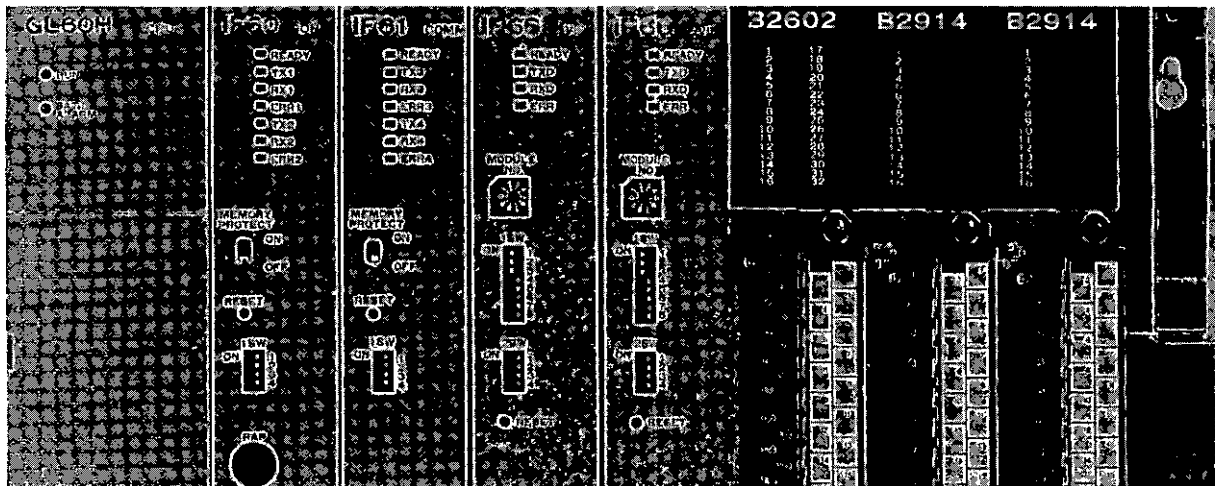


MEMOCON-SC GL60H/GL70H DESCRIPTIVE INFORMATION

PROGRAMMABLE CONTROLLER





YASKAWA

MANUAL NO. SIE-C815-17.1B

NOTES FOR SAFE OPERATION


Read these manuals thoroughly before use of MEMOCON-SC GL60H/GL70H. In these manuals, NOTES FOR SAFE OPERATION are classified as "WARNING" and "CAUTION."


 **WARNING** : Indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury to personnel.

 **CAUTION** : Indicates a potentially hazardous situation which, if not avoided, may result in minor or moderate injury to personnel and damage to equipment. It may also be used to alert against unsafe practices.

Even items described in  **CAUTION** may result in a vital accident in some situations. In either case, follow these important notes.

The following shows the symbols of prohibition and mandatory action.

 **PROHIBITION** : Specifies prohibited handling.

 **MANDATORY ACTION** : Specifies actions that must be taken.

After reading these manuals, keep them readily available for those using the equipment.

1 INSTALLATION

CAUTION

- The installation environment must meet the environmental conditions given in the product catalog and manuals.

Using the GL60H/GL70H in environments subject to high temperatures, high humidity, excessive dust, corrosive gases, vibration, or shock can lead to electric shock, fire, or faulty operation.

—Do not use the GL60H/GL70H in the following locations.

- Locations subject to direct sunlight or ambient temperatures not between 0 and 55°C.
- Locations subject to relative humidity in excess of 95%, rapid changes in humidity, or condensation.
- Locations subject to corrosive or flammable gas.
- Locations that would subject the GL60H/GL70H to direct vibration or shock.
- Locations subject to contact with water, oil, chemicals, etc.

- Install products correctly according to the instructions.

Improper installation may result in accidents or malfunctions.

- ① Be sure all screws are tight.

All screws for installation and terminal board should be securely tightened and checked for loosening. Malfunctions in the GL60H/GL70H may occur as a result of loose screws.

- ② Install the mounting base correctly.

Install the mounting base facing in the correct direction. Incorrect installation may result in accidents or malfunctions.

- When installing the mounting base, leave the cover on to prevent contamination from foreign matter.

Foreign matter can cause malfunction in the GL60H/GL70H.

- Do not remove the cover of the connector where a module is not mounted. Foreign matter can cause malfunction in the GL60H/GL70H.

2 WIRING

CAUTION

- Connect a power supply complying with the rated specifications.
A power supply that does not comply with the rating may cause a fire.
- Wiring must be performed by qualified personnel.
Mistakes in wiring can cause fires, product failure, or malfunctions.
- When wiring, do not allow foreign matters such as wire shrapnel to enter the mounting base or the module.
Foreign matter can cause fires, product failure, or malfunctions.
- When using output modules without built-in fuses, connect the fuse in series with the load to conform to load specifications.
Unconnected fuses may cause fire, breakdown, and damage output circuits.

MANDATORY ACTION

- Ground the protective ground terminal to a resistance of 100Ω max.
Failure to observe this instruction may result in electric shock or malfunction.

INSERT THE INTERFACE CABLES PROPERLY

- Insert the connectors of the various interface cables that are connected to GL60H/GL70H into the communication parts and secure them properly.
Failure to observe this instruction may result in malfunctions.

