structural Elements of SRCH**

Element	Meaning	Possible Settings
Top (S)	Reference number of the leading register in the source table	Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Middle (P)	Reference number of the pointer	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 R20001 to R21023
Bottom (Z)	Size of the source table	Constant: #00001 to #00100

**7. TABLE SET (TSET)**

**A. Function**

- 1) The source is a single register and the destination is a data table. There is no pointer.
- 2) Copies the content of the source (1 word of data) to all of the registers in the destination table. The transfer is completed in one scan.

**B. Structure**

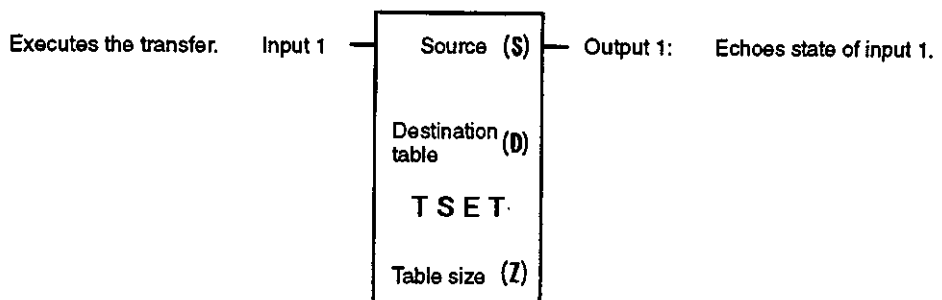


Table 4.33 Structural Elements of TSET

Element	Meaning	Possible Settings
Top (S)	Reference number of the source	Input register: 300001 to 300512 (Z00001 to Z00512)  Holding register: 400001 to 409999 (W00001 to W09999)  Constant register: 700001 to 704096 (K00001 to K04096)  Link register: R10001 to R11024 R20001 to R21024
Middle (D)	Reference number of the leading register in the destination table	Holding register: 400001 to 409999 (W00001 to W09999)  Link register: R10001 to R11024 R20001 to R21024
Bottom (Z)	Size of the destination table	Constant: #00001 to #00100

## 8. BLOCK MOVE (BLKM)

### A. Function

- 1) The source and destination are data tables of the same size. There is no pointer.
- 2) Copies the contents of the registers in the source table to the corresponding registers in the destination table in a single scan.

### B. Structure

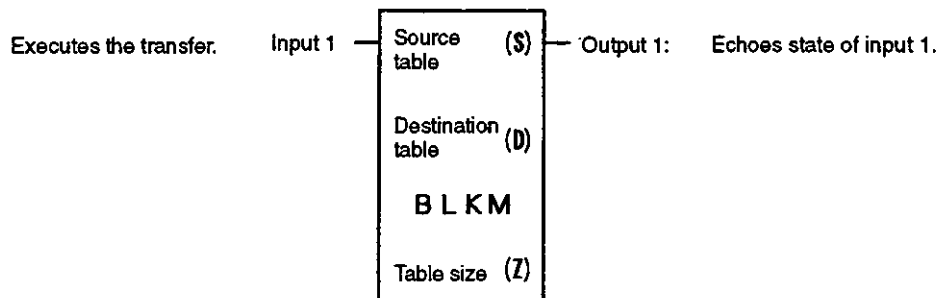


Table 4.34 Structural Elements of BLKM

Element	Meaning	Possible Settings
Top (S)	Reference number of the leading register in the source table	Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (D)	Reference number of the leading register in the destination table	Coil: 000001 to 008161 (O00001 to O08161) Holding register: 400001 to 409999 (W00001 to W09999) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145
Bottom (Z)	Size of the source and destination table	Specify the constant. The maximum value of the constant differs with specified reference type. Coil: #00001 to #00100 Input relay: #00001 to #00064 Input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

## 9. BLOCK-TO-TABLE MOVE (BLKT)

### A. Function

- 1) Data is transferred from a data block to a data table. The pointer is located just before the destination table and the pointer value determines which block in the table is the destination block.
- 2) The source block is a single data block. (The size of a data block is defined by the user in the bottom element of the instruction.)
- 3) The destination table is made up of a series of data blocks which are referred to as destination block 1, destination block 2, destination block 3, etc.
- 4) The data in the source block can be copied to any destination block in the destination table by adjusting the pointer value. The transfer is completed in one scan.

### B. Structure

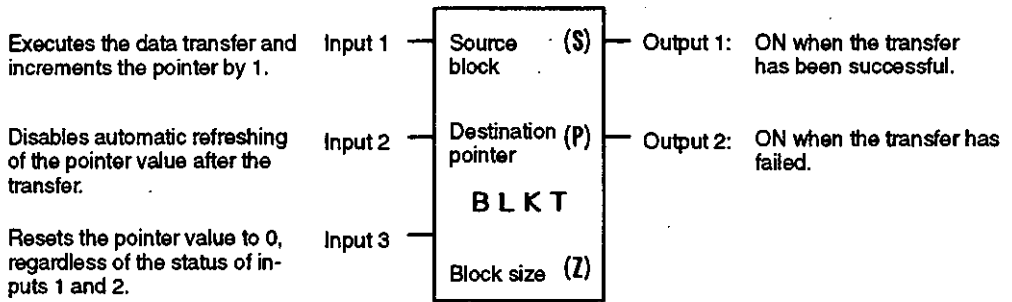


Table 4.35 Structural Elements of BLKT

Element	Meaning	Possible Settings
Top (S)	Reference number of the leading register in the source block	Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (P)	Reference number of the pointer	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 or R20001 to R21023
Bottom (Z)	Size of the source and destination blocks	Specify the constant. The maximum value of the constant differs with specified reference type. Coil: #00001 to #00100 Input relay: #00001 to #00064 Input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

## 10. TABLE-TO-BLOCK MOVE (TBLK)

### A. Function

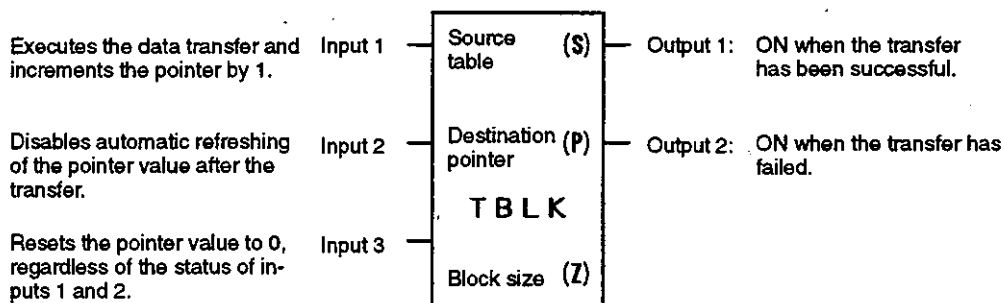
- 1) Data is transferred from a data table to a data block. The pointer is located just before the destination block and the pointer value determines which block in the source table is the source block.
- 2) The destination block is a single data block. (The size of a data block is defined by the user in the bottom element of the instruction.)



3) The source table is made up of a series of data blocks which are referred to as source block 1, source block 2, source block 3, etc.

4) The data from any source block in the source table can be copied to the destination block by adjusting the pointer value. The transfer is completed in one scan.

**B. Structure**



**Table 4.36 Structural Elements of TBLK**

Element	Meaning	Possible Settings
Top (S)	Reference number of the leading register in the source table	Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (P)	Reference number of the pointer	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 or R20001 to R21023

Element	Meaning	Possible Settings
Bottom (Z)	Size of the source and destination blocks	Specify the constant. The maximum value of the constant differs with specified reference type. Coil: #00001 to #00100 Input relay: #00001 to #00064 Input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

## 11. INDIRECT BLOCK WRITE (IBKW)

### A. Function

- 1) Data is transferred between a source table and destination registers using a pointer table that is the same size as the source table. The destination registers are determined by the pointer values in the pointer table, so the destination registers do not have to be consecutive or in any particular order.
- 2) The source table is known as the source block and the data table containing the pointers is known as the pointer block.
- 3) The content of each word in the source block can be copied to any destination holding register by adjusting the content of the corresponding registers in the pointer block. The transfer is completed in one scan.

### B. Structure

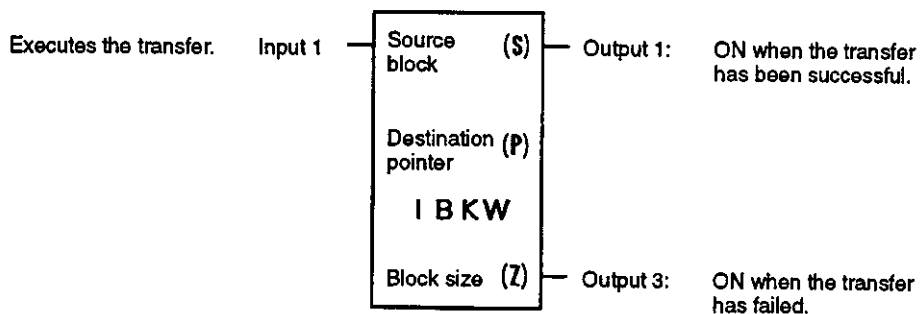


Table 4.37 Structural Elements of IBKW

Element	Meaning	Possible Settings
Top (S)	Reference number of the leading register in the source block	Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Middle (P)	Reference number of the leading register in the pointer block	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 R20001 to R21024
Bottom (Z)	Size of the source and pointer blocks	Constant: #00001 to #00255

## 12.INDIRECT BLOCK READ (IBKR)

### A. Function

- 1) Data is transferred between a source table and destination registers using a pointer table that is the same size as the destination table. The source registers are determined by the pointers values in the pointer table, so the source registers do not have to be consecutive or in any particular order.
- 2) The data table containing the pointers is known as the pointer block and the data table containing the destination registers is known as the destination block.
- 3) The content of any holding register can be copied to a destination register by adjusting the content of the corresponding register in the pointer block. The transfer is completed in one scan.

### B. Structure

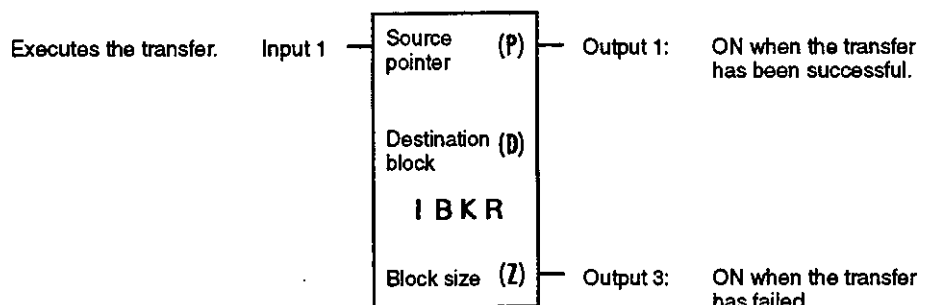


Table 4.38 Structural Elements of IBKR

Element	Meaning	Possible Settings
Top (P)	Reference number of the leading register in the pointer block	Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Middle (D)	Reference number of the leading register in the destination block	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 R20001 to R21024
Bottom (Z)	Size of the pointer and destination blocks	Constant: #00001 to #00255

## 4.2.4 Indexed Block Transfer Instructions

### 1. DESTINATION INDEXED BLOCK TRANSFER 1 (DIBT)

#### A. Function

- 1) Data is transferred using indexed blocks between source and destination data tables that differ in size. A pointer is on the destination side and used to determine the destination of the data.
- 2) The source data table is called a source block. Coil tables, relay tables, and register tables can be used as source blocks. Only input relay tables can be used as the destination tables.
- 3) The contents of the source block are copied to the block in the destination table specified by the pointer (called the destination block). The transfer is completed in one scan.

#### B. Structure

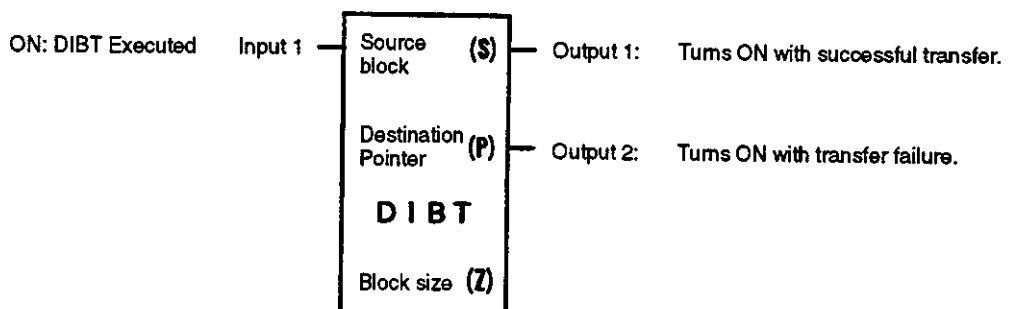


Table 4.39 DIBT Structural Elements

Element	Meaning	Possible settings
Top (S)	Reference number of the leading register in the source block	Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (P)	Reference number of the pointer	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 or R20001 to R21024
Bottom (Z)	Size of the source block and destination blocks	Specifies the constant. The maximum value of constant differs with specified reference type. Coil: #00001 to #00100 Input relay: #00001 to #00064 Input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

## 2. DESTINATION INDEXED BLOCK TRANSFER 2 (DIBR)

### A. Function

- 1) Data is transferred using indexed blocks between source and destination data tables that differ in size. The pointer is on the destination side and determines the destination of the data.
- 2) The source data table is called a source block. Coil tables, relay tables, and register tables can be used as the source block. Only holding register tables can be used as the

destination table.

- 3) The contents of the source block are copied to the block in the destination table specified by the pointer (called the destination block). The transfer is executed in one scan.

### B. Structure

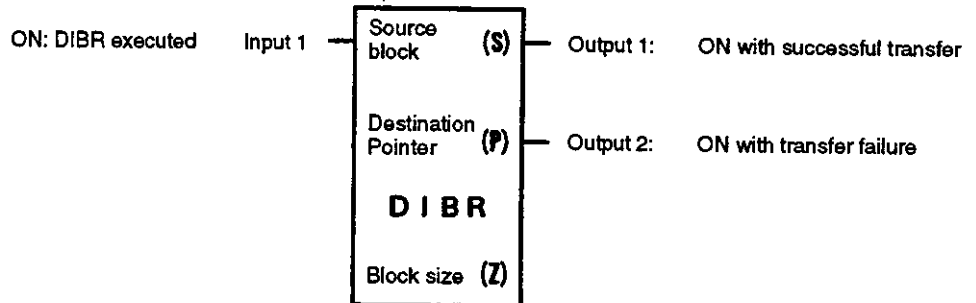


Table 4.40 DIBR Structural Elements

Element	Meaning	Possible settings
Top (S)	Reference number of the leading register in the source block	Coil: 00001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (P)	Reference number of the pointer	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 or R20001 to R21024

Element	Meaning	Possible settings
Bottom (Z)	Size of the source and destination blocks	Specify the constant. The maximum value of constant differs with specified reference type. Coil: #00001 to #00100 Input relay: #00001 to #00064 Input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

### 3. SOURCE INDEXED BLOCK TRANSFER 1 (SIBT)

#### A. Function

- 1) Data is transferred using indexed blocks between source and destination data tables that differ in size. The pointer is on the source side and determines the source of the data.
- 2) Coil tables, input relay tables, and input register tables can be used as the source table. Only holding register tables can be used as the destination block.
- 3) The contents of the source block specified by the pointer are copied to the block (holding register block) in the destination table. The transfer is completed in one scan.

#### B. Structure

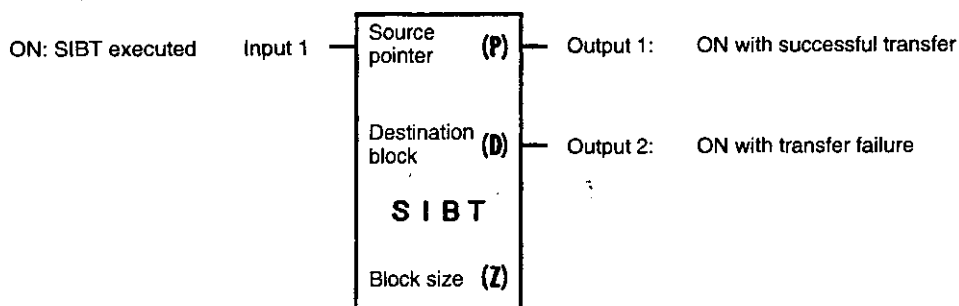


Table 4.41 SIBT Structural Elements

Element	Meaning	Possible settings
Top (P)	Pointer reference number	Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 or R20001 to R21024
Middle (D)	Leading reference number in the destination block	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 or R20001 to R21024
Bottom (Z)	Size of the source and destination blocks	Constant: #00001 to #00100

#### 4. SOURCE INDEXED BLOCK TRANSFER 2 (SIBR)

##### A. Function

- 1) Data is transferred using indexed blocks between source and destination data tables that differ in size. The pointer is on the source side and determines the source of the data.
- 2) Source tables and destination data tables must be holding register tables.
- 3) The contents of the block in the source table specified by the pointer are copied to the destination block. The transfer is completed in one scan.