NOISE REDUCTION MEASURES

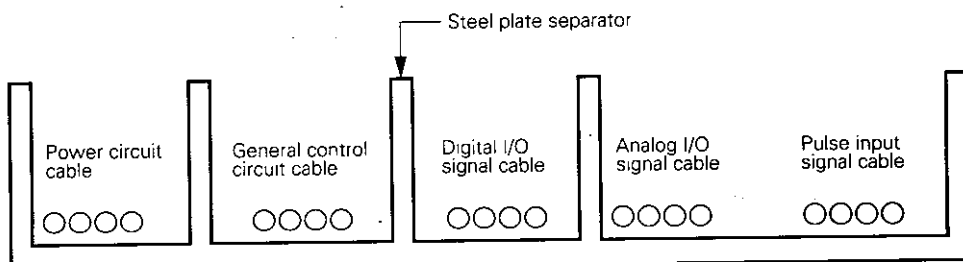
- When noise from external power supply lines causes problems, install an insulated transformer and noise filter for effective noise prevention.
Insufficient noise reduction measure may cause malfunctions in the GL60H/GL70H.



SEPARATE WIRING PROPERLY

- I/O lines connecting external devices to the GL60H/GL70H must be selected based on the following considerations:
 - mechanical strength, resistance to noise, wiring distance, signal voltage, etc.
- I/O lines must be separated from power lines both within and outside of the control panel to minimize the affects of noise.
Faulty operation can result if I/O lines are not sufficiently separated from power lines.

(Example of external wiring)



3 PRECAUTION UPON USE

⚠ WARNING

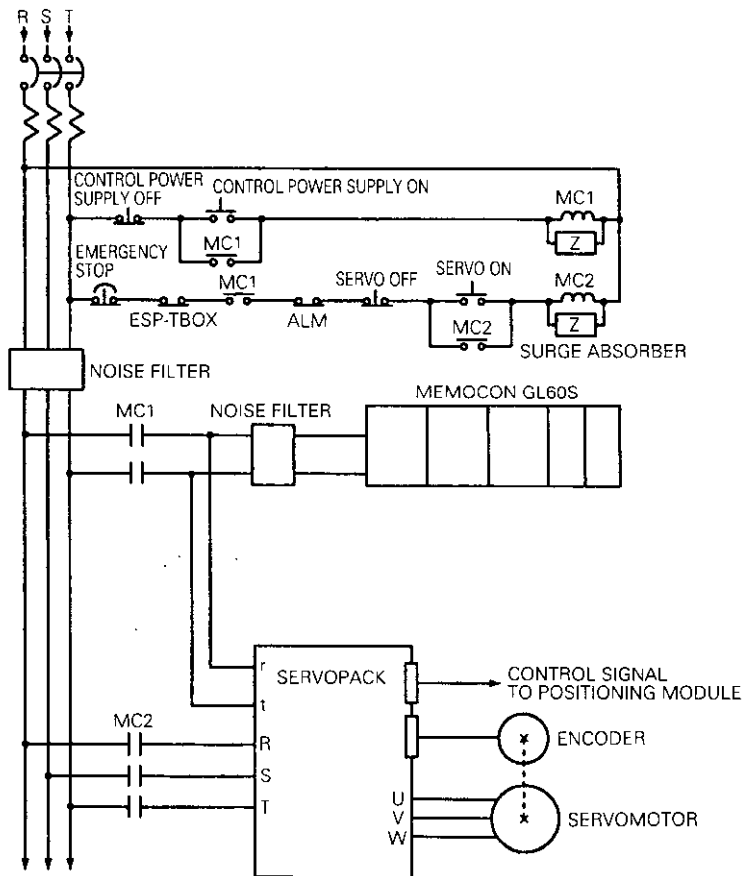
- Do not touch module terminals under current conditions. There is danger of electric shock.
- Provide an emergency stop circuit, interlock circuit, etc. at the exterior of GL60H/GL70H. Failure to observe this instruction may result in injury or damage to equipment.

— Provide an emergency stop circuit at the exterior of GL60H/GL70H. —

An emergency stop circuit for the control system should not be constructed using the ladder programming in the GL60H/GL70H. Install an emergency stop circuit to an external relay as shown below.

Use a NC contact (mechanical contact) to connect the emergency stop switch. The emergency stop switch should cut off the main power supply when depressed.

If these steps are not followed, the emergency switch will not engage even if input circuits are damaged or cables are cut. Failure to follow instructions may cause damage to machines and injury to personnel.



(Continued)