##### B. Structure

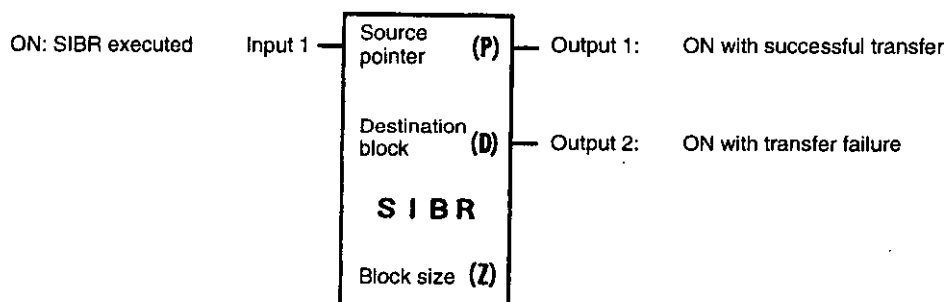




Table 4.42 SIBR Structural Elements

Element	Meaning	Possible settings
Top (P)	Pointer reference number	Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Middle (D)	Leading reference number in the destination block	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 R20001 to R21024
Bottom (Z)	Size of the source and destination blocks	Constant: #00001 to #00100

## 4.2.5 Matrix Instructions

### 1. LOGICAL AND (AND)

#### A. Function

A logical AND operation is performed between a source table and a destination table of the same size and the result is stored in the destination table. The operation is completed in one scan.

#### B. Structure

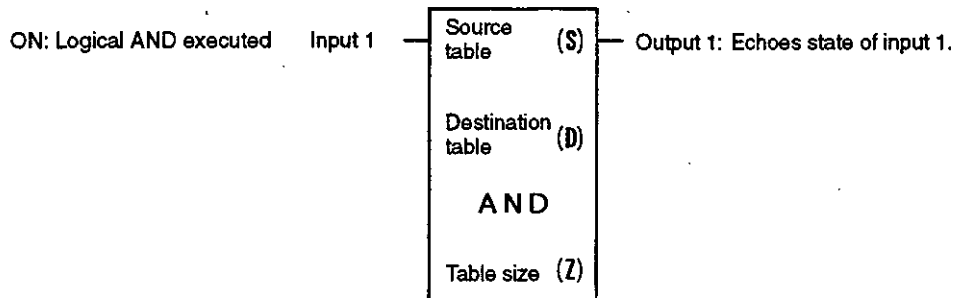


Table 4.43 Structural Elements of AND

Element	Meaning	Settings
Top (S)	Leading reference number of the source table	Constant: #00000 to #65535 Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (D)	Leading reference number of the destination table	Coil: 000001 to 008161 (O00001 to O08161) Holding register: 400001 to 409999 (W00001 to W09999) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145
Bottom (Z)	Size of the source and destination tables	Specify the constant. The maximum value of constant differs with specified reference type. Coil: #00001 to #00100 Input relay: #00001 to #00064 Constant, input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

## 2. LOGICAL OR (OR)

### A. Function

A logical OR operation is performed between a source table and a destination table of the same size and the result is stored in the destination table. The operation is completed in one scan.

### B. Structure

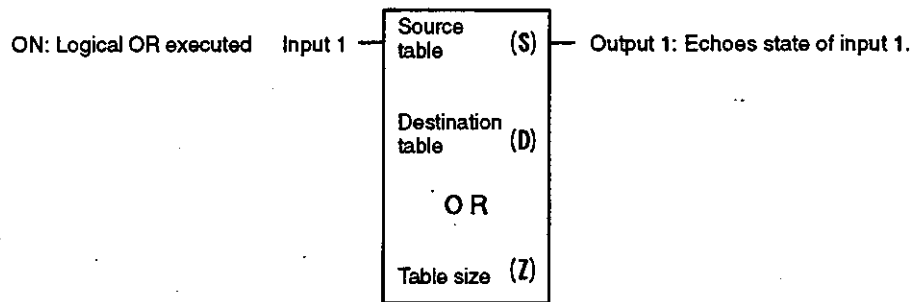


Table 4.44 Structural Elements of OR

Element	Meaning	Possible settings
Top (S)	Leading reference number of the source table	Constant: #00000 to #65535 Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (D)	Leading reference number of the destination table	Coil: 000001 to 008161 (O00001 to O08161) Holding register: 400001 to 409999 (W00001 to W09999) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145

Element	Meaning	Possible settings
Bottom (Z)	Size of the source and destination tables	Specify the constant. The maximum value of the constant differs with specified reference type. Coil: #00001 to #00100 Input relay: #00001 to #00064 Constant, input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

### 3. LOGICAL EXCLUSIVE OR (XOR)

#### A. Function

A LOGICAL EXCLUSIVE OR operation is performed between a source table and a destination table of the same size and the result is stored in the destination table. The operation is completed in one scan.

#### B. Structure

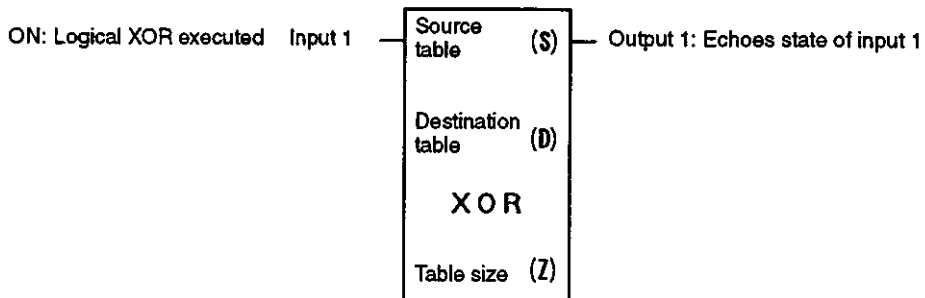


Table 4.45 Structural Elements of XOR

Element	Meaning	Possible settings
Top (S)	Leading reference number of the source table	Constant: #00000 to #65535 Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (D)	Leading reference number of the destination table	Coil: 000001 to 008161 (O00001 to O08161) Holding register: 400001 to 409999 (W00001 to W09999) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145
Bottom (Z)	Size of the source and destination tables	Specify the constant. The maximum value of the constant differs with specified reference type. Coil: #00001 to #00100 Input relay: #00001 to #00064 Constant, input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

## 4. LOGICAL COMPLEMENT (COMP)

### A. Function

The inverse of the status of each bit of the source table is stored in a destination table of the same size. The operation is completed in one scan.

### B. Structure

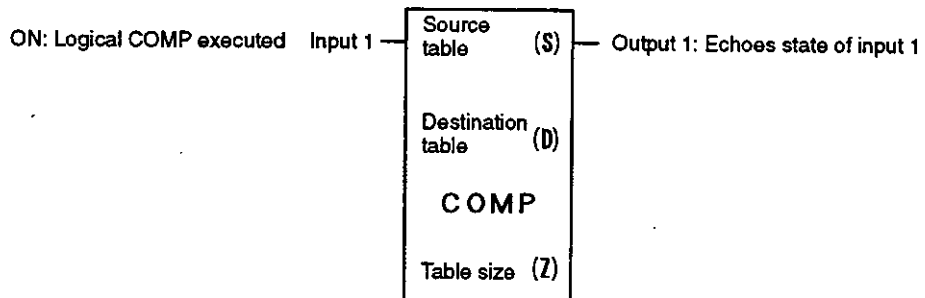


Table 4.46 Structural Elements of COMP

Element	Meaning	Possible settings	
Top (S)	Leading reference number of the source table	Coil:	000001 to 008177 (O00001 to O08177)
		Input relay:	100001 to 101009 (I00001 to I01009)
		Input register:	300001 to 300512 (Z00001 to Z00512)
		Holding register:	400001 to 409999 (W00001 to W09999)
		Constant register:	700001 to 704096 (K00001 to K04096)
		Link coil:	D10001 to D11009 or D20001 to D21009
		Link register:	R10001 to R11024 or R20001 to R21024
		MC coil:	Y10001 to Y10241 or Y20001 to Y20241
		MC control coil:	Q10001 to Q10145 or Q20001 to Q20145
		MC relay:	X10001 to X10241 or X20001 to X20241
		MC control relay:	P10001 to P10241 or P20001 to P20241
M code relay:	M10001 to M10081 or M20001 to M20081		
Middle (D)	Leading reference number of the destination table	Coil:	000001 to 008161 (O00001 to O08161)
		Holding register:	400001 to 409999 (W00001 to W09999)
		Link coil:	D10001 to D11009 or D20001 to D21009
		Link register:	R10001 to R11024 or R20001 to R21024
		MC coil:	Y10001 to Y10241 or Y20001 to Y20241
		MC control coil:	Q10001 to Q10145 or Q20001 to Q20145

Element	Meaning	Possible settings
Bottom (Z)	Size of the source and destination tables	Specify the constant. The maximum value of the constant differs with specified reference type. Coil: #00001 to #00100 Input relay: #00001 to #00064 Input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

## 5. LOGICAL COMPARE (CMPR)

### A. Function

The bit patterns of a source table and a destination table of the same size are compared one bit at a time. If bits are found that do not match, the bit number is stored in the pointer to indicate the position of the unmatched bits. One pair of unmatched bits can be found each scan.

### B. Structure

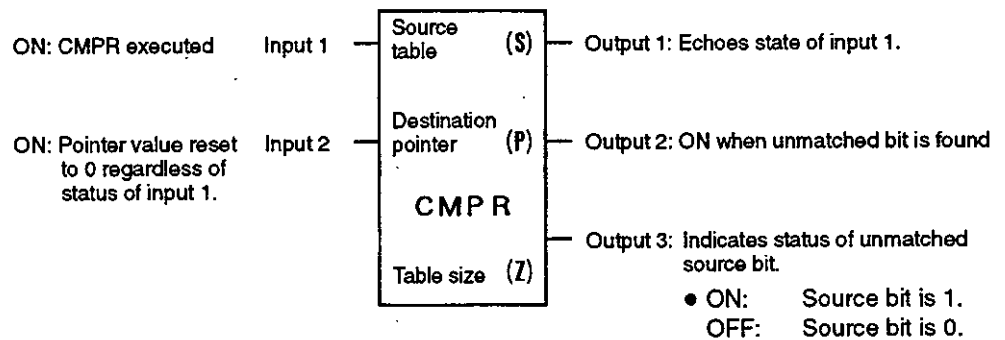


Table 4.47 Structural Elements of CMPR

Element	Meaning	Possible settings
Top (S)	Leading reference number of the source table	Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (P)	Reference number of pointer	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 or R20001 to R21023
Bottom (Z)	Size of the source and destination tables	Specify the constant. The maximum value of the constant differs with specified reference type. Coil: #00001 to #00100 Input relay: #00001 to #00064 Input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

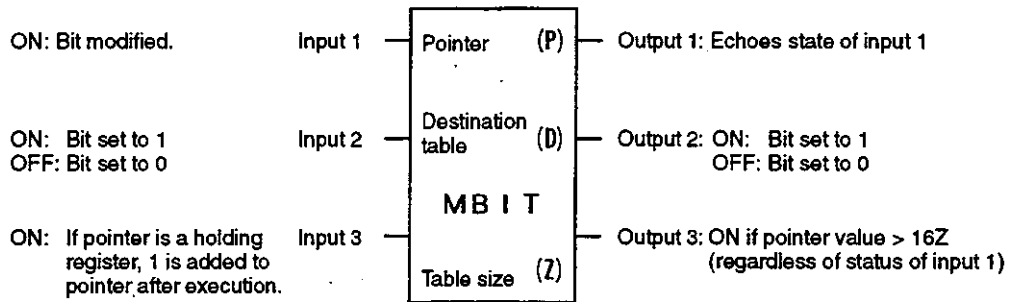
## 6. LOGICAL BIT MODIFY (MBIT)

### A. Function

A pointer (P) is used to force the status of an arbitrary bit in the destination table (DT) to either 1 or 0. The operation is completed in one scan.



**B. Structure**



**Table 4.48 Structural Elements of MBIT**

Element	Meaning	Possible settings
Top (P)	Reference number of pointer	Constant: #00001 to #09600 Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 or R20001 to R21024
Middle (D)	Leading reference number of the destination table	Coil: 000001 to 008161 (O00001 to O08161) Holding register: 400001 to 409999 (W00001 to W09999) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145
Bottom (Z)	Size of destination table	Specify the constant. The maximum value of the constant differs with reference type specified in middle element. Coil: #00001 to #00512 Holding register: #00001 to #00600 Link coil: #00001 to #00064 Link register: #00001 to #00600 MC coil: #00001 to #00016 MC control coil: #00001 to #00010

**7. LOGICAL SENSE (SENS)**

**A. Function**

A pointer is used to read the status of an arbitrary bit in the destination table. The operation is completed in one scan.

## B. Structure

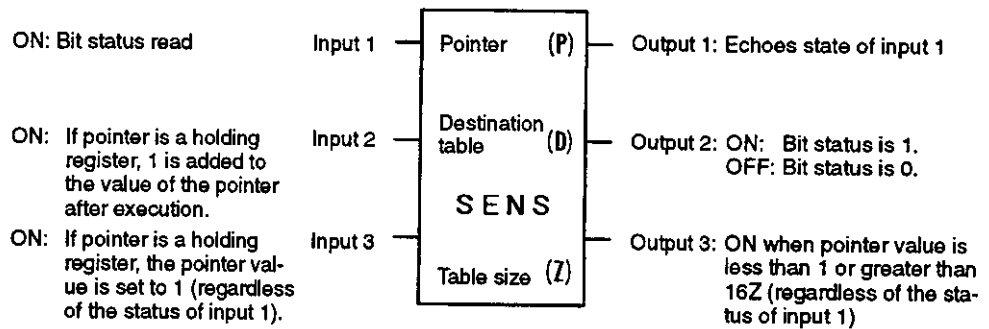


Table 4.49 Structural Elements of SENS

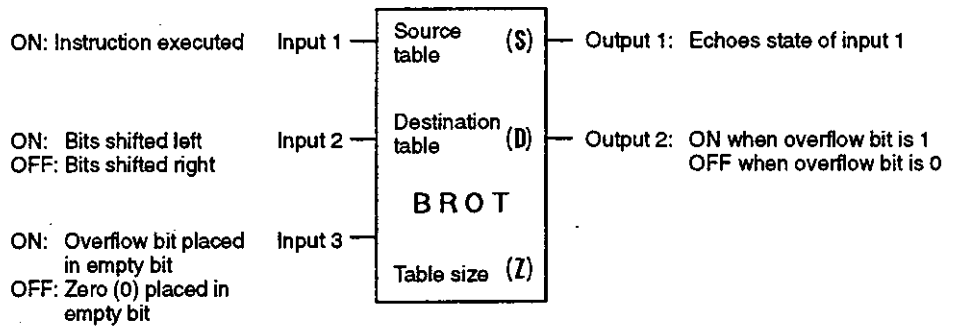
Element	Meaning	Possible settings
Top (P)	Reference number of pointer	Constant: #00001 to #09600 Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 or R10001 to R11024
Middle (D)	Leading reference number of the destination table	Coil: 000001 to 008161 (O00001 to O08161) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Bottom (Z)	Size of destination table	Specify the constant. The maximum value of the constant differs with the reference type specified in middle element. Coil: #00001 to #00512 Input relay: #00001 to #00064 Input register, holding register: #00001 to #00600 Link coil: #00001 to #00064 Link register: #00001 to #00600 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

## 8. LOGICAL BIT ROTATE (BROT)

### A. Function

The bit pattern in the source table is shifted one bit to the left or to the right and stored in a destination table of the same size. The operation is completed in one scan.

**B. Structure**



**Table 4.50 Structural Elements of BROT**

Element	Meaning	Possible settings
Top (S)	Leading reference number of the source table	Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (D)	Leading reference number of the destination table	Coil: 000001 to 008161 (O00001 to O08161) Holding register: 400001 to 409999 (W00001 to W09999) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145

Element	Meaning	Possible settings
Bottom (Z)	Size of source and destination tables	Specify the constant. The maximum value of the constant differs with the reference type. Coil: #00001 to #00100 Input relay: #00001 to #00064 Input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

## 9. LOGICAL MULTI-BIT ROTATE (MROT)

### A. Function

The bits in the destination table are shifted to the left or to the right by the number of bits specified by the source pointer (1 to 15). The operation is completed in one scan.

### B. Structure

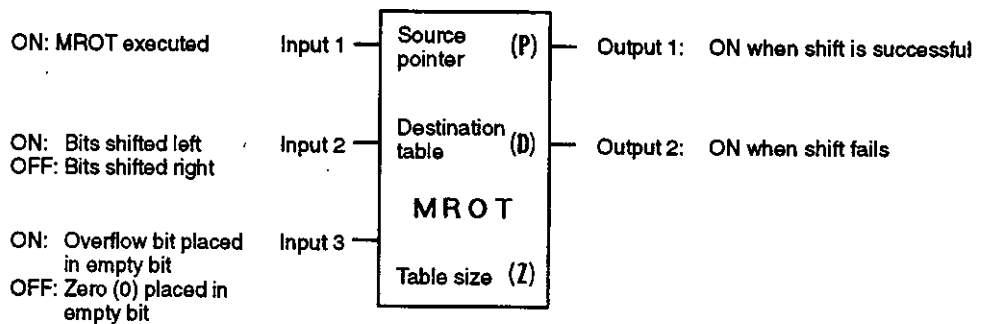


Table 4.51 Structural Elements of MROT

Element	Meaning	Possible settings
Top (P)	Reference number of the source pointer	Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 or R20001 to R21024
Middle (D)	Leading reference number of the destination table	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 or R20001 to R21024
Bottom (Z)	Size of destination table	Constant: #00001 to #00100

## 10. LOGICAL BIT COUNT (BCNT)

### A. Function

The number of 1 or 0 bits in the source table is counted. The operation is completed in one scan.