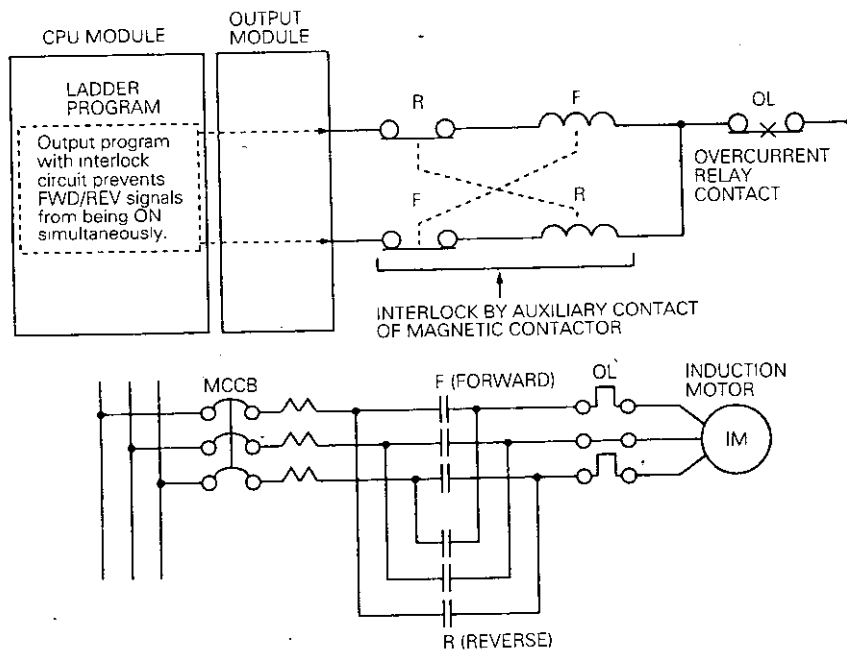
Provide an interlock circuit at the exterior of the GL60H/GL70H.

Provide an interlock circuit at the exterior of the GL60H/GL70H to prevent injury or damage to equipment.

(Example) Interlock phase for forward and reverse drives

To prevent forward and reverse operation signals from being turned on simultaneously, install an interlock circuit in the GL60H/GL70H ladder program.

At the same time, use auxiliary contacts of external magnetic contactor to install a second interlock circuit to prevent forward and reverse drive magnetic contactors from being turned on simultaneously.



⚠ CAUTION

- When using output modules without built-in fuses, connect the fuse in series with the load to conform to load specifications. Unconnected fuses may cause fires, breakdowns, or damage output circuits.

4 MAINTENANCE

WARNING

- Do not reverse polarity, charge, disassemble, or expose battery to heat or flame.
There is danger of bursting or fire.

PROHIBITION

- Do not attempt to disassemble or modify the MEMOCON-SC in any way.
Doing so can cause fires, product failure, or malfunctions.

MONITOR THE LIFE OF BATTERY

- Monitor the life of CPU module built-in battery.
If the "BATTERY ALARM" indicator lights, replace the battery with a new one within a month. CPU module memory (ladder program, etc.) may be erased if battery change is delayed.

OVERHAUL POWER MODULE REGULARLY

- Perform power module overhaul every 5 years.
Malfunction in the power supply unit may occur due to deterioration in product lifetime of smoothing capacitors, etc. Shortening overhaul frequency should be considered under the following conditions:
 - When used in areas of high humidity or heavy climate change.
 - When there are large fluctuations in electrical voltage, load, frequency and waveform.
 - When equipment was stored in a harsh environment or left unused for long periods.

5 GENERAL PRECAUTION

- GL60H/GL70H was not designed or manufactured for use in devices or systems that concern peoples' lives.
Users who intend to use the product described in this manual for special purposes such as devices or systems relating to transportaion, medical, space aviation, atomic power control, or underwater use must contact YASKAWA representatives 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 GL60H/GL70H 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 example 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 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 representaive listed on the back of this manual to order new nameplates whenever a nameplate becomes worn or damaged.
- YASKAWA cannot make any guarantee for products which have been modified.
YASKAWA assumes no responsibility for any injury or damage caused by a modified product.

OVERVIEW OF MANUAL

- This manual describes the following items of GL60H/GL70H.
 - ① System configuration
 - ② System function and specification
 - ③ Operation function
 - ④ SFC function
 - ⑤ Installation and wiring
 - ⑥ Internal board installation and mounting hole dimensions
 - ⑦ Dimensions
- Read this manual carefully in order to use the GL60H/GL70H properly. Also, keep this manual in a safe place so that it can be used whenever necessary.
- Refer to the following manuals as necessary.