### B. Structure

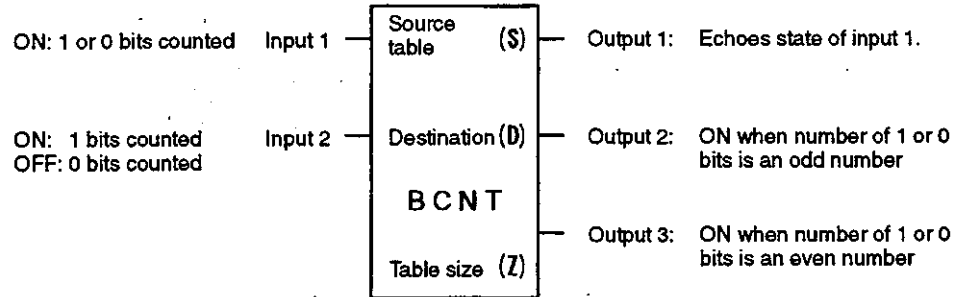


Table 4.52 Structural Elements of BCNT

Element	Meaning	Possible settings
Top (S)	Leading reference number of the source table	Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 or R20001 to R21024
Middle (D)	Reference number of the destination	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 or R20001 to R21024
Bottom (Z)	Size of the source table	Constant: #00001 to #00100

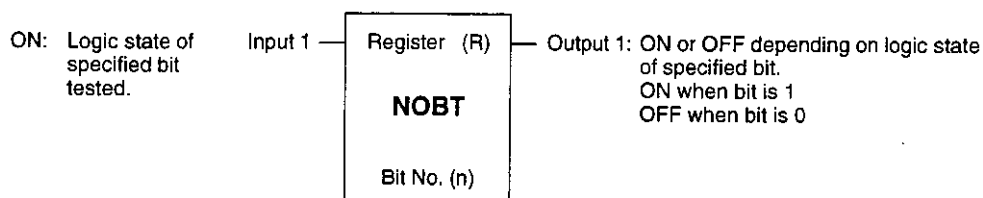
## 4.2.6 Bit Manipulation Instructions

### 1. NORMALLY OPEN BIT (NOBT)

#### A. Function

- 1) The logic state of a specified bit in a specified register is tested and output 1 is turned ON or OFF accordingly.
- 2) The same operation is performed as that for a normally open contact relay element. For a normally open contact, power flow is controlled based on the ON/OFF status of a coil. With NORMALLY OPEN BIT, power flow is controlled based on the logic state (1/0) of a specified bit.

## B. Structure



**Table 4.53 Structural Elements of NOBT**

Element	Meaning	Possible Settings
Top (R)	The logic state of the bit n in R, the register specified for the top element, is tested and output 1 is turned ON or OFF accordingly.	Input register: 300001 to 300512 (Z00001 to Z00512)
Bottom (n)		Holding register: 400001 to 409999 (W00001 to W09999)  Constant register: 700001 to 704096 (K00001 to K04096)  Link register: R10001 to R11024 R20001 to R21024  #00001 to #00016

## 2. NORMALLY CLOSED BIT (NCBT)

### A. Function

- 1) The logic state of a specified bit in a specified register is tested and output 1 is turned ON or OFF accordingly.
- 2) The same operation is performed as that for a normally closed contact relay element. For a normally closed contact, power flow is controlled based on the ON/OFF status of a coil. With **NORMALLY CLOSED BIT**, power flow is controlled based on the logic state (1/0) of a specified bit.

### B. Structure

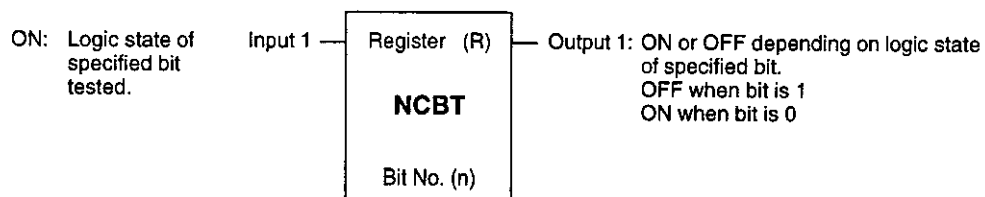


Table 4.54 Structural Elements of NCBT

Element	Meaning	Possible Settings
Top (R)	The logic state of the bit n in R, the register specified for the top element, is tested and output 1 is turned ON or OFF accordingly.	Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Bottom (n)		#00001 to #00016

3. NORMAL BIT (NBIT)

A. Function

- 1) The logic state of a specified bit in a specified register is set to 1 or reset to 0.
- 2) The same operation is performed as that for a coil relay element. For a coil, power flow controls the ON/OFF status of a coil. With NORMAL BIT, power flow controls the logic state (1/0) of a specified bit.

B. Structure

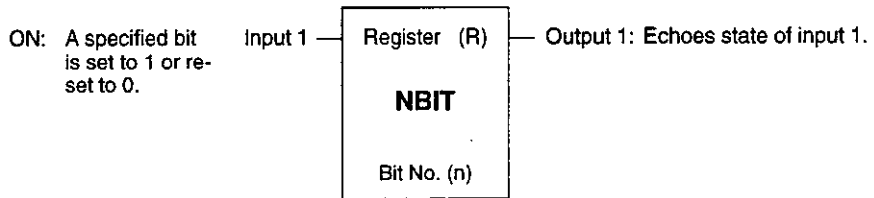


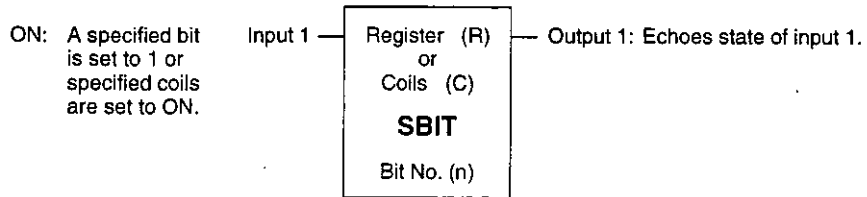
Table 4.55 Structural Elements of NBIT

Element	Meaning	Possible Settings
Top (R)	The logic state of the bit n in R, the register specified for the top element, is set to 1 or reset to 0.	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 R20001 to R21024
Bottom (n)		#00001 to #00016

4. SET BIT (SBIT)

A. Function

- 1) The logic state of a specified bit in a specified register is set to 1. SET BIT can be combined with RESET BIT to perform the same operation as that for a latched coil relay element.
- 2) The state of a specified coil in the sixteen continuous coils is set to ON. Coil multiple usage can be made available by this function. This function also can be used for link coils, MC coils, and MC control coils.

**B. Structure****Table 4.56 Structural Elements of SBIT**

Element	Meaning	Possible Settings
Top (R) (C)	The logic state of the bit n in R, the register specified for the top element, is set to 1, or the state of coils, the coils specified for the top element, is set to OFF.	Holding register: 400001 to 409999 (W00001 to W09999)  Link register: R10001 to R11024 R20001 to R21024  Coil: 000001 to 008177 (000001 to 008177)  Link coil: D10001 to D11009 D20001 to D21009  MC coil: Y10001 to Y10241 Y20001 to Y20241  MC control coil: Q10001 to Q10145 Q20001 to Q20145
Bottom (n)		#00001 to #00016

**5. RESET BIT (RBIT)****A. Function**

- 1) The logic state of a specified bit in a specified register is reset to 0. RESET BIT can be combined with SET BIT to perform the same operation as that for a latched coil relay element.
- 2) The state of a specified coil in the sixteen continuous coils is set to ON. Coil multiple usage can be made available by this function. This function also can be used for link coils, MC coils, and MC control coils.

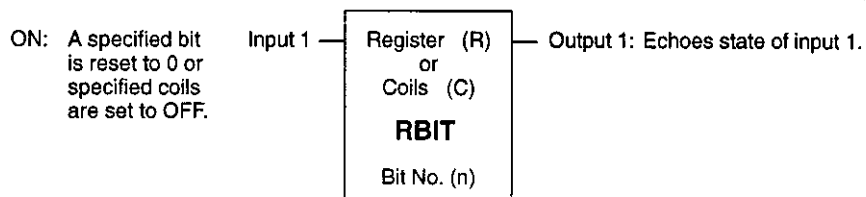
**B. Structure**



Table 4.57 Structural Elements of RBIT

Element	Meaning	Settings
Top (R) (C)	The logic state of the bit n in R, the register specified for the top element, is set to 1, or the state of coils, the coils specified for the top element, is set to OFF.	Holding register: 400001 to 409999 (W00001 to W09999)  Link register: R10001 to R11024 R20001 to R21024  Coil: 000001 to 008177 (000001 to 008177)  Link coil: D10001 to D11009 D20001 to D21009  MC coil: Y10001 to Y10241 Y20001 to Y20241  MC control coil: Q10001 to Q10145 Q20001 to Q20145
Bottom (n)		#00001 to #00016

## 4.2.7 Data Conversion Instructions

### 1. BCD-TO-BINARY CONVERSION (BIN)

#### A. Function

The data in the registers of the source table is treated as 4-digit BCD data, converted to binary data, and stored in corresponding registers of the destination table. The conversion is completed in one scan.

#### B. Structure

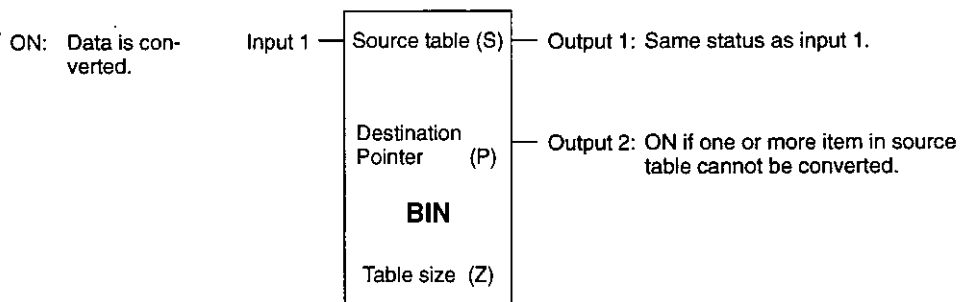


Table 4.58 Structural Elements of BIN

Element	Meaning	Possible Settings
Top (S)	Leading reference number of source table	Input register: 300001 to 300512 (Z00001 to Z00512)  Holding register: 400001 to 409999 (W00001 to W09999)  Constant register: 700001 to 704096 (K00001 to K04096)  Link register: R10001 to R11024 or R20001 to R21024
Middle (P)	Reference number of pointer	Holding register: 400001 to 409998 (W00001 to W09998)  Link register: R10001 to R11023 or R20001 to R21023
Bottom (Z)	Size of source and destination tables	Constant: #00001 to #00016

## 2. BINARY-TO-BCD CONVERSION (BCD)

### A. Function

The data in the registers of the source table is converted from binary data to 4-digit BCD data and stored in corresponding registers of the destination table. The conversion is completed in one scan.

### B. Structure

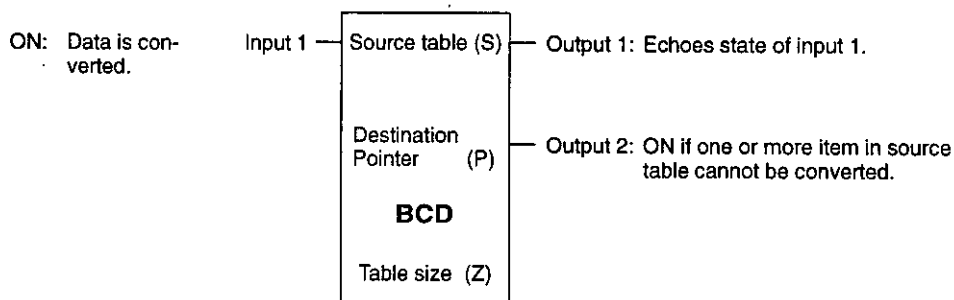


Table 4.59 Structural Elements of BCD

Element	Meaning	Possible Settings
Top (S)	Leading reference number of source table	Input register: 300001 to 300512 (Z00001 to Z00512)  Holding register: 400001 to 409999 (W00001 to W09999)  Constant register: 700001 to 704096 (K00001 to K04096)  Link register: R10001 to R11024 or R20001 to R21024
Middle (P)	Reference number of pointer	Holding register: 400001 to 409998 (W00001 to W09998)  Link register: R10001 to R11023 or R20001 to R21023
Bottom (Z)	Size of source and destination tables	Constant: #00001 to #00016

### 3. ASCII-TO-BINARY CONVERSION (ATOB)

#### A. Function

The data in each pair of registers forming a block in the source table is treated as four ASCII characters and converted to 4-bit binary data and stored in corresponding registers of the destination table. The conversion is completed in one scan. The only ASCII characters that can be converted are 0 to 9 and A to F.

#### B. Structure

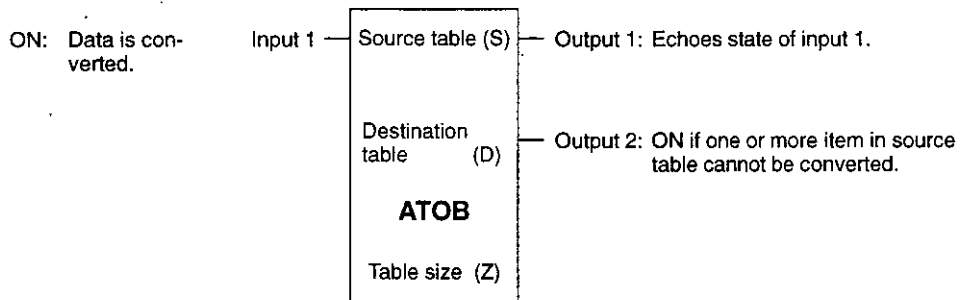


Table 4.60 Structural Elements of ATOB

Element	Meaning	Possible Settings
Top (S)	Leading reference number of source table	Input register: 300001 to 300511 (Z00001 to Z00511) Holding register: 400001 to 409998 (W00001 to W09998) Constant register: 700001 to 704095 (K00001 to K04095) Link register: R10001 to R11023 R20001 to R21023
Middle (D)	Leading reference number of destination table	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 R20001 to R21024
Bottom (Z)	Size of destination table	Constant: #00001 to #00100

#### 4. BINARY-TO-ASCII CONVERSION (BTOA)

##### A. Function

The data in each register of the source table is separated into 4 sets of 4-bit binary data, converted into four ASCII characters, and stored in corresponding blocks of the destination table. The conversion is completed in one scan.

##### B. Structure

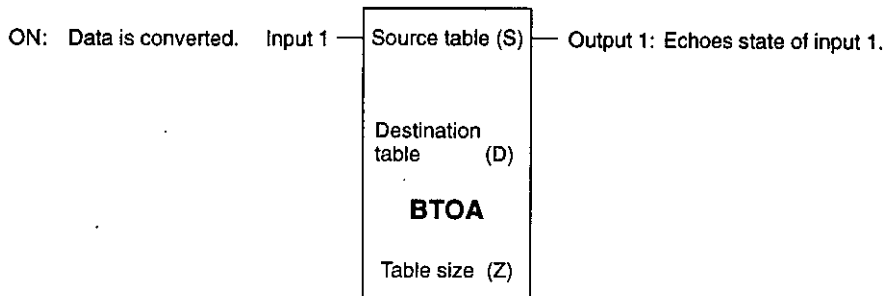


Table 4.61 Structural Elements of BTOA

Element	Meaning	Possible Settings
Top (S)	Leading reference number of source table	Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Middle (D)	Leading reference number of destination table	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 R20001 to R21023
Bottom (Z)	Size of source table	Constant: #00001 to #00100

## 5. 16-BIT CONVERSION (CAST)

### A. Function

The numeric expression of a 16-bit binary integer is changed from type 1 to type 2 or from type 2 to type 1. The conversion is completed in one scan.