	Document Title	Document Number	Content
Man/ Machine Interface	MEMOCON-SC GL60S P150 PROGRAMMING PANEL DESCRIPTIVE INFORMATION	SIE-C815-14. 3	Describes the SFC function, specifications, application methods, etc., for the P150 Programming Panel.
I/O Modules	MEMOCON-SC GL40S, GL60S, GL70H 2000 Series I/O Modules DESCRIPTIVE INFORMATION	SIE-C815-13. 3	Describes functions, specifications, application methods, etc., for the 2000 Series Digital I/O Modules.
	MEMOCON-SC GL40S, GL60S, GL70H 2000 Series Analog I/O Modules DESCRIPTIVE INFORMATION	SIE-C815-13. 9	Describes functions, specifications, application methods, etc., for the 2000 Series Analog I/O Modules.
Intelligent Modules	MEMOCON-SC GL40S, GL60S, GL70H 2000 Series Reversible Counter Module DESCRIPTIVE INFORMATION	SIE-C815-13. 11	Describes functions, specifications, application methods, etc., for the 2000 Series Reversible Counter Module (B2801).
	MEMOCON-SC GL40S, GL60S, GL70H 2000 Series PRESET COUNTER MODULE DESCRIPTIVE INFORMATION	SIE-C815-13. 12	Describes functions, specifications, application methods, etc., for the 2000 Series Preset Counter Module (B2802).
	MEMOCON-SC GL40S, GL60S, GL70H 2000 SERIES POSITIONING MODULE B2803 DESCRIPTIVE INFORMATION	SIE-C815-13. 13	Describes functions, specifications, application methods, etc., for the 2000 Series Positioning Module B2803.
	MEMOCON-SC GL40S, GL60S, GL70H 2000 SERIES POSITIONING MODULE B2813 DESCRIPTIVE INFORMATION	SIE-C815-13. 14	Describes functions, specifications, application methods, etc., for the 2000 Series Positioning Module B2813.
	MEMOCON-SC GL40S, GL60S, GL70H 2000 SERIES POSITIONING MODULE B2823 DESCRIPTIVE INFORMATION	SIE-C815-13. 16	Describes functions, specifications, application methods, etc., for the 2000 Series Positioning Module B2823.
	MEMOCON-SC GL40S, GL60S, GL70H 2000 SERIES I/O POSITIONING MODULE B2833 DESCRIPTIVE INFORMATION	SIE-C815-13. 17	Describes functions, specifications, application methods, etc., for the 2000 Series Positioning Module B2833.
	MEMOCON-SC GL40S, GL60S, GL70H 2000 SERIES I/O PID MODULE DESCRIPTIVE INFORMATION	SIE-C815-14. 24	Describes functions, specifications, application methods, etc., for the 2000 Series I/O PID Module (B2800).
Motion Control	MEMOCON-SC GL40S MOTION CONTROL VERSION DESCRIPTIVE INFORMATION	SIE-C815-15. 30	Describes functions, specifications, application methods, etc., for the MEMOCON-SC GL40S MOTION CONTROL VERSION.

Communi- cation Modules	MEMOCON-SC GL40S, GL60S, GL70H MEMOBUS DESCRIPTIVE INFORMATION	SIE-C815-13. 60	Describes functions, specifications, application methods, etc., for MEMOBUS.
	MEMOCON-SC GL60S, GL60H, GL70H 2000 SERIES ASCII MODULE DESCRIPTIVE INFORMATION	SIE-C815-14. 4	Describes functions, specifications, application methods, etc., for the ASCII Modules.
	MEMOCON-SC GL60S COMM COMMAND DESCRIPTIVE INFORMATION	SIE-C815-14. 5	Describes functions, specifications, application methods, etc., for the COMM COMMAND.
	MEMOCON-SC GL60S REMOTE I/O SYSTEM DESCRIPTIVE INFORMATION	SIE-C815-14. 7	Describes functions, specifications, application methods, etc., for me REMOTE I/O SYSTEM.
	MEMOCON-SC GL40S, GL60S, GL70H 2000 Series PC Link Module DESCRIPTIVE INFORMATION	SIE-C815-14. 8	Describes functions, specifications, application methods, etc., for the 2000 Series PC Link Module (IF64).
	MEMOCON-SC YENET-3200 OPTICAL LAN SYSTEM DESCRIPTIVE INFORMATION	SIE-C815-14. 16	Describes functions, specifications, application methods, etc., for the YENET-3200 Optical Lan System.
	MEMOCON-SC GL20/GL60S INTERFACE MODULE DESCRIPTIVE INFORMATION	SIE-C815-14. 18	Describes functions, specifications, application methods, etc., for the Interface Module.
	MEMOCON-SC GL40S, GL60S, GL70H 2000 Series Uni-Wire Interface Module DESCRIPTIVE INFORMATION	SIE-C815-14. 32	Describes functions, specifications, application methods, etc., for the 2000 Series Uni-wire Interface Module (B2808).

USING THIS MANUAL

- This manual concerns people involved with the following activities.
 - ① Preparing an estimate for the GL60H/GL70H
 - ② Evaluating the GL60H/GL70H for use
 - ③ Design and setup of GL60H/GL70H installed control panels and operation panels
 - ④ Manufacture of GL60H/GL70H installed control panels and operation panels
 - ⑤ Inspection of GL60H/GL70H installed control panels and operation panels
 - ⑥ Test run adjustment of GL60H/GL70H installed control panels and operation panels
 - ⑦ Maintenance of GL60H/GL70H installed control panels and operation panels
- Meaning of Basic Terms
 In this manual, the following terms indicate the meanings as described below, unless otherwise specified.
 - PC=Programmable Controller
 - PP=Programming Panel
 - GL40S, GL60S = MEMOCON-SC GL40S, GL60S, GL60H GL70H
GL60H, GL70H Programmable Controllers

CONTENTS

	Page
1 INTRODUCTION	1
2 GL60H, GL70H CONFIGURATION	2
3 GL60H, GL70H SPECIFICATIONS	4
4 IMPORTANT MACHINE CONCEPTS	24
5 PROGRAMMING FUNCTIONS	47
6 SFC FUNCTIONS	298
7 I/O ALLOCATION	373
8 GL60H, GL70H APPLICATIONS	379
9 GL60H, GL70H HANDLING AND MAINTENANCE	398
APPENDIX	437

CONTENTS

	Page
SECTION 1 INTRODUCTION	1
1.1 GENERAL	1
SECTION 2 GL60H, GL70H CONFIGURATION	2
SECTION 3 GL60H, GL70H SPECIFICATIONS	4
3.1 BASIC GL60H, GL70H SPECIFICATIONS	4
3.2 CPU MODULE	4
3.3 POWER SUPPLY MODULE	6
3.4 COMMUNICATION MODULE	8
3.5 I/O BUFFER MODULE	18
3.6 I/O MODULE	18
3.7 MOUNTING BASE	20
3.8 I/O CABLE	21
3.9 P150 PROGRAMMING PANEL	22
SECTION 4 IMPORTANT MACHINE CONCEPTS	24
4.1 USER PROGRAM CONFIGURATION	24
4.2 NETWORKS	25
4.3 CONTROLLER REFERENCE NUMBERS	25
4.4 GL60H, GL70H INTERNAL PROCESS	31
4.5 SCANNING	34
4.5.1 Solving a Ladder Circuit	34
4.5.2 Scan Time	36
4.5.3 Two-Level Scan	41
4.6 ALLOWABLE NUMBER OF MEMORY WORDS	44
4.7 DISABLE FUNCTION	45
4.8 TRACE BACK FUNCTION	45
SECTION 5 PROGRAMMING FUNCTIONS	47
5.1 PROGRAMMING FUNCTION LIST	47
5.2 RELAYS	51
5.2.1 Relay Logic Elements	51
5.2.2 Example Relay Logic Circuit	59
5.2.3 Creating of Relay Circuits	60
5.2.4 Sample Application Circuits of Relays	65
5.3 TIMERS	68
5.3.1 Types of Timers	68
5.3.2 Timer Configuration	68
5.3.3 Function and Operation of Timer	69
5.3.4 Programming Timer Circuit and Precautions	71
5.3.5 Application Timer Circuits	73
5.4 COUNTERS	75
5.4.1 Types of Counters	75
5.4.2 Counter Configuration	75
5.4.3 Function and Operation of Counter	76
5.4.4 Programming Counter Circuit and Precautions	79
5.4.5 Application Counter Circuits	81
5.5 ARITHMETIC FUNCTIONS	84
5.5.1 Types of Arithmetic Functions	84
5.5.2 Addition (ADD)	84
5.5.3 Double-precision Addition (DADD)	86
5.5.4 Subtraction (SUB)	88
5.5.5 Double-precision Subtraction (DSUB)	90
5.5.6 Multiply (MUL)	92
5.5.7 Double-precision Multiply (DMUL)	93
5.5.8 Divide (DIV)	95
5.5.9 Double-precision Divide Function (DDIV)	98
5.5.10 Programming Arithmetic Logic and Precautions	101
5.5.11 Example-Application Circuits of Arithmetic Functions	103

	Page
5.6 SIGNED ARITHMETIC FUNCTIONS	108
5.6.1 Types of Signed Arithmetic Functions	108
5.6.2 Sign Representation	108
5.6.3 Signed Addition (SADD)	109
5.6.4 Signed Double-Precision Addition (SDAD)	112
5.6.5 Signed Subtraction (SSUB)	115
5.6.6 Signed Double-precision Subtraction (SDSB)	117
5.6.7 Signed Multiply (SMUL)	120
5.6.8 Signed Divide (SDIV)	122
5.6.9 Programming Signed Arithmetic Logic and Precautions	125
5.7 SQUARE ROOT	127
5.7.1 Types of Square Root	127
5.7.2 Square Root (SQRT)	127
5.7.3 Double-precision Square Root (DSQR)	129
5.7.4 Programming Square Root Circuit and Precautions	130
5.8 TRIGONOMETRIC FUNCTION	132
5.8.1 Types of Trigonometric Function Operations	132
5.8.2 Sine (SIN)	132
5.8.3 Cosine (COS)	134
5.8.4 Programming Trigonometric Function Circuit and Precautions	135
5.9 DATA MOVE	137
5.9.1 Basic Terminology	137
5.9.2 Types of Data Move	138
5.9.3 Block Move (BLKM)	139
5.9.4 Register-to-Table Move (R→T)	142
5.9.5 Table-to-Register Move (T→R)	145
5.9.6 Table-to-Table Move (T→T)	148
5.9.7 First In (FIN)	151
5.9.8 First Out (FOUT)	153
5.9.9 Table Search (SRCH)	157
5.9.10 Table Set (TSET)	160
5.9.11 Get Controller System Status (STAT)	162
5.9.12 Programming Data Move Circuit and Precautions	163
5.9.13 Example-Application Circuits of Data Move	164
5.10 INDEXED BLOCK MOVE	167
5.10.1 Types of Indexed Block Move	167
5.10.2 Block Move 1 with Destination Index (DIBT)	167
5.10.3 Block Move 2 with Destination Index (DIBR)	172
5.10.4 Block Move 1 with Source Index (SIBT)	175
5.10.5 Block Move 2 with Source Index (SIBR)	179
5.10.6 Programming Indexed Block Move Circuit and Precautions	182
5.11 DATA CONVERSION	183
5.11.1 Types of Data Conversion	183
5.11.2 BCD→Binary Conversion (BIN)	183
5.11.3 Binary→BCD Conversion (BCD)	186
5.11.4 Swap (SWAP)	189
5.11.5 Sort (SORT)	191
5.11.6 Byte Split (BYSL)	194
5.11.7 Byte Composition (BYCM)	196
5.11.8 Block Addition (BADD)	198
5.12 MATRIX	200
5.12.1 Types of Matrix	200
5.12.2 Form of Matrix	201
5.12.3 AND, OR, XOR	202
5.12.4 Complement (COMP)	205
5.12.5 Compare (CMPR)	207