### B. Structure

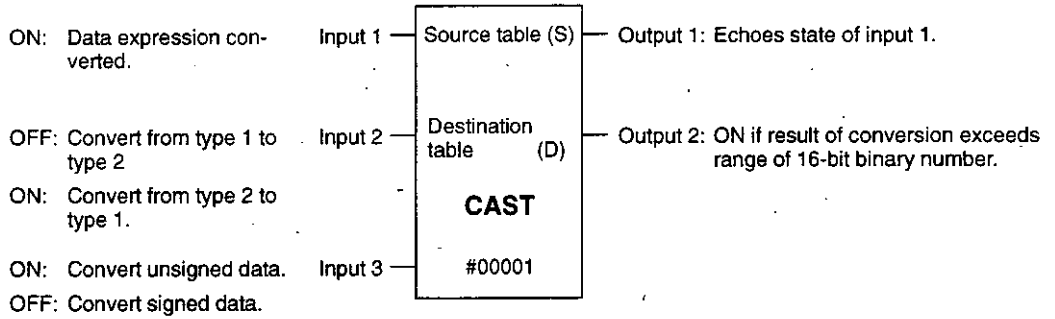


Table 4.62 Structural Elements of CAST

Element	Meaning	Possible Settings
Top (S)	Leading reference number of source table (size:2)	Input register: 300001 to 300511 (Z00001 to Z00511) Holding register: 400001 to 409998 (W00001 to W09998) Constant register: 700001 to 704095 (K00001 to K04095) Link register: R10001 to R11023 R20001 to R21023
Middle (D)	Leading reference number of destination table (size:2)	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 R20001 to R21023
Bottom	Fixed	Constant: #00001

## 6. 32-BIT CONVERSION (DCST)

### A. Function

The numeric expression of a signed or unsigned 8-digit decimal integer is changed from type 1 to type 2 or from type 2 to type 1. The conversion is completed in one scan.

### B. Structure

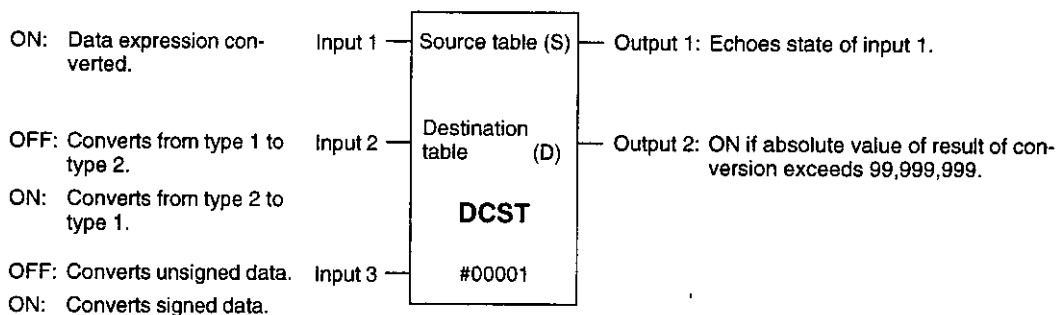


Table 4.63 Structural Elements of DCST

Element	Meaning	Possible Settings
Top (S)	Leading reference number of source table (size:2)	Input register: 300001 to 300511 (Z00001 to Z00511) Holding register: 400001 to 409998 (W00001 to W09998) Constant register: 700001 to 704095 (K00001 to K04095) Link register: R10001 to R11023 R20001 to R21023
Middle (D)	Leading reference number of destination table (size:2)	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 R20001 to R21023
Bottom	Fixed	Constant: #00001

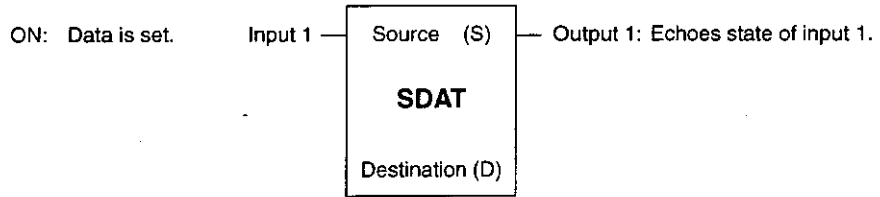
## 4.2.8 Other Data Manipulation Instructions

### 1. SET WORD DATA (SDAT)

#### A. Function

Word data (16-bit data) is set in a register. Execution of the instruction is completed in one scan. This instruction can be thought of as one type of data transfer instruction.

**B. Structure**



**Table 4.64 Structural Elements of SDAT**

Element	Meaning	Possible Settings
Top (S)	Reference number of source	Constant: #00000 to #65535 Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 or R20001 to R21024
Bottom (D)	Reference number of destination	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 or R20001 to R21024

**2. SET DOUBLE WORD DATA (SDDT)**

**A. Function**

A 32-bit binary integer is set in two consecutive registers using a type 2 numeric expression. Execution of the instruction is completed in one scan. This instruction can be thought of as one type of data transfer instruction.

**B. Structure**

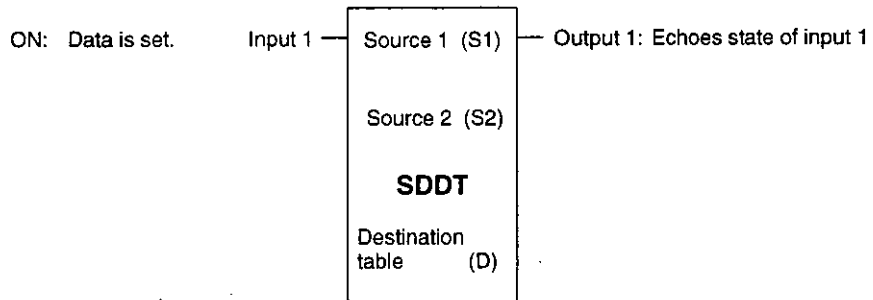


Table 4.65 Structural Elements of SDDT

Element	Meaning	Possible Settings						
Top (S1)	Specify the upper 16 bits of the 32-bit data between 0 and 65,535.	Constant: #00000 to #65535						
Middle (S2)	Specify the lower 16 bits of the 32-bit data between 0 and 65,535.							
Bottom (D)	Leading reference number of destination table. The 32-bit data is set as follows (table size fixed at 2):  <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>D+1</td> <td>Upper 16 bits</td> <td>2nd</td> </tr> <tr> <td>D</td> <td>Lower 16 bits</td> <td>1st</td> </tr> </table>	D+1	Upper 16 bits	2nd	D	Lower 16 bits	1st	Holding register: 400001 to 409998 (W00001 to W09998)  Link register: R10001 to R11023 or R20001 to R21023
D+1	Upper 16 bits	2nd						
D	Lower 16 bits	1st						

### 3. LOGICAL BYTE REARRANGEMENT (TWST)

#### A. Function

Each register in the destination table is separated into upper and lower bytes and then the order of the bits within each byte is placed in reverse order. The rearrangement is completed in one scan.

#### B. Structure

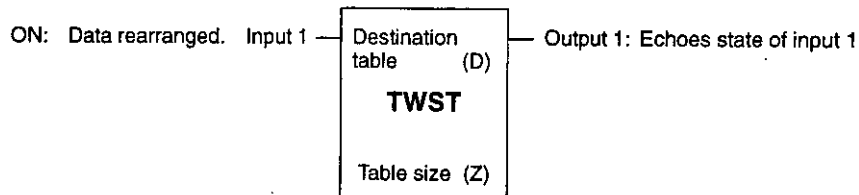


Table 4.66 Structural Elements of TWST

Element	Meaning	Possible Settings
Top (D)	Leading reference number in destination table	Holding register: 400001 to 409999 (W00001 to W09999)  Link register: R10001 to R11024 R20001 to R21024
Bottom (Z)	Size of destination table	Constant: #00001 to #00100

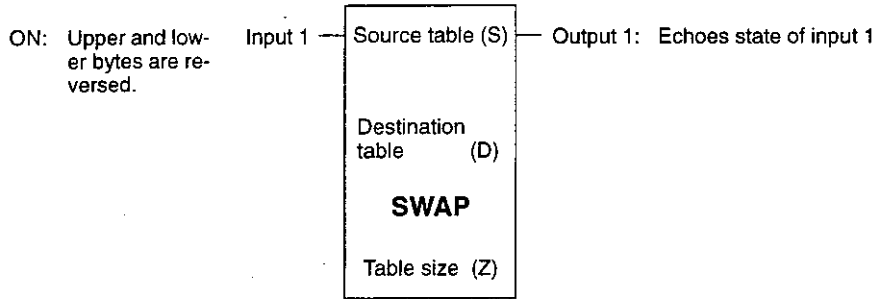
### 4. SWAP (SWAP)

#### A. Function

Each register in the source table is separated into upper and lower bytes and then the order of the bytes is reversed and stored in the corresponding register of the destination table. The rearrangement is completed in one scan.



**B. Structure**



**Table 4.67 Structural Elements of SWAP**

Element	Meaning	Possible Settings
Top (S)	Leading reference number in source table	Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W099999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Middle (D)	Leading reference number in destination table	Holding register: 400001 to 409999 (W00001 to W099999) Link register: R10001 to R11024 R20001 to R21024
Bottom (Z)	Size of source and destination tables	Constant: #00001 to #00100

**5. SORT (SORT)**

**A. Function**

The data in each register of the source table or of the destination table is treated as 16-bit binary data (0 to 65,535), the data is sorted into ascending or descending order, and then the sorted result is stored. The sort is completed in one scan.

**B. Structure**

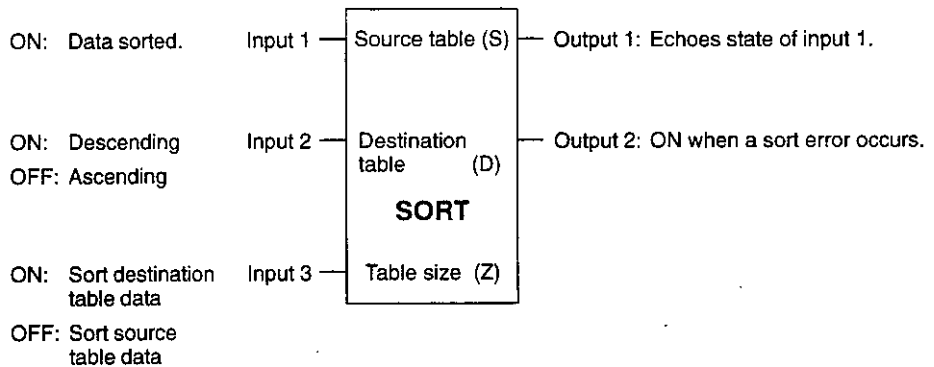


Table 4.68 Structural Elements of SORT

Element	Meaning	Possible Settings
Top (S)	Leading reference number in source table	Input register: 300001 to 300512 (Z00001 to Z00512)  Holding register: 400001 to 409999 (W00001 to W09999)  Constant register: 700001 to 704096 (K00001 to K04096)  Link register: R10001 to R11024 R20001 to R21024
Middle (D)	Leading reference number in destination table	Holding register: 400001 to 409999 (W00001 to W09999)  Link register: R10001 to R11024 R20001 to R21024
Bottom (Z)	Size of source and destination tables	Constant: #00001 to #00100

## 6. BYTE SPLIT (BYSL)

### A. Function

The data (word data) in each register of the source table is split into bytes and then the bytes are stored in the two registers of the corresponding block in the destination table. The split is completed in one scan.

### B. Structure

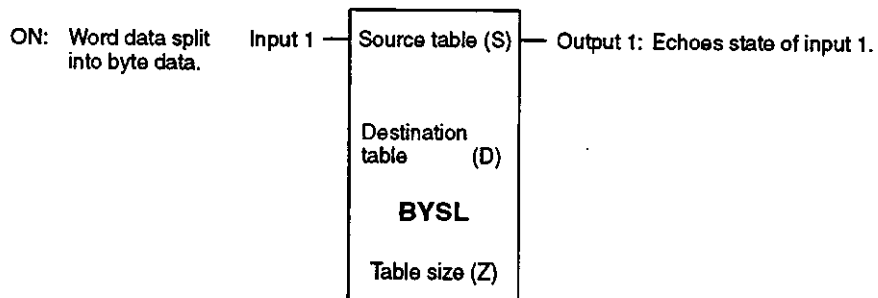


Table 4.69 Structural Elements of BYSL

Element	Meaning	Possible Settings
Top (S)	Leading reference number in source table	Input register: 300001 to 300512 (Z00001 to Z00512)  Holding register: 400001 to 409999 (W00001 to W09999)  Constant register: 700001 to 704096 (K00001 to K04096)  Link register: R10001 to R11024 R20001 to R21024
Middle (D)	Leading reference number in destination table	Holding register: 400001 to 409998 (W00001 to W09998)  Link register: R10001 to R11023 R20001 to R21023
Bottom (Z)	Size of source table	Constant: #00001 to #00100

## 7. BYTE COMPOSITION (BYCM)

### A. Function

The data in the lower bytes of the two registers in each block in the source table is combined into word data and the data is stored in the corresponding register in the destination table. Execution is completed in one scan.

### B. Structure

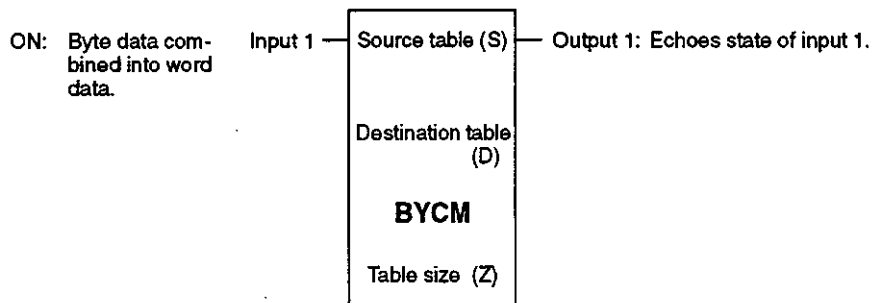


Table 4.70 Structural Elements of BYCM

Element	Meaning	Possible Settings
Top (S)	Leading reference number in source table	Input register: 300001 to 300511 (Z00001 to Z00511)  Holding register: 400001 to 409998 (W00001 to W09998)  Constant register: 700001 to 704095 (K00001 to K04095)  Link register: R10001 to R11023 R20001 to R21023
Middle (D)	Leading reference number in destination table	Holding register: 400001 to 409999 (W00001 to W09999)  Link register: R10001 to R11024 R20001 to R21024
Bottom (Z)	Size of destination table	Constant: #00001 to #00100

## 8. NIBBLE SPLIT (NBSL)

### A. Function

The data (word data) for each register or 16 coils/relays of the source table is split into four nibbles (4-bit data) and then the nibbles are stored in the four registers of the corresponding block in the destination table. The split is completed in one scan.

### B. Structure

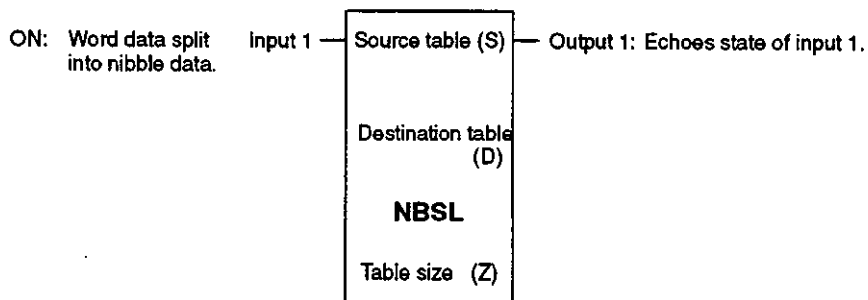


Table 4.71 Structural Elements of NBSL

Element	Meaning	Possible Settings
Top (S)	Leading reference number of source table	Coil: 000001 to 008177 (O00001 to O08177) Input relay: 100001 to 101009 (I00001 to I01009) Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145 MC relay: X10001 to X10241 or X20001 to X20241 MC control relay: P10001 to P10241 or P20001 to P20241 M code relay: M10001 to M10081 or M20001 to M20081
Middle (D)	Leading reference number of destination table	Holding register: 400001 to 409996 (W00001 to W09996) Link register: R10001 to R11021 or R20001 to R21021
Bottom (Z)	Size of source table	A constant must be specified. The maximum value of the constant that can be specified depends on the type of reference that is used. Coil: #00001 to #00100 Input relay: #00001 to #00064 Input register, holding register or constant register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil, MC relay or MC control relay: #00001 to #00016 MC control coil: #00001 to #00010 M code relay: #00001 to #00006

## 9. NIBBLE COMPOSITION (NBCM)

### A. Function

The data in the lower four bits (nibbles) of the four registers in each block in the source table is combined into word data and the data is stored in the corresponding register or 16 coils/relays in the destination table. Execution is completed in one scan.