CONTENTS (Cont'd)

	Page
5.12.6 Modify (MBIT)	211
5.12.7 Sense (SENS)	213
5.12.8 Rotate (BROT)	217
5.12.9 Multi-Rotate (MROT)	220
5.12.10 Byte Rearrangement (TWST)	223
5.12.11 Bit Count (BCNT)	225
5.13 SKIP	227
5.14 SUBROUTINE	228
5.15 STEPPING SWITCH	230
5.15.1 Stepping Switch Functions and Operation	230
5.15.2 Example of Stepping Switches	231
5.16 ASCII COMMANDS	233
5.16.1 Read Command (READ)	233
5.16.2 Write Command (WRIT)	236
5.17 COMM COMMAND	239
5.18 COMMUNICATION COMMANDS	242
5.18.1 MBUS	243
5.18.2 PEER	245
5.18.3 BROD	247
5.18.4 BOOK	249
5.18.5 POLL	252
5.18.6 DIAG	254
5.18.7 SND	257
5.18.8 RCV	259
5.19 MOTION COMMAND	261
5.19.1 Outline of Motion Commands	262
5.19.2 Status	292
5.19.3 M Relay, N Relay	293
5.20 EXTENSIVE REGISTER ACCESS COMMANDS	294
5.20.1 File Read (FRED)	294
5.20.2 File Write (FWRT)	296
SECTION 6 SFC FUNCTIONS	298
6.1 OUTLINE OF SFC	300
6.2 CONFIGURATION OF SFC PROGRAM	301
6.2.1 SFC Flow	301
6.2.2 Action Circuit	302
6.2.3 Transition Condition Circuit	304
6.2.4 SFC Mode Condition Setting	305
6.2.5 SFC Simulated Operation Functions	306
6.3 SFC PROGRAM ELEMENTS	307
6.3.1 Steps	307
6.3.2 Transition	309
6.3.3 Link	309
6.4 STEP EVOLUTION RULE	312
6.4.1 Evolution Condition	312
6.4.2 Most Simple Connection	313
6.4.3 Divergence Connection	314
6.4.4 Simultaneous Divergence Connection	314
6.4.5 Simultaneous Convergence Connection	315
6.4.6 Loop Connection	316
6.4.7 Macro Step Evolution	317
6.5 SFC MODE PROCESSING FUNCTION	318
6.5.1 Initialization Function	319
6.5.2 Presetting Function	320
6.5.3 Resetting Function	322
6.5.4 Precautions for Presetting and Resetting	323

	Page
6.6 SFC PROGRAMMING PROHIBITED ITEMS	324
6.6.1 SFC Storage Prohibited Item	324
6.6.2 SFC Evolution Prohibited Items	331
6.7 SFC VIEW FORMAT	336
6.8 EXAMPLES OF SFC FLOW SEQUENCE	338
6.8.1 Program Example (1)	338
6.8.2 Program Example (2)	341
6.8.3 Program Example (3)	342
6.8.4 Program Example (4)	343
6.8.5 Program Example (5)	346
6.9 METHOD OF EVOLUTION IN VARIOUS CIRCUMSTANCES	348
6.9.1 Evolution of Program Unfinished	348
6.9.2 Evolution with Active Steps Arranged vertically	349
6.10 SFC SIMULATED OPERATION FUNCTIONS	372
6.10.1 Disable Function	372
6.10.2 Hold Function	372
SECTION 7 I/O ALLOCATION	373
7.1 I/O CONFIGURATION	373
7.2 I/O MODULE LAYOUT	374
7.3 I/O NUMBERS	374
7.4 I/O MODULE LOCATION	375
7.5 TYPES OF I/O MODULES AND I/O ALLOCATION	376
7.6 I/O ALLOCATION REFERENCE NO.	377
7.7 HIGH-SPEED STATION	377
7.8 I/O ALLOCATION AND HIGH-SPEED STATION	377
SECTION 8 GL60H, GL70H APPLICATIONS	379
8.1 NOTES ON APPLICATIONS	379
8.1.1 Backup Circuit	379
8.1.2 Interlock	379
8.1.3 Control Panel Layout	379
8.1.4 Local Station Network Processing	379
8.1.5 Remote Station Network Processing	380
8.2 CALCULATION OF MEMORY CAPACITY	380
8.3 PRECAUTIONS FOR USING I/O MODULES	381
8.3.1 Input Module	381
8.3.2 Output Module	385
8.3.3 Connection between I/O Modules	388
8.3.4 External Power Supply	389
8.3.5 Precautions when Installing I/O Module	389
8.4 CONSTRUCTION, INSTALLATION AND WIRING	391
8.4.1 Construction of Control Panel	391
8.4.2 Device Configuration in Control Panel	392
8.4.3 Grounding Wire	396
8.4.4 External Wiring	397
8.5 SPARE PARTS	397
SECTION 9 GL60H, GL70H HANDLING AND MAINTENANCE	398
9.1 GL60H, GL70H INSTALLATION PROCEDURE	398
9.1.1 Installation of Mounting Bases	398
9.1.2 Installation of Modules	398
9.1.3 Connection of I/O Cables	400
9.1.4 Wiring of Modules	402

CONTENTS (Cont'd)

	Page
9.2 IOP AND COMM SETTING AND ERROR INDICATION	402
9.3 REMOTE LINE SETTING	404
9.4 MODULE REPLACEMENT	405
9.5 BATTERY REPLACEMENT	407
9.6 REGISTER ACCESS PANEL	409
9.6.1 Operation Keys	409
9.6.2 Operation	411
9.6.3 Error Codes	419
9.6.4 Status LED Lamp	420
9.7 GL60H, GL70H SYSTEM STATUS	421
9.8 TROUBLESHOOTING	429
APPENDIX A GL60H, GL70H COMPONENTS LIST	437
APPENDIX B OPERATION FUNCTION REFERENCE LIST	441
APPENDIX C DIMENSIONS	454
APPENDIX D Memocon-SC GL60H, GL70H LAYOUT AND DRILLING PLAN	463

INDEX

Subject	Chapter	Par.	Page
1-axis Program Operation (MOVI)	5	5.19.1.1	262
1-axis Single-block Operation (MVIS)	5	5.19.1.2	264
1-axis Zero Point Return Operation (ZRN)	5	5.19.1.6	272
3-axis (2-axis) Current Value Adjustment (ADJ)	5	5.19.1.5	290
3-axis (2-axis) Program Operation (MOVL)	5	5.19.1.4	268
8-point Indication of Input Relay and Output Coil Status	9	9.6.2(9)	416
A Action Circuit	6	6.2.2	302
Addition (ADD)	5	5.5.2	84
All Resetting	6	6.5.3	322
Alarm Reset (ARES)	5	5.19.1.13	286
ALLOWABLE NUMBER OF MEMORY WORDS	4	4.6	44
AND, OR, XOR	5	5.12.3	202
Application Counter Circuits	5	5.4.5	81
Application Timer Circuits	5	5.3.5	73
ARITHMETIC FUNCTIONS	5	5.5	84
Arithmetic Operation	APPENDIX B (4)		442
ASCII COMMANDS	5	5.16	233
Auxiliary Power Supply Module (Type JRMSP-PS21)	APPENDIX C		455
B Backup Circuit	8	8.1.1	379
BASIC GL60H, GL70H SPECIFICATIONS	3	3.1	4
Basic Terminology	5	5.9.1	137
BATTERY REPLACEMENT	9	9.5	407
Battery Replacement Interval	9	9.5(2)	407
Battery Replacement Procedures	9	9.5(3)	408
Battery Specifications	9	9.5(1)	407
BCD→Binary Conversion (BIN)	5	5.11.2	183
Binary and BCD	4	4.3	29
Binary→BCD Conversion (BCD)	5	5.11.3	186
Bit Count (BCNT)	5	5.12.11	225
Block Addition (BADD)	5	5.11.8	198
Block Move 1 with Destination Index (DIBT)	5	5.10.2	167
Block Move 1 with Source Index (SIBT)	5	5.10.4	175
Block Move 2 with Destination Index (DIBR)	5	5.10.3	172
Block Move 2 with Source Index (SIBR)	5	5.10.5	179
Block Move with Index	APPENDIX B (8)		448
Block Move (BLKM)	5	5.9.3	139
BROD	5	5.18.3	247
BOOK	5	5.18.4	249
Byte Composition (BYCM)	5	5.11.7	196
Byte Rearrangement (TWST)	5	5.12.10	223
Byte Split (BYSL)	5	5.11.6	194
C Cables for I/O Signal Lines	8	8.4.4(1)	397
CALCULATION OF MEMORY CAPACITY	8	8.2	380
Caution in Using B2603	8	8.3.1(6)	385
Checksum	4	4.4(5)	34
Circuit for Measuring Scan Time	5	5.4.5(4)	83
Clearing Contents of Holding Registers to 0	5	5.5.11(1)	103
Clock	5	5.4.5(3)	82
Coil	5	5.2.1(7)	55
Coils (0××××)	4	4.3(1)	26
COMM COMMAND	5	5.17	239

INDEX (Cont'd)