## B. Structure

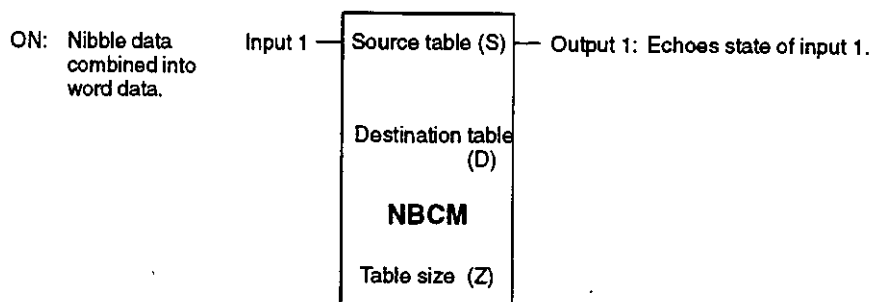


Table 4.72 Structural Elements of NBCM

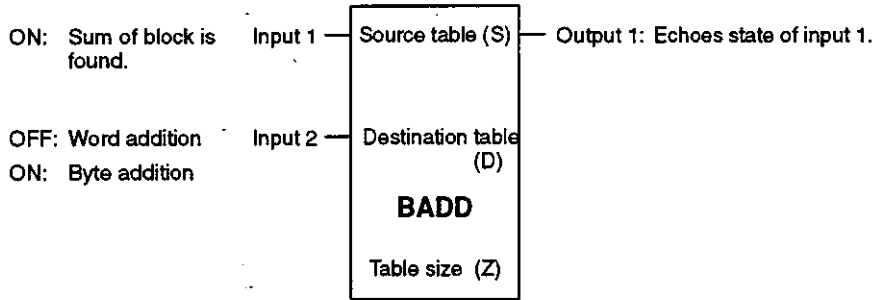
Element	Meaning	Possible Settings
Top (S)	Leading reference number of source table	Input register: 300001 to 300509 (Z00001 to Z00509) Holding register: 400001 to 409996 (W00001 to W09996) Constant register: 700001 to 704093 (K00001 to K04093) Link register: R10001 to R11021 or R20001 to R21021
Middle (D)	Leading reference number of destination table	Coil: 000001 to 008177 (O00001 to O08177) Holding register: 400001 to 409999 (W00001 to W09999) Link coil: D10001 to D11009 or D20001 to D21009 Link register: R10001 to R11024 or R20001 to R21024 MC coil: Y10001 to Y10241 or Y20001 to Y20241 MC control coil: Q10001 to Q10145 or Q20001 to Q20145
Bottom (Z)	Size of destination table	A constant must be specified. maximum value of constant that can be specified depends on type of reference that is used. Coil: #00001 to #00100 Holding register: #00001 to #00100 Link coil: #00001 to #00064 Link register: #00001 to #00100 MC coil: #00001 to #00016 MC control coil: #00001 to #00010

## 10. BLOCK ADD (BADD)

### A. Function

The data in registers of the source table is added by word or by byte in unsigned addition and the result is stored in the two registers of the destination table. Execution is completed in one scan.

**B. Structure**



**Table 4.73 Structural Elements of BADD**

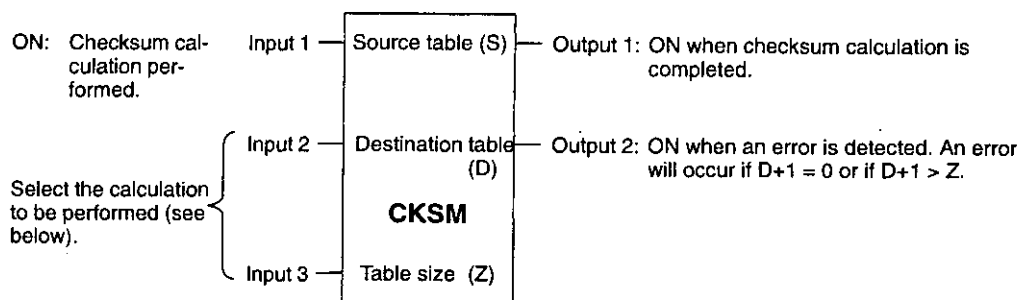
Element	Meaning	Possible Settings
Top (S)	Leading reference number in source table	Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Middle (D)	Leading reference number in destination table	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 R20001 to R21023
Bottom (Z)	Size of source table	Constant: #00001 to #00100

**11. CHECKSUM (CKSM)**

**A. Function**

Straight check calculation, binary addition check calculation, CRC-16 (cyclic redundancy check) calculation, or LRC (longitudinal redundancy check) calculation is performed for the data in the source table. Execution is completed in one scan.

## B. Structure



The type of checksum calculation to be performed is specified by combining inputs 2 and 3.

Input 2	Input 3	Checksum Calculation
OFF	ON	Straight check
ON	ON	Binary addition check
ON	OFF	CRC-16 (cyclic redundancy)
OFF	OFF	LRC (longitudinal redundancy)

Table 4.74 Structural Elements of CKSM

Element	Meaning	Possible Settings
Top (S)	Leading reference number in source table	Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Middle (D)	Leading reference number in destination table (size: 2)	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 R20001 to R21023
Bottom (Z)	Size of source table	Constant: #00001 to #01000

## 4.2.9 System Status Monitoring Instruction

### SYSTEM STATUS MONITORING (STAT)

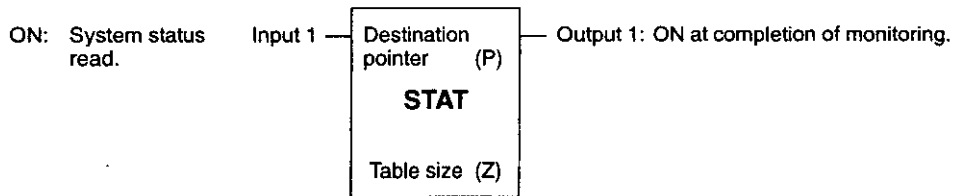
#### 1. Function

- 1) The data stored in the system status table in the CPU Module is read. The system status table is part of system memory.



- 2) The system status table contains data on the following Modules.
  - a) CPU Module
  - b) Communications Modules
  - c) I/O Modules
  - d) Four-axis Motion Modules
  - e) Motion Modules
- 3) SYSTEM STATUS MONITORING can be thought of as a type of data transfer instructions, similar to those introduced earlier in this manual.
- 4) The value of the pointer and the size of the destination table can be set so that the status data from any particular block of the source table (system status table) can be transferred to the destination table. The transfer is completed in one scan.

**2. Structure**



**Table 4.75 Structural Elements of STAT**

Element	Meaning	Possible Settings
Top (P)	Reference number of the pointer	Holding register: 400001 to 409998 (W00001 to W09998) Link register: R10001 to R11023 R20001 to R21023
Bottom (Z)	Size of the destination table	Constant: #00001 to #00256

**4.2.10 Sequencers**

**1. Function**

**1) Sequencer Numbers**

Thirty-two stepping sequencers are provided. The sequencers are identified by sequencer numbers running from 1 to 32, e.g., sequencer 1, sequencer 2, etc.

## 2) Step Numbers

Each stepping sequencer contains a maximum of 99 steps. The steps are identified by step numbers running from 1 to 99, e.g., step 1, step 2, etc.

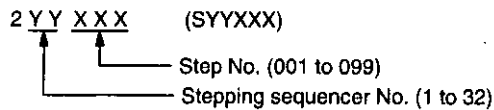
## 3) Sequencer Control Registers

Stepping sequencer 1 is controlled by the content of holding register 402001; sequencer 2, by the content of holding register 402032, etc. These registers are called stepping sequencer control registers. The contents of these registers can be manipulated with timer, counter, arithmetic, or other instructions to advance the stepping sequencers, to implement returns, jumps, etc.

## 2. Structure

### 1) Reference Numbers

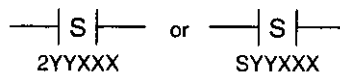
Reference numbers are allocated to stepping sequencers as shown below. For example, 201032 is the reference number allocated to step 32 of sequencer 1. The reference numbers in parentheses are used for alphanumeric numbers.



### 2) Contacts

Stepping sequencers are used in ladder programming through N.O. and N.C. contacts. Transitional contacts cannot be used. The structure of the N.O. and N.C. contacts for stepping sequencers is shown below.

#### a) N.O. Contacts for Stepping Sequencers



#### b) N.C. Contacts for Stepping Sequencers

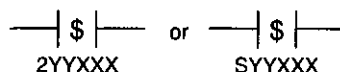


Table 4.76 Operation of Stepping Sequencer

Stepping Sequencer No.	Control Register Reference No. (See Note.)	Contents of Stepping Sequencer Control Register				
		1	2	3	—	99
1	402001	Only 201001 ON	Only 201002 ON	Only 201003 ON	—	Only 201099 ON
2	402002	Only 202001 ON	Only 202002 ON	Only 202003 ON	—	Only 202099 ON
3	402003	Only 203001 ON	Only 203002 ON	Only 203003 ON	—	Only 203099 ON
—	—	—	—	—	—	—
32	402032	Only 232001 ON	Only 232002 ON	Only 232003 ON	—	Only 232099 ON

Note Holding registers 402001 to 402032 can be used as normal holding registers when they are not being used to control stepping sequencers.

## 4.2.11 Program Control Instructions

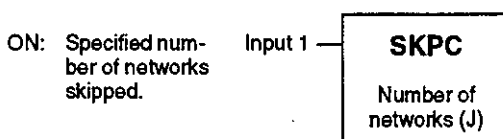
### 1. Skip Node Instructions (SKPC, SKPR)

#### A. Function

- 1) The skip node instructions are used to freeze solving the program of a specified number of networks by skipping over them. None of the ladder logic programming in the skipped networks will be solved.
- 2) The skip node instructions can be used to reduce the scan time, to give solving specific networks higher priority, or to create subroutines in the high-speed and normal segments.
- 3) There are two skip node instructions. The function and operation of these instructions are the same.
  - a) **SKIP CONSTANT (SKPC):** The number of networks to be skipped is specified as a constant between 0 and 9,999.
  - b) **SKIP REGISTER (SKPR):** The number of networks to be skipped is specified as the contents of a register between 0 and 65,535.

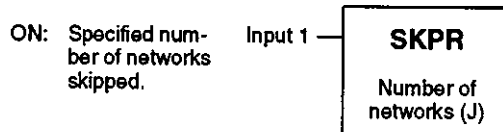
#### B. Structure

##### 1) SKIP CONSTANT



- 1) SKPC is the symbol for SKIP CONSTANT.
- 2) SKPC requires one element (bottom) on the network. Specify a constant between #00000 and #09999 to specify the number of networks to be skipped.

### 3) SKIP REGISTER



- 1) SKPR is the symbol for SKIP REGISTER.
- 2) SKPR requires one element (bottom) on the network. Refer to *Table 4.77* to specify the reference number of a register. The contents of the specified register is the number of networks to be skipped.

**Table 4.77 Structural Elements of SKPR**

Element	Meaning	Possible Settings
Bottom (J)	The contents of the specified register provide the number of networks (J) to be skipped. J can be between 0 and 65,535.	Input register: 300001 to 300512 (Z00001 to Z00512) Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024

## 2. Subroutine Instructions

### A. Subroutines

- 1) Up to 1,023 subroutines can be stored in the subroutine segment.
- 2) Subroutines are identified by subroutine numbers from 1 to 1,023, e.g., subroutine 1, subroutine 2, etc.
- 3) Each subroutine is made up of ladder programming constituting one or more networks. Each subroutine can consist of as many networks as required as long as user program memory capacity is not exceeded.
- 4) Each subroutine starts with the SUBROUTINE LABEL (LAB) instruction and ends with the SUBROUTINE RETURN (RET) instruction.

5) The following illustration shows two subroutines.

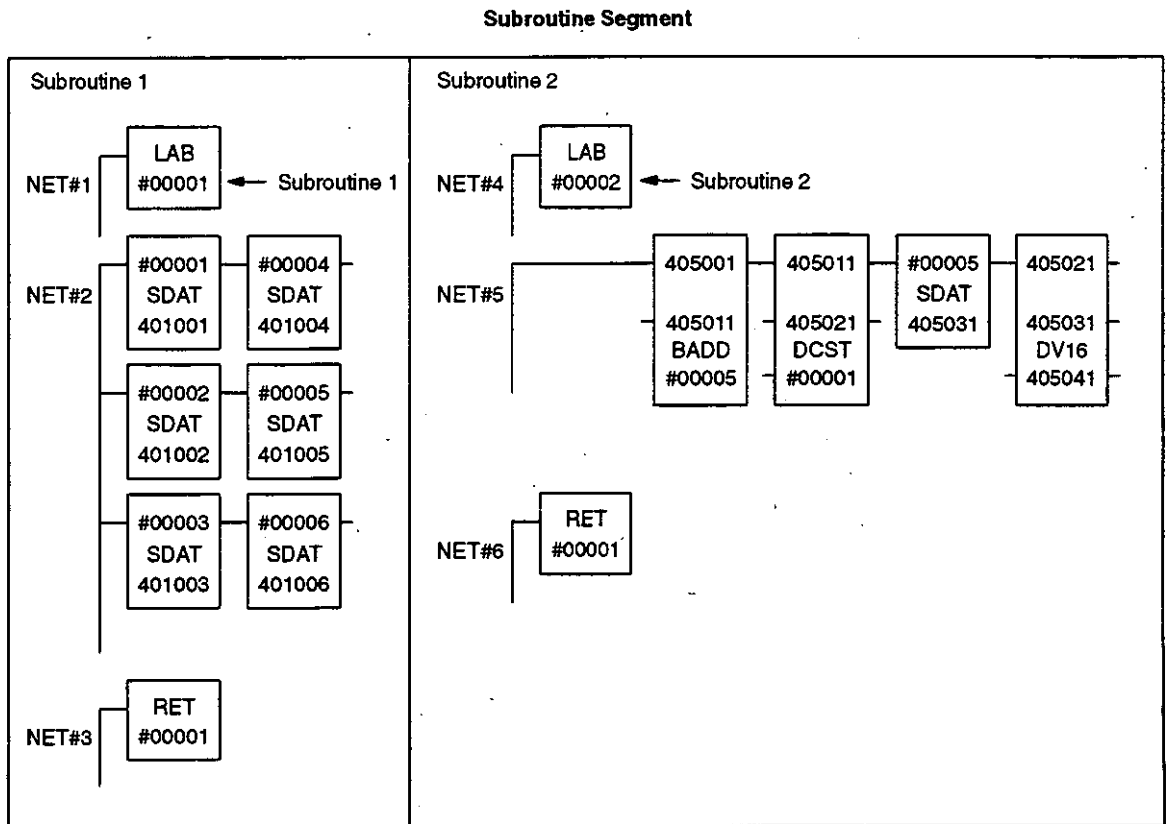


Figure 4.1 Subroutines

**B. Subroutine Instructions**

1) Three instructions are used with subroutines.

**a) SUBROUTINE JUMP (JSR)**

Used to call a subroutine.

**b) SUBROUTINE LABEL (LAB)**

Used to start a subroutine and assign it a subroutine number.

**c) SUBROUTINE RETURN (RET)**

Used to end a subroutine and return to the SUBROUTINE JUMP (JSR) instruction from which the subroutine was called.

2) Subroutines can be nested, i.e., another subroutine can be called from within a subroutine. Up to 100 subroutines can be nested. Excessive nesting should be avoided to simplify the program and avoid difficulties in debugging and troubleshooting.

3) Looping is possible by nesting subroutines, i.e., the subroutine can be repeatedly executed in one scan while a specific condition is met. Up to 100 loops can be executed. Excessive looping of subroutines with long execution times can increase the scan time, causing inconsistency in scan times or even causing the CPU to stop due to a watchdog timer error. Be careful when programming loops.

### C. Subroutine Usefulness

Subroutines can be used for the following purposes.

#### 1) Reducing User Program Memory Usage

User program memory can be saved by programming subroutines for ladder programming used repeatedly at various places in the program in one scan.

#### 2) Reducing Scan Time

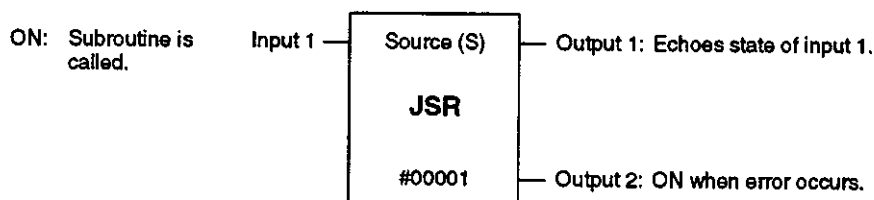
The scan time can be reduced by programming subroutines for ladder programming that is not used very often. The ladder programming in the subroutine is solved only when it is required, reducing the scan time whenever the subroutine is not executed.

### 3. SUBROUTINE JUMP (JSR)

#### A. Function

- 1) A subroutine is called, i.e., a jump is made to the specified subroutine and the ladder programming is solved.
- 2) SUBROUTINE JUMP is used together with SUBROUTINE LABEL (used to assign a subroutine number) and SUBROUTINE RETURN (used to end the subroutine).

#### B. Structure



#### JSR Storage Locations

JSR can be stored in the high-speed segment, normal segment, or the subroutine segment.

Table 4.78 Structural Elements of JSR

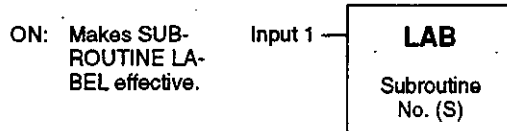
Element	Meaning	Possible Settings
Top (S)	The value of the constant or the contents of the register with the specified reference number specifies the number of the subroutine to call (1 to 1,023).	Constant: #00001 to #01023 Holding register: 400001 to 409999 (W00001 to W09999) Constant register: 700001 to 704096 (K00001 to K04096) Link register: R10001 to R11024 R20001 to R21024
Bottom	Fixed	Constant: #00001

#### 4. SUBROUTINE LABEL (LAB)

##### A. Function

- 1) A subroutine is started and assigned a subroutine number.
- 2) SUBROUTINE LABEL is used together with SUBROUTINE JUMP (used to call a subroutine number) and SUBROUTINE RETURN (used to end the subroutine).

##### B. Structure



- 1) LAB is the symbol for SUBROUTINE LABEL.
- 2) LAB requires one element (bottom) on the network. Specify a constant between #00001 and #01023 for the element. The specified constant is the subroutine number.

##### Note

###### (1) Programming Operation

LAB cannot be stored on a network, deleted, or changed while the CPU Module is running. The CPU Module must be stopped to perform any of these programming operations for LAB.

###### (2) Storage Locations

LAB must be stored on the first row and first column of a network in the subroutine segment. It cannot be stored anywhere else.

###### (3) Subroutine Numbers

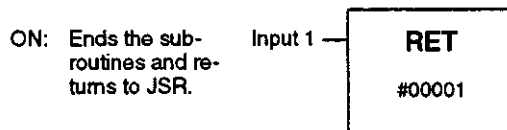
Each subroutine number (a constant between #00001 and #01023) can be used only once in a LAB instruction, i.e., each subroutine must have a unique subroutine number.

#### 5. SUBROUTINE RETURN (RET)

##### A. Function

- 1) A subroutine is ended, i.e., solving the ladder programming in the subroutine is ended and a return is made to the JSR that called the subroutine.
- 2) SUBROUTINE RETURN is used together with SUBROUTINE JUMP (used to call a subroutine number) and SUBROUTINE LABEL (used to assign a subroutine number).

##### B. Structure



- 1) RET is the symbol for SUBROUTINE RETURN.

2) RET requires one element (bottom) on the network. Specify the constant #00001 for the element.

3) **Storage Locations**

RET can be stored on any column of the first row in a network in the subroutine segment. It cannot be stored after a coil (including link coils, MC coils, and MC control coils) or in the high-speed or a normal segment.

## 6. Master Control Instructions

1) **Function**

a) Master control instructions can be used to turn OFF the power rail in specified networks. If the power rail is turned OFF in a network, the ladder programming will be solved without power flow from the power rail.

b) One example of using master control instructions is to turn OFF all the coils used in the relay circuits of the specified network.

2) **Instructions**

There are two master control instructions. These instructions are always used as a pair, as shown in the following example.

1) **MASTER CONTROL ON (MSON)**

MSON turns OFF the network's power rail.

2) **MASTER CONTROL OFF (MSOF)**

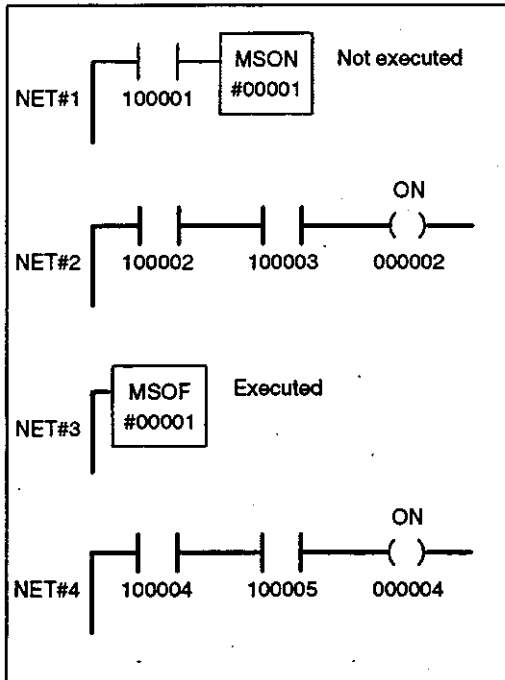
MSOF turns ON the network's power rail.



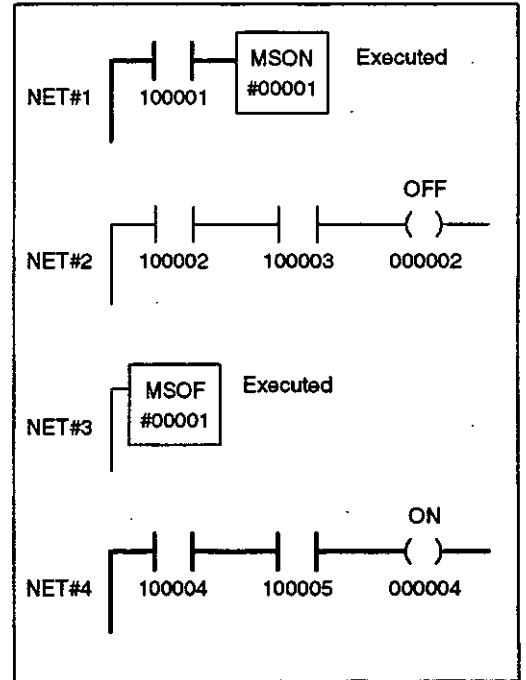
**Example**

The power rails in networks 2 and 3 will be turned OFF when input relay 100001 turns ON.

**a) Input Relay 100001 is OFF.**



**b) Input Relay 100001 is ON.**

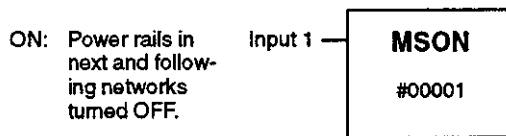


**A. MASTER CONTROL ON (MSON)**

**1) Function**

The power rail to a network is turned OFF. MASTER CONTROL ON is used in a pair with MASTER CONTROL OFF, which turns the power rail back ON.

**2) Structure**



a) MSON is the symbol for MASTER CONTROL ON.

b) MSON requires one element (bottom) on the network. Specify the constant #00001 for the element.

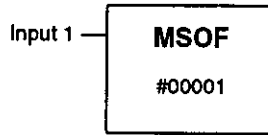
**B. MASTER CONTROL OFF (MSOF)**

**1) Function**

a) The power rail to a network is turned back ON. MASTER CONTROL OFF is used in a pair with MASTER CONTROL ON, which turns the power rail OFF.

## 2) Structure

ON or OFF:  
Power rails in next  
and following  
networks turned ON.



- a) MSOF is the symbol for MASTER CONTROL OFF.
- b) MSOF requires one element (bottom) on the network. Specify the constant #00001 for the element.

**Note** Master Control Instructions cannot be stored on a network, deleted, or moved while the CPU Module is running. The CPU Module must be stopped to perform any of these programming operations for MSOF.

## 4.2.12 I/O Control Instructions

### 1. DIRECT IN (DIN)

#### A. Function

- 1) ON/OFF data is input during the scan cycle from a Digital Input Module mounted to a local channel.
- 2) The leading register in the source table (three consecutive registers) stores the rack number (1 to 4) where the Digital Input Module to be read is mounted.
- 3) The second register in the source table stores the slot number (1 to 16) where the Digital Input Module to be read is mounted.
- 4) The third register in the source table is used to store error information resulting from execution of DIN.
- 5) The ON/OFF data read from the Digital Input Module by DIN is stored in the destination table.
- 6) Specify the size of the destination table for the table size (1 to 4).

#### B. Structure

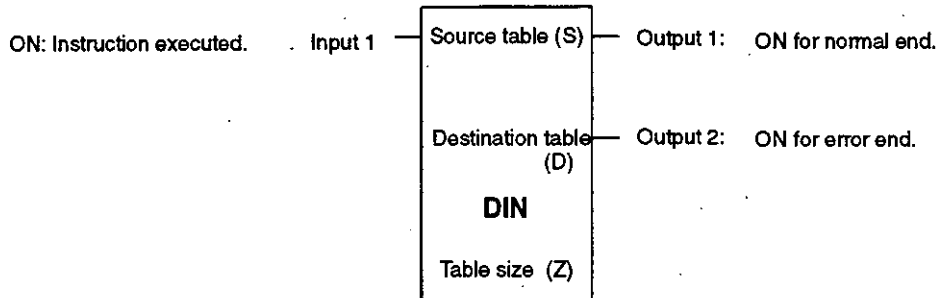


Table 4.79 Structural Elements of DIN

Element	Meaning	Possible Settings
Top (S)	Reference number of the leading register in the source table	Holding register: 400001 to 409997 (W00001 to W09997) Link register: R10001 to R11022 or R20001 to R21022 Constant register: 700001 to 704094 (K00001 to K04094)
Middle (D)	Reference number of the leading register in the destination table	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 or R20001 to R21024
Bottom (Z)	Size of the destination table	Constant: #00001 to #00004

**Note** Do not perform I/O allocation for the Digital Input Modules for which DIN is to be used. DIN will not read data from Digital Input Modules for which I/O allocation has been performed, and output 2 from DIN will turn ON if an attempt to do so is made.

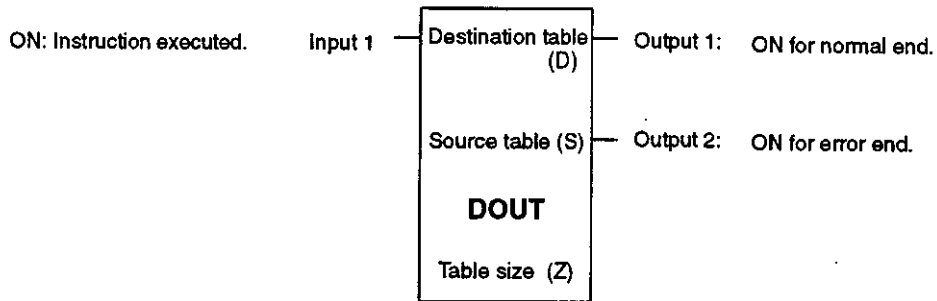
## 2. DIRECT OUT (DOUT)

### A. Function

- 1) ON/OFF data is output during the scan cycle from a Digital Output Module mounted to a local channel.
- 2) The leading register in the destination table stores the rack number (1 to 4) where the Digital Output Module to be read is mounted.
- 3) The second register in the destination table stores the slot number (1 to 16) where the Digital Output Module to be read is mounted.
- 4) The third register in the destination table is used to store error information resulting from execution of DOUT.
- 5) The ON/OFF data output from the Digital Output Module by DOUT is stored in the source table.

6) Specify the size of the source table for the table size (1 to 4).

### B. Structure



**Table 4.80 Structural Elements of DIN**

Element	Meaning	Possible Settings
Top (D)	Reference number of the leading register in the destination table	Holding register: 400001 to 409997 (W00001 to W09997) Link register: R10001 to R11022 or R20001 to R21022 Constant register: 700001 to 704094 (K00001 to K04094)
Middle (S)	Reference number of the leading register in the source table	Holding register: 400001 to 409999 (W00001 to W09999) Link register: R10001 to R11024 or R20001 to R21024
Bottom (Z)	Size of the source table	Constant: #00001 to #00004

**Note** Do not perform I/O allocation for the Digital Output Modules for which DOUT is to be used. DOUT will not output data from Digital Output Modules for which I/O allocation has been performed, and output 2 from DOUT will turn ON if an attempt to do so is made.

# Appendix **A**

---

## Instruction Processing Times

A

Table A.1 Instruction Processing Times

Instruction		Conditions		GL120 Processing Time (μs)	GL130 Processing Time (μs)			
Basic Instructions	Contact	Normally Open (N.O.) and Normally Closed (N.C.) Contacts		1.30	0.95			
	Transitional Contacts	Positive and Negative Transitional Contacts		1.80	1.30			
	Coils	Using transitional contact for coil.		9.85	7.50			
		Not using transitional contact for coil.		1.90	1.55			
	Timers	Not executed	1-, 0.1- and 0.01-s Timers		2.35	1.65		
		Executed			5.05	3.15		
		Not executed	0.001-s Timer		26.50	21.45		
		Executed			29.90	24.20		
	Counters	Not executed	UP COUNTER (UCTR)		14.25	7.50		
		Executed			15.55	7.65		
		Not executed	DOWN COUNTER (DCTR)		14.05	7.40		
		Executed			16.20	8.75		
Math Instructions	Un-signed, Four-digit, Decimal Arithmetic Instructions	UNSIGNED SINGLE PRECISION DECIMAL ADDITION (ADD)	Not executed	No overflow		1.05	1.10	
			Executed			4.45	2.70	
		UNSIGNED SINGLE PRECISION DECIMAL SUBTRACTION (SUB)	Not executed		1.05	1.10		
			Executed	4.45	2.75			
		UNSIGNED SINGLE PRECISION DECIMAL MULTIPLICATION (MUL)	Not executed		No overflow		1.05	1.10
			Executed	17.30	10.30			
	UNSIGNED SINGLE PRECISION DECIMAL DIVISION (DIV)	Not executed		Division for decimal portion		1.25	1.20	
		Executed	25.15	17.25				
	Un-signed, Eight-digit, Decimal Arithmetic Instructions	UNSIGNED DOUBLE PRECISION DECIMAL ADDITION (DADD)	Not executed		1.05	1.10		
			Executed	22.50	12.90			
		UNSIGNED DOUBLE PRECISION DECIMAL SUBTRACTION (DSUB)	Not executed		1.05	1.10		
			Executed	23.60	13.00			

Instruction		Conditions		GL120 Processing Time ( $\mu$ s)	GL130 Processing Time ( $\mu$ s)	
Math Instructions	Unsigned, Eight-digit, Decimal Arithmetic Instructions	UNSIGNED DOUBLE PRECISION DECIMAL MULTIPLICATION (DMUL)	Not executed		1.05	1.10
			Executed		46.95	22.90
		UNSIGNED DOUBLE PRECISION DECIMAL DIVISION (DDIV)	Not executed		1.25	1.20
			Executed	Division for remainder	231.70	26.45
				Division for decimal portion	416.15	29.90
				Quotient overflow	38.35	20.75
	Signed, Four-digit, Decimal Arithmetic Instructions	SIGNED SINGLE PRECISION DECIMAL ADDITION (SADD)	Not executed		1.05	1.10
			Executed	$0 + 0$	21.85	9.85
				$(-9999) + (-9999)$	27.70	12.45
		SIGNED SINGLE PRECISION DECIMAL SUBTRACTION (SSUB)	Not executed		1.05	1.10
			Executed	$0 - 0$	21.85	9.85
				$(-9999) - (-9999)$	23.45	10.40
		SIGNED SINGLE PRECISION DECIMAL MULTIPLICATION (SMUL)	Not executed		1.05	1.10
			Executed	$0 \times 0$	21.65	11.45
				$(-9999) \times (-9999)$	25.75	11.70
		SIGNED SINGLE PRECISION DECIMAL DIVISION (SDIV)	Not executed		1.25	1.25
			Executed	Division for decimal portion	29.10	14.95
		Signed, Eight-digit, Decimal Arithmetic Instructions	SIGNED DOUBLE PRECISION DECIMAL ADDITION (SDAD)	Not executed		1.05
	Executed			$0 + 0$	33.55	18.80
				$(99999999) + (99999999)$	34.45	18.95
$(-99999999) - (-99999999)$				36.30	17.45	
SIGNED DOUBLE PRECISION DECIMAL SUBTRACTION (SDSB)	Not executed		1.05	1.10		
	Executed		$0 - 0$	33.55	18.80	
		$(99999999) - (99999999)$	33.55	19.90		
$(-99999999) - (-99999999)$		34.45	18.70			
Decimal Square Root Instructions	SINGLE PRECISION DECIMAL SQUARE ROOT (SQRT)	Not executed		1.05	1.10	
		Executed	$\sqrt{0}$	228.50	162.60	
			$\sqrt{9999}$	235.80	162.50	
	DOUBLE PRECISION DECIMAL SQUARE ROOT (DSQR)	Not executed		1.05	1.10	
Executed		$\sqrt{0}$	308.00	221.55		
		$\sqrt{99999999}$	319.60	221.70		
Decimal Trigonometric Instruction	DECIMAL SINE (SIN)	Not executed		1.05	1.10	
		Executed	SIN ( $90^\circ$ )	32.50	20.55	
			SIN ( $0.0001^\circ$ )	397.15	296.30	
	DECIMAL COSINE (COS)	Not executed		1.05	1.10	
		Executed	COS ( $0^\circ$ )	31.40	19.95	
	COS ( $89.9999^\circ$ )		395.65	295.95		

Appendix A Instruction Processing Times

Instruction			Conditions			GL120 Processing Time (μs)	GL130 Processing Time (μs)
Math Instructions	Sixteen-bit Arithmetic Instructions	16-BIT ADDITION (AD16)	Not executed			1.25	1.20
			Executed	Unsigned, No overflow		14.50	3.55
				Signed, No overflow		14.25	3.20
		16-BIT SUBTRACTION (SB16)	Not executed			1.35	1.25
			Executed	Unsigned		15.00	3.85
				Signed		14.50	3.75
		16-BIT MULTIPLICATION (MU16)	Not executed			1.05	1.00
			Executed	Unsigned, signed			
		16-BIT DIVISION (DV16)	Not executed			1.25	1.20
			Unsigned	Executed	Integer	19.15	14.05
					Overflow error	18.40	13.80
			Signed	Executed	Integer	25.85	18.10
	Overflow error				22.00	15.30	
	Thirty-two-bit Arithmetic Instructions		32-BIT ADDITION (AD32)	Not executed			1.25
		Executed		Unsigned, No overflow		10.50	4.05
				Signed, No overflow		9.65	3.70
		32-BIT SUBTRACTION (SB32)	Not executed			1.25	1.25
			Executed	Unsigned, No overflow		11.50	4.25
				Signed, No overflow		10.95	4.15
		32-BIT COMPARE (TEST)	16-BIT COMPARE	Not executed			1.35
Executed				Unsigned, OPR1 = OP2		12.20	2.35
				Unsigned, OPR1 < OP2		14.50	4.35
				Unsigned, OPR1 > OP2		15.10	4.50
				Signed, OPR1 = OP2		12.20	2.35
				Signed, OPR1 < OP2		14.65	3.85
	Signed, OPR1 > OP2		14.05	4.75			
32-BIT COMPARE	Not executed			1.45	1.25		
	Executed		Unsigned, OPR1 = OP2		18.30	2.55	
			Unsigned, OPR1 < OP2		18.20	4.45	
			Unsigned, OPR1 > OP2		18.35	4.60	
			Signed, OPR1 = OP2		19.50	2.55	
		Signed, OPR1 < OP2		18.25	3.90		
Signed, OPR1 > OP2		18.50	4.80				
Data Transfer Instructions	REGISTER-TO-TABLE MOVE (R→T)		Not executed			13.15	9.45
	TABLE-TO-REGISTER MOVE (T→R)	TABLE-TO-TABLE MOVE (T→T)	Executed	Pointer < Table size	16.45	11.00	
				Pointer = Table size	17.45	12.00	
				Pointer not updated (pointer incrementing disabled) (input 2 ON)	15.50	10.85	
				Pointer = 0 (input 3 ON)	14.15	10.45	



Instruction		Conditions	GL120 Processing Time (μs)	GL130 Processing Time (μs)	
Data Transfer Instruc- tions	FIRST IN (FIN)	Not executed	13.70	10.20	
		Executed	Table size = 10	21.95	15.00
			Table size = 50	37.95	23.00
			Table size = 100 (Max.)	57.95	33.00
			Pointer ≥ Table size	14.60	10.65
	FIRST OUT (FOUT)	Not executed	13.65	10.10	
		Executed	Table size = 10	16.40	10.70
			Table size = 50	16.40	10.70
			Table size = 100 (Max.)	16.40	10.70
			Pointer ≥ Table size	14.30	10.55
	TABLE SEARCH (SRCH)	Not executed	14.25	9.35	
		Executed	Table size of 100, match at 1st entry	18.10	11.75
			Table size of 100, match at 100th entry	92.50	56.70
			Table size of 100, no match	92.70	56.75
	TABLE SET (TSET)	Not executed	1.05	1.10	
		Executed	Table size = 1	12.80	4.30
			Table size = 50	34.85	16.55
			Table size = 100 (Max.)	57.35	29.05
	BLOCK MOVE (BLKM) *	Not executed	1.05 (42.50)	1.10 (22.05)	
		Executed	Table size = 1	2.65 (14.05)	1.85 (9.90)
			Table size = 50	22.25 (53.25)	11.65 (29.50)
			Table size = 100 (Max.)	42.25 (93.25)	21.65 (49.50)
	BLOCK-TO-TABLE MOVE (BLKT)	Not executed	17.15	9.55	
		Executed	Table size = 1	37.95	13.30
Table size = 50			57.55	23.30	
Table size = 100			77.55	33.35	
TABLE-TO-BLOCK MOVE (TBLK)	Not executed	12.80	9.60		
	Executed	Table size = 1	20.60	13.30	
		Table size = 50	40.20	23.30	
		Table size = 100	60.20	33.35	
INDIRECT BLOCK WRITE (IBKW)	Not executed	1.05	1.10		
	Executed	Table size = 1	22.00	15.55	
		Table size = 50	365.00	255.65	
		Table size = 100	715.00	500.65	
		Table size = 255	1800.00	1260.15	
INDIRECT BLOCK READ (IBKR)	Not executed	1.05	1.10		
	Executed	Table size = 1	19.10	15.10	
		Table size = 50	195.50	125.35	
		Table size = 100	375.50	237.85	
		Table size = 255	933.90	586.60	

\*Values in parentheses are for processing discrete bits.

Appendix A Instruction Processing Times

Instruction		Conditions		GL120 Processing Time (μs)	GL130 Processing Time (μs)	
Indexed Block Transfer Instructions	DESTINATION INDEXED BLOCK TRANSFER 1 (DIBT)	Not executed		1.45	1.30	
		Executed	Table size = 1	39.45	30.10	
			Table size = 50	174.20	120.75	
	Table size = 100		311.70	213.25		
	DESTINATION INDEXED BLOCK TRANSFER 2 (DIBR)	Not executed		1.45	1.30	
		Executed	Table size = 1	35.50	28.25	
			Table size = 50	177.60	118.90	
	Table size = 100		322.60	211.40		
	SOURCE INDEXED BLOCK TRANSFER 1 (SIBT)	Not executed		1.45	1.30	
		Executed	Table size = 1	40.50	31.60	
			Table size = 50	175.25	124.70	
	Table size = 100		312.75	219.70		
SOURCE INDEXED BLOCK TRANSFER 2 (SIBR)	Not executed		1.45	1.30		
	Executed	Table size = 1	36.20	27.65		
		Table size = 50	178.30	120.75		
Table size = 100		323.30	215.75			
Matrix Instructions	LOGICAL AND (AND)	Register	Not executed		1.05	1.25
			Executed	Table size = 1	3.75	3.20
				Table size = 50	123.80	74.25
		Table size = 100		246.30	146.75	
		Digital	Not executed		42.45	21.75
			Executed	Table size = 1	14.80	10.25
	Table size = 50			154.45	88.65	
	Table size = 100	296.95		168.65		
	LOGICAL OR (OR)	Register	Not executed		1.05	1.25
			Executed	Table size = 1	3.75	3.20
				Table size = 50	123.80	74.25
		Table size = 100		246.30	146.75	
		Digital	Not executed		42.45	21.75
			Executed	Table size = 1	14.80	11.20
	Table size = 50			154.45	99.40	
Table size = 100	296.95	189.40				
LOGICAL EXCLUSIVE OR (XOR)	Register	Not executed		1.05	1.25	
		Executed	Table size = 1	3.75	3.20	
			Table size = 50	123.80	74.25	
	Table size = 100		246.30	146.75		
	Digital	Not executed		42.45	21.75	
		Executed	Table size = 1	14.80	10.35	
Table size = 50			154.45	98.55		
Table size = 100	296.95		188.55			

Instruction		Conditions		GL120 Processing Time (μs)	GL130 Processing Time (μs)		
Matrix Instruc- tions	LOGICAL COMPLEMENT (COMP)	Register	Not executed		1.05	1.25	
			Executed	Table size = 1		3.50	3.15
				Table size = 50		103.95	66.85
				Table size = 100		206.45	131.85
		Digital	Not executed		42.45	21.75	
			Executed	Table size = 1		14.45	10.60
				Table size = 50		132.05	84.10
				Table size = 100		252.05	159.10
	LOGICAL COMPARE (CMPR)	Not executed		2.30	2.15		
		Table size = 100	Executed	All matches		129.90	64.30
				No match at 1st entry		27.80	18.05
				No match at 800th entry		81.95	42.40
				No match at 1600th entry		136.95	67.40
	LOGICAL BIT MODIFY (MBIT)	Constant for source, registers for des- tination	Not executed		1.25	1.20	
			Executed	#600 OFF		2.80	1.95
				#600 ON		3.70	1.80
		Constant for source, digitals for destina- tion	Not executed		242.70	104.35	
			Executed	#511 OFF		263.25	116.40
				#511 ON		262.60	115.80
		Registers for source, registers for des- tination	Not executed		1.25	1.20	
Executed			#600 OFF		19.70	13.90	
			#600 ON		19.35	13.30	
Registers for source, digitals for destina- tion		Not executed		207.10	104.35		
		Executed	#511 OFF		230.65	119.00	
			#511 ON		230.30	118.40	
LOGICAL SENSE (SENS)	Not executed		2.70	2.35			
	Executed	Constant for source, registers for des- tination	#600 OFF		4.10	1.60	
			#600 ON		4.10	1.10	
		Constant for source, digitals for destina- tion	#511 OFF		4.25	1.60	
			#511 ON		4.25	1.10	

Appendix A Instruction Processing Times

Instruction		Conditions			GL120 Processing Time (μs)	GL130 Processing Time (μs)	
Matrix Instructions	LOGICAL SENSE (SENS)	Executed	Registers for source, registers for destination	#600 +1	20.70	14.45	
				#600 -1	18.00	12.05	
			Registers for source, digitals for destination	#511 +1	20.30	14.20	
				#511 -1	18.00	11.80	
	LOGICAL BIT ROTATE (BROT)	Register table size = 100	Not executed			1.25	1.20
			Executed	Shifted right	210.05	162.90	
				Rotated right	208.75	161.35	
				Shifted left	210.20	162.85	
		Rotated left		209.75	162.10		
		Digital table size = 100	Not executed			42.70	22.15
			Executed	Shifted right	256.25	185.75	
				Rotated right	254.95	184.20	
	Shifted left			256.40	185.70		
	Rotated left	255.95		184.95			
	LOGICAL MULTI-BIT ROTATE (MROT)	Not executed				1.45	1.30
		Executed	Table size = 1	Shifted right	25.30	17.65	
				Rotated right	26.50	18.00	
				Shifted left	25.20	17.75	
				Rotated left	26.40	18.10	
			Table size = 50	Shifted right	199.25	110.75	
				Rotated right	200.45	111.10	
				Shifted left	199.15	110.85	
				Rotated left	200.35	111.20	
			Table size = 100	Shifted right	376.75	205.75	
Rotated right				377.95	206.10		
Shifted left				376.65	205.85		
Rotated left	377.85			206.20			
LOGICAL BIT COUNT (BCNT)	Not executed				1.25	1.20	
	Executed	Table size = 1	0 bits counted	18.80	12.20		
			1 bits counted	19.05	12.20		
		Table size = 50	0 bits counted	190.30	107.75		
			1 bits counted	183.20	107.75		
		Table size = 100	0 bits counted	365.30	205.25		
1 bits counted			350.70	205.25			

Instruction		Conditions		GL120 Process- ing Time ( $\mu$ s)	GL130 Process- ing Time ( $\mu$ s)
Bit Manipu- lation Instructions	NORMALLY OPEN BIT (NOBT)	Executed	Specified bit OFF	1.70	1.15
			Specified bit ON	1.85	1.55
	NORMALLY CLOSED BIT (NCBT)	Executed	Specified bit OFF	1.85	1.55
			Specified bit ON	1.70	1.15
	NORMAL BIT (NBIT)	Not executed		2.30	1.75
		Executed		2.95	1.65
	SET BIT (SBIT)	Not executed	Register	1.05	1.10
		Executed	Register	1.90	1.20
RESET BIT (RBIT)	Not executed	Register	1.05	1.10	
	Executed	Register	1.90	1.20	
Data Conversion Instructions	BCD-TO-BINARY CONVERSION (BIN)	Not executed		1.45	1.30
		Executed	Table size = 1	26.15	19.10
			Table size = 8	79.00	56.20
	Table size = 16 (Max.)		139.40	98.60	
	BINARY-TO-BCD CONVERSION (BCD)	Not executed		1.45	1.30
		Executed	Table size = 1	26.65	18.55
			Table size = 8	86.15	51.80
	Table size = 16 (Max.)		154.15	89.80	
	ASCII-TO-BINARY CONVERSION (ATOB)	Not executed		1.25	1.20
		Executed	Table size = 1	21.65	15.90
			Table size = 50	406.30	221.70
	Table size = 100 (Max.)		798.80	431.70	
	BINARY-TO-ASCII CONVERSION (BTOA)	Not executed		1.05	1.10
		Executed	Table size = 1	19.00	14.30
			Table size = 50	347.30	200.50
	Table size = 100 (Max.)		682.30	390.50	
16-BIT DATA CONVERSION (CAST)	Not executed		1.45	1.20	
	Executed	No sign, to 16-bit binary	Error	18.85	10.10
			Error	18.90	9.70
	Executed	Sign, to 16-bit binary	+	19.95	11.05
			Error	21.85	10.65
	Executed	No signed, Eight-dig- it, Deci- mal		18.50	9.70
Executed	Signed, Eight-dig- it, Deci- mal	+	19.00	9.95	
		-	19.60	9.50	

Appendix A Instruction Processing Times

Instruction			Conditions		GL120 Processing Time (μs)	GL130 Processing Time (μs)		
Data Conversion Instructions	32-BIT DATA CONVERSION (DCST)		Not executed		1.45	1.20		
			Executed	No sign, to 32-bit	65535 : 65535	19.50	10.45	
				Sign, to 32-bit	-32767 : 65535	21.40	10.65	
			Executed	No signed, Eight-digit, Decimal	99999999	26.00	12.05	
					4294967295	193.55	11.50	
				Signed, Eight-digit, Decimal	-99999999	28.10	11.80	
2147483647	194.05	11.75						
Other Data Manipulation Instructions	Data Setting Instructions	SET WORD DATA (SDAT)	Not executed		1.05	1.10		
			Executed	Register	2.15	0.95		
		Constant		1.45	0.75			
		SET DOUBLE WORD DATA (SDDT)	Not executed		2.45	0.95		
	Executed			1.05	1.10			
	Data Rearrangement Instructions	LOGICAL BYTE REARRANGEMENT (TWST)	Not executed		1.05	1.15		
			Executed	Table size = 1	14.85	12.35		
				Table size = 50	191.25	125.05		
				Table size = 100	371.25	240.05		
		SWAP (SWAP)	Not executed		1.05	1.10		
			Executed	Table size = 1	12.85	5.20		
				Table size = 50	113.30	78.70		
				Table size = 100 (Max.)	215.80	153.70		
		SORT (SORT)	Not executed		1.25	1.20		
			Sort destination in descending order	Executed	Size = 1	29.85	23.60	
					Source in descending order	Executed	Size = 50	700.35
				Source in ascending order		Executed	Size = 100	1552.50
					Source in random order	Executed	Size = 50	777.55
				Executed		Size = 100	1700.20	1285.20
				Sort destination in ascending order	Executed	Size = 1	29.85	22.45
Source in descending order			Executed			Size = 50	787.05	558.65
	Source in ascending order		Executed		Size = 100	1719.60	1229.70	
Source in random order			Executed		Size = 50	707.45	519.55	
	Executed		Size = 100		1566.90	1156.75		
Executed	Size = 50		1195.75		828.55			
Executed	Size = 100	2860.55	1993.25					

Instruction			Conditions			GL120 Processing Time (μs)	GL130 Processing Time (μs)	
Other Data Manipulat ion Instructio ns	Data Rear- range- ment Instruc- tions	SORT (SORT)	Indexed sort in de- scending order	Executed		Size = 1	32.55	25.00
				Source in descending order	Executed	Size = 50	684.35	513.85
						Size = 100	1517.60	1138.75
				Source in ascending order	Executed	Size = 50	841.65	593.90
						Size = 100	1825.30	1293.90
				Source in random order	Executed	Size = 50	1364.85	928.60
			Size = 100			3081.10	2094.05	
			Indexed sort in as- cending order	Executed		Size = 1	32.35	24.90
				Source in descending order	Executed	Size = 50	841.45	599.25
						Size = 100	1825.10	1307.00
				Source in ascending order	Executed	Size = 50	684.15	519.10
						Size = 100	1517.40	1151.70
				Source in random order	Executed	Size = 50	1371.50	935.90
			Size = 100			3312.75	2267.45	
Data Split/ Combine Instruc- tions	BYTE SPLIT (BYSL)	Not executed			1.05	1.10		
		Executed	Table size = 1		13.85	5.90		
			Table size = 50		160.85	106.35		
			Table size = 100 (Max.)		310.85	208.85		
	BYTE COMPOSI- TION (BYCM)	Not executed			1.05	1.10		
		Executed	Table size = 1		13.45	5.55		
			Table size = 50		140.85	88.85		
			Table size = 100 (Max.)		270.85	173.85		
	NIBBLE SPLIT (NBSL)	Not executed			1.05	1.10		
		Executed	Table size = 1		16.75	7.15		
			Table size = 50		283.80	161.50		
			Table size = 100 (Max.)		556.30	319.00		
	NIBBLE COMPOSI- TION (NBCM)	Not executed			3.00	1.10		
		Executed	Table size = 1 (Register)		18.80	7.40		
Table size = 50 (Register)			293.20	166.65				
Table size = 100 (Max.) (Register)			573.20	329.15				
Other Instruc- tions	BLOCK ADD (BADD)	Not executed			1.25	1.20		
		Executed	Byte addi- tion	Table size = 1	20.25	13.00		
				Table size = 50	140.30	86.50		
				Table size = 100	262.80	161.50		
		Executed	Word addition	Table size = 1	19.60	13.35		
				Table size = 50	100.45	72.15		
				Table size = 100	182.95	132.15		

Appendix A Instruction Processing Times

Instruction			Conditions		GL120 Processing Time (μs)	GL130 Processing Time (μs)			
Other Data Manipulation Instructions	Other Instructions	CHECKSUM (CKSM)	Not executed		1.45	1.30			
			Executed	Word sum	Table size = 1	17.70	13.30		
					Table size = 100	171.15	142.00		
					Table size = 255	411.40	343.50		
				Byte sum	Table size = 1	18.10	13.35		
					Table size = 100	181.45	137.10		
					Table size = 255	437.20	330.85		
				CRC	Table size = 1	24.20	20.40		
					Table size = 100	823.70	681.90		
					Table size = 255	2072.80	1693.80		
			LRC	Table size = 1	17.75	12.85			
				Table size = 100	181.10	136.60			
Table size = 255	436.85	330.35							
System Status Monitoring Instruction	SYSTEM STATUS MONITORING (STAT)	Not executed		1.25	1.20				
		Executed	Words 1 to 8 Reading system status		70.60	52.45			
			Words 9 to 132 Reading I/O Module health table		866.50	571.00			
			Words 133 to 162 Reading remote status		227.30	152.70			
			Words 163 to 255 Reading I/O error counters		655.70	433.05			
			Words 256 to 274 Reading Option Module status		151.40	103.10			
			Words 275 to 296 Reading Option Module revision		172.90	117.10			
			Words 297 to 336 Reading Local I/O status history		295.30	197.20			
			Words 337 to 376 Reading Stop status history		295.30	197.20			
			Words 377 to 386 Reading No. 1 MC20 Module error status history		91.30	63.70			
			Words 387 to 396 Reading No. 2 MC20 Module error status history		91.30	63.70			
			Words 397 to 464 Reading PC Link bit status		674.40	415.30			
			Sequence Control Instructions	SEQUENCE CONTROL INTER-FACE (SCIF)	Not executed	Drum sequence mode		22.65	28.60
						ICMP mode		20.55	27.35
Executed	Drum sequence mode, Table size = 255	increment			34.70	35.70			
		Pointer 0 clear			34.25	34.90			
		Last step			35.10	36.40			
	ICMP mode, Table size = 255	Comparison error			29.35	32.90			
		Comparison okay			30.70	33.55			



Instruction			Conditions	GL120 Processing Time (μs)	GL130 Processing Time (μs)	
Program Control Instructions	Skip Node Instructions	SKIP CONSTANT (SKPC) SKIP REGISTER (SKPR)	Not executed		0.85	0.90
			Executed	Skip network =1	12.65	17.95
				Skip network =50	175.50	110.85
				Skip network =100	341.65	205.85
	Subroutine Instructions	SUBROUTINE JUMP (JSR)	Not executed		1.05	1.10
			Executed		20.95	18.70
		SUBROUTINE RETURN (RET)	Not executed		0.85	0.90
			Executed		10.25	9.70
	Master Control Instructions	MASTER CONTROL ON (MSON)	Not executed		0.85	0.70
			Executed		1.35	0.90
MASTER CONTROL OFF (MSOF)		Not executed		-	-	
		Executed		0.95	0.25	
I/O Control Instructions	DIRECT IN* (DIN)		Not executed		1.25	1.20
			Executed	Input 1 word	79.35	50.75
				Input 2 words	82.40	52.75
	DIRECT OUT* (DOUT)		Not executed		1.25	1.20
			Executed	Output 1 word	78.70	48.70
				Input 2 word	81.75	50.35
Expansion Math Instructions (EMTH)	Integer Calculations	BASE 10 LOG-ARITHM (EMTH LOG)	Not executed		1.05	1.10
			Executed	LOG (99999999)	137.95	108.15
				LOG (9999)	159.10	124.70
				Error	26.10	19.65
		BASE 10 ANTILOG-ARITHM (EMTH ANLOG)	Not executed		1.05	1.10
			Executed	ANLOG (7999)	36.55	26.40
	ANLOG (7997)			222.75	182.15	
	Error			26.40	19.75	
	Floating Point Decimal Calculations	INTEGER TO FLOATING POINT CONVERSION (EMTH CNVIF)	Not executed		1.05	1.10
			Executed	99999999	87.75	68.20
				0	73.35	55.00
		INTEGER + FLOATING POINT ADDITION (EMTH ADDIF)	Not executed		1.05	1.10
			Executed	0 + 0.0	88.25	70.65
				99999999 + (9.999999E+7)	121.75	97.00
INTEGER - FLOATING POINT SUBTRACTION (EMTH SUBIF)		Not executed		1.05	1.10	
		Executed	0 - 0.0	90.80	74.10	
	99999999 - 5.0		144.35	116.45		
INTEGER × FLOATING POINT MULTIPLICATION (EMTH MULIF)	Not executed		1.05	1.10		
	Executed	0 × 0.0	93.45	74.15		
		99999999 × (9.999999E+7)	126.80	98.65		

\*Not including Module response time.

Appendix A Instruction Processing Times

Instruction			Conditions	GL120 Processing Time (μs)	GL130 Processing Time (μs)	
Expansion Math Instructions (EMTH)	Floating Point Decimal Calculations	INTEGER DIVIDED BY FLOATING POINT (EMTH DIVIF)	Not executed	1.05	1.10	
			Executed	99999999 ÷ (9.999999E+7)	142.40	108.25
			100000000 ÷ (3E - 24)	144.40	108.55	
		FLOATING POINT - INTEGER SUBTRACTION (EMTH SUBFI)	Not executed	1.05	1.10	
			Executed	0.0 - 0	90.95	73.55
			(9.999999E+7) - 99999999	144.95	115.75	
		FLOATING POINT DIVIDED BY INTEGER (EMTH DIVFI)	Not executed	1.05	1.10	
			Executed	0.0 ÷ 0	95.55	74.25
			(9.999999E +7) ÷ 99999999	142.45	107.45	
		INTEGER TO FLOATING POINT COMPARISON (EMTH CMPIF)	Not executed	1.05	1.10	
			Executed	99999999 and 0.0	107.15	88.35
			99999999 and 9.999999E+7	131.35	104.85	
		FLOATING POINT TO INTEGER CONVERSION (EMTH CNVFI)	Not executed	1.05	1.10	
			Executed	9.999999E + 7	83.80	64.15
			Executed	Error	90.15	70.65
		FLOATING POINT ADDITION (EMTH ADDFP)	Not executed	1.05	1.10	
			Executed	Adding 0.0s	85.55	62.25
			Adding other than 0.0s	98.25	73.60	
		FLOATING POINT SUBTRACTION (EMTH SUBFP)	Not executed	1.05	1.10	
			Executed	Subtracting 0.0s	89.10	65.65
			Subtracting other than 0.0s	101.35	77.15	
		FLOATING POINT MULTIPLICATION (EMTH MULFP)	Not executed	1.05	1.10	
			Executed	Multiplying 0.0s	64.85	48.30
			Multiplying other than 0.0s	83.70	58.65	
FLOATING POINT DIVISION (EMTH DIVFP)	Not executed	1.05	1.10			
	Executed	Dividing 0.0	66.75	49.45		
	Dividing other than 0.0	99.45	69.05			
FLOATING POINT COMPARISON (EMTH CMPFP)	Not executed	1.05	1.10			
	Executed	0.0 and 0.0	60.20	46.75		
	3.402823E+38 and 3.402823E+38	100.50	76.85			
FLOATING POINT SQUARE ROOT (EMTH SQRF)	Not executed	1.05	1.10			
	Executed	$\sqrt{-1.0}$	55.50	41.05		
	$\sqrt{3.402823E + 38}$	180.85	137.35			

Instruction		Conditions		GL120 Processing Time ( $\mu$ s)	GL130 Processing Time ( $\mu$ s)	
Expansion Math Instructions (EMTH)	Floating Point Decimal Calcula- tions	CHANGING THE SIGN OF A FLOAT- ING POINT NUM- BER (EMTH CHSIN)	Not executed		1.05	1.10
			Executed	0.0	46.25	29.95
				Other than 0.0	47.35	29.95
		LOAD FLOATING POINT $\pi$ (EMTH PI)	Not executed		1.05	1.10
			Executed		44.75	28.90
		FLOATING POINT SINE OF AN ANGLE (IN RA- DIANS) (EMTH SINE)	Not executed		1.05	1.10
			Executed	SIN ( $0 \times \pi/180$ )	408.40	351.05
				SIN ( $90 \times \pi/180$ )	481.55	405.95
				SIN ( $135 \times \pi/180$ )	726.20	581.40
				SIN ( $180 \times \pi/180$ )	508.90	430.25
				SIN ( $360 \times \pi/180$ )	508.70	431.60
				Error (65536)	56.70	41.45
		FLOATING POINT COSINE OF AN ANGLE (IN RA- DIANS) (EMTH COS)	Not executed		1.05	1.10
			Executed	COS ( $0 \times \pi/180$ )	378.50	324.10
				COS ( $90 \times \pi/180$ )	511.85	433.30
				COS ( $135 \times \pi/180$ )	679.75	545.20
				COS ( $180 \times \pi/180$ )	479.00	403.30
				COS ( $360 \times \pi/180$ )	478.80	404.65
				Error (65536)	56.70	41.50
		FLOATING POINT TANGENT OF AN ANGLE (IN RA- DIANS) (EMTH TAN)	Not executed		1.05	1.10
			Executed	TAN ( $0 \times \pi/180$ )	599.00	519.40
				TAN ( $45 \times \pi/180$ )	1051.95	837.30
				TAN ( $135 \times \pi/180$ )	1050.10	836.15
				TAN ( $180 \times \pi/180$ )	699.50	598.60
				TAN ( $360 \times \pi/180$ )	699.30	599.95
				Error (65536)	56.70	41.50
		FLOATING POINT ARC SINE OF AN ANGLE (IN RA- DIAN) (EMTH AR- SIN)	Not executed		1.05	1.10
			Executed	SIN <sup>-1</sup> (0. 0)	531.65	444.35
				SIN <sup>-1</sup> (1. 0)	45.85	29.85
				SIN <sup>-1</sup> (1.0E - 32)	622.30	499.50
				SIN <sup>-1</sup> (0.999999)	907.05	706.55
		FLOATING POINT ARC COSINE OF AN ANGLE (IN RA- DIAN) (EMTH AR- COS)	Not executed		1.05	1.10
			Executed	COS <sup>-1</sup> (0. 0)	44.00	28.90
COS <sup>-1</sup> (1. 0)	450.25			377.35		
COS <sup>-1</sup> (1.0E - 32)	716.95			570.00		
COS <sup>-1</sup> (0.99999999)	792.90			620.35		
COS <sup>-1</sup> (0.899)	1062.10			836.50		
Not executed		1.05	1.10			
FLOATING POINT ARC TANGENT OF AN ANGLE (IN RA- DIAN) (EMTH AR- TAN)	Not executed		1.05	1.10		
	Executed	TAN <sup>-1</sup> (0. 0)	314.95	269.10		
		TAN <sup>-1</sup> (1. 0)	516.95	433.40		
		TAN <sup>-1</sup> (1.0E - 32)	355.40	293.95		
		TAN <sup>-1</sup> (0.999999)	688.50	544.20		
		TAN <sup>-1</sup> (0.899)	743.80	587.35		

Appendix A Instruction Processing Times

Instruction		Conditions	GL120 Processing Time (μs)	GL130 Processing Time (μs)		
Expansion Math Instructions (EMTH)	Floating Point Decimal Calculations	RADIAN TO DEGREE CONVERSION (EMTH CNVRD)	Not executed	1.05	1.10	
			Executed	0.0	68.00	49.45
				Other than 0.0	86.40	60.45
		DEGREE TO RADIAN CONVERSION (EMTH CNVDR)	Not executed	1.05	1.10	
			Executed	0.0	67.75	49.45
				Other than 0.0	86.10	60.35
		RAISING A FLOATING POINT NUMBER TO AN INTEGER POWER (EMTH POW)	Not executed	1.05	1.10	
			Executed	0.0 <sup>1</sup>	57.70	42.85
				15.999999 <sup>32</sup>	314.30	237.95
				0.999999 <sup>32767</sup>	1304.30	971.50
		FLOATING POINT EXPONENTIAL FUNCTION (EMTH EXP)	Not executed	1.05	1.10	
			Executed	EXP (0. 0)	662.20	577.40
				EXP (1. 0)	1078.05	865.85
				EXP (88.72283) (Max. value)	1057.95	846.65
		FLOATING POINT NATURAL LOGARITHM (EMTH LNFP)	Not executed	1.05	1.10	
			Executed	LOG (1. 0)	505.15	434.80
				LOG (9.999998 (E+37))	806.50	645.15
				Error	56.70	42.15
		FLOATING POINT COMMON LOGARITHM (EMTH LOGFP)	Not executed	1.05	1.10	
			Executed	LOG <sub>10</sub> (1. 0)	538.85	464.95
LOG <sub>10</sub> (9.999998E+37)	859.25			686.60		
Error	57.45			42.90		
FLOATING POINT ERROR REPORT LOG (EMTH ER-LOG)	Not executed	1.05	0.90			
	Executed	Error	39.50	26.70		
		No error	39.55	27.15		

# MEMOCON GL120, GL130 SOFTWARE USER'S MANUAL Vol.1

**IRUMA BUSINESS CENTER**

480, Kamifujisawa, Iruma, Saitama 358-8555, Japan  
Phone 81-42-962-5696 Fax 81-42-962-6138

**YASKAWA ELECTRIC AMERICA, INC.**

2121 Norman Drive South, Waukegan, IL 60085, U.S.A.  
Phone 1-847-887-7000 Fax 1-847-887-7370

**MOTOMAN INC. HEADQUARTERS**

805 Liberty Lane West Carrollton, OH 45449, U.S.A.  
Phone 1-937-847-6200 Fax 1-937-847-6277

**YASKAWA ELÉTRICO DO BRASIL COMÉRCIO LTD.A.**

Avenida Fagundes Filho, 620 Bairro Saude-Sao Paulo-SP, Brazil CEP: 04304-000  
Phone 55-11-5071-2552 Fax 55-11-5581-8795

**YASKAWA ELECTRIC EUROPE GmbH**

Am Kronberger Hang 2, 65824 Schwalbach, Germany  
Phone 49-6196-569-300 Fax 49-6196-569-398

**Motoman Robotics Europe AB**

Box 504 S38525 Torsås, Sweden  
Phone 46-486-48800 Fax 46-486-41410

**Motoman Robotec GmbH**

Kammerfeldstraße 1, 85391 Allershausen, Germany  
Phone 49-8166-90-100 Fax 49-8166-90-103

**YASKAWA ELECTRIC UK LTD.**

1 Hunt Hill Orchardton Woods Cumbernauld, G68 9LF, United Kingdom  
Phone 44-1236-735000 Fax 44-1236-458182

**YASKAWA ELECTRIC KOREA CORPORATION**

Kfpa Bldg #1201, 35-4 Youido-dong, Yeongdungpo-Ku, Seoul 150-010, Korea  
Phone 82-2-784-7844 Fax 82-2-784-8495

**YASKAWA ELECTRIC (SINGAPORE) PTE. LTD.**

151 Lorong Chuan, #04-01, New Tech Park Singapore 556741, Singapore  
Phone 65-6282-3003 Fax 65-6289-3003

**YASKAWA ELECTRIC (SHANGHAI) CO., LTD.**

4F No.18 Aona Road, Wagaogiao Free Trade Zone, Pudong New Area, Shanghai 200131, China  
Phone 86-21-5866-3470 Fax 86-21-5866-3869

**YATEC ENGINEERING CORPORATION**

4F, No.49 Wu Kong 6 Rd, Wu-Ku Industrial Park, Taipei, Taiwan  
Phone 886-2-2298-3676 Fax 886-2-2298-3677

**YASKAWA ELECTRIC (HK) COMPANY LIMITED**

Rm. 2909-10, Hong Kong Plaza, 186-191 Connaught Road West, Hong Kong  
Phone 852-2803-2385 Fax 852-2547-5773

**BEIJING OFFICE**

Room No. 301 Office Building of Beijing International Club, 21  
Jianguomenwai Avenue, Beijing 100020, China  
Phone 86-10-6532-1850 Fax 86-10-6532-1851

**TAIPEI OFFICE**

9F, 16, Nanking E. Rd., Sec. 3, Taipei, Taiwan  
Phone 886-2-2502-5003 Fax 886-2-2505-1280

**SHANGHAI YASKAWA-TONGJI M & E CO., LTD.**

27 Hui He Road Shanghai China 200437  
Phone 86-21-6553-6060 Fax 86-21-5588-1190

**BEIJING YASKAWA BEIKE AUTOMATION ENGINEERING CO., LTD.**

30 Xue Yuan Road, Haidian, Beijing P.R. China Post Code: 100083  
Phone 86-10-6233-2782 Fax 86-10-6232-1536

**SHOUGANG MOTOMAN ROBOT CO., LTD.**

7, Yongchang-North Street, Beijing Economic Technological Investment & Development Area,  
Beijing 100076, P.R. China  
Phone 86-10-6788-0551 Fax 86-10-6788-2878



YASKAWA

YASKAWA ELECTRIC CORPORATION

In the event that the end user of this product is to be the military and said product is to be employed in any weapons systems or the manufacture thereof, the export will fall under the relevant regulations as stipulated in the Foreign Exchange and Foreign Trade Regulations. Therefore, be sure to follow all procedures and submit all relevant documentation according to any and all rules, regulations and laws that may apply. Specifications are subject to change without notice for ongoing product modifications and improvements.

MANUAL NO. SIEZ-C825-20.11B

© Printed in Japan November 2002 96-5  
02-5①  
95-81192, 95-C82-031