Subject	Chapter	Par.	Page
COMM Module Check	9	9.8(4)	431
COMM Module (Type JAMSC-1F61)	APPENDIX C (5)		456
COMMUNICATION COMMANDS	5	5.18	242
COMMUNICATION MODULE	3	3.4	8
Compare (CMPR)	5	5.12.5	207
Complement (COMP)	5	5.12.4	205
CONFIGURATION OF SFC PROGRAM	6	6.2	301
Connection between I/O Modules	8	8.3.3	388
Connection of B2904	8	8.3.2(8)	388
Connection of I/O Cables	9	9.1.3	400
Connection of Solenoid with Diode	8	8.3.2(7)	387
Connection to Contacts	8	8.3.2(1)	385
Connection to Input Equipment with Different Voltage	8	8.3.1(5)	384
Constant Registers (3××××)	4	4.3(9)	29
Construction of Control Panel	8	8.4.1	391
CONSTRUCTION, INSTALLATION AND WIRING	8	8.4	391
Control Panel Layout	8	8.1.3	379
CONTROLLER REFERENCE NUMBERS	4	4.3	25
Convergence	6	6.3.3	310
Convergence inside Simultaneous Sequence	6	6.6.2	331
Cosine (COS)	5	5.8.3	134
Counter	APPENDIX B (3)		442
Counter Configuration	5	5.4.2	75
Counter Inputs	5	5.4.4(2)	79
Counter Link Line	6	6.3.3	311
Counter Outputs	5	5.4.4(3)	79
Counter Transition	6	6.3.2	309
COUNTERS	5	5.4	75
CPU MODULE	3	3.2	4
CPU Module Check	9	9.8(2)	430
CPU Module (Type DDSCR-GL60H, GL70H)	APPENDIX C (1)		454
Creating of Relay Circuits	5	5.2.3	60
Current Value (TIMERS)	5	5.3.2(3)	69
Current Value (COUNTERS)	5	5.4.2(3)	76
Current Value Setting (POS)	5	5.19.1.10	280
Current Values of Counter at Power ON	5	5.4.4(8)	81
D DATA CONVERSION	5	5.11	183
Data Convert	APPENDIX B (9)		448
Data Handling	5	5.6.9(4)	126
DATA MOVE	5	5.9	137
Data Table and Table Size	5	5.9.1	137
Data Transfer	APPENDIX B (7)		445
Device Configuration in Control Panel	8	8.4.2	392
DIAG	5	5.18.6	254
DISABL/ENABL of input Relay or Coil	9	9.6.2(3)	412
Disable Function (SFC SIMULATED OPERATION FUNCTIONS)	6	6.10.1	372
DISABLE FUNCTION	4	4.7	45
Displaying ON/OFF Status of Input Relay 1 ×××× or Coil 0 ××××	9	9.6.2(1)	411
Displaying Contents of Input Register 30 ××××, Constant Register 3 ×××× or Holding Register 4 ××××	9	9.6.2(2)	411
Displaying of Contents of Communication Buffer	9	9.6.2(12)	418

Subject	Chapter	Par.	Page
Displaying Port Parameters	9	9.6.2(10)	416
Divergence	6	6.3.3	310
Divergence Connection	6	6.4.3	314
Divergence in Simultaneous Convergence	6	6.6.2	332
Divide (DIV)	5	5.5.8	95
Double-precision Addition (DADD)	5	5.5.3	86
Double-precision Divide Function (DDIV)	5	5.5.9	98
Double-precision Multiply (DMUL)	5	5.5.7	93
Double-precision Square Root (DSQR)	5	5.7.3	129
Double-precision Subtraction (DSUB)	5	5.5.5	90
Down Counter	5	5.4.3.2	77
Down Counter Function	5	5.4.3.2(1)	77
Down Counter Operation	5	5.4.3.2(2)	78
Dummy Transition	6	6.3.2	309
E Electrical Noise	8	8.4.2(3)	394
Elements and Their Meanings	5	5.12.2	201
Error Codes	9	9.6.3	419
Evolution Condition	6	6.4.1	312
Evolution of Program Unfinished	6	6.9.1	348
Evolution with Active Steps Arranged vertically	6	6.9.2	349
Example Relay Logic Circuit	5	5.2.2	59
Example (FIN, FOUT)	5	5.9.8(4)	155
Example-Application Circuit of Data Move	5	5.9.13	164
Example-Application Circuits of Arithmetic Functions	5	5.5.11	103
EXAMPLES OF SFC FLOW SEQUENCE	6	6.8	338
Example of Stepping Switches	5	5.15.2	231
Execution of Arithmetic Operations (Only One Scanning Cycle)	5	5.5.10(5)	102
Execution of Signed Arithmetic Operations (Only One Scanning Cycle)	5	5.6.9(5)	126
Execution of Square Root (Only 1 Scan Cycle)	5	5.7.4(4)	131
Execution of Trigonometric Function (Only 1 Scan Cycle)	5	5.8.4(5)	136
EXTENSIVE REGISTER ACCESS COMMANDS	5	5.20	294
External Power Supply	8	8.3.4	389
External Wiring	8	8.4.4	397
F Failure History State Display	9	9.6.2	419
First In (FIN)	5	5.9.7	151
First Out (FOUT)	5	5.9.8	153
File Read (FRED)	5	5.20.1	294
File Write (FWRT)	5	5.20.2	296
Flicker Relay	5	5.3.5(5)	74
Form (TIMERS)	5	5.3.2(1)	68
Form (COUNTERS)	5	5.4.2(1)	75
Form of Matrix	5	5.12.2	201
FROM and TO	6	6.3.3	311
Function and Operation of Counter	5	5.4.3	76
Function and Operation of Timer	5	5.3.3	69
G GENERAL	1	1.1	1
Get Controller System Status (STAT)	5	5.9.11	162
GL60H, GL70H APPLICATIONS	8	8	379
GL60H, GL70H COMPONENTS LIST	APPENDIX A		437
GL60H, GL70H CONFIGURATION	2	2	2

INDEX (Cont'd)

Subject	Chapter	Par.	Page
GL60H, GL70H HANDLING AND MAINTENANCE	9	9	398
GL60H, GL70H INSTALLATION PROCEDURE	9	9.1	398
GL60H, GL70H INTERNAL PROCESS	4	4.4	31
GL60H, GL70H LOCAL STATION	8	8.3.5(2)	390
GL60H, GL70H REMOTE STATION	8	8.3.5(1)	390
GL60H, GL70H SPECIFICATIONS	3	3	4
GL60H, GL70H SYSTEM STATUS	9	9.7	421
Grounding Wire	8	8.4.3	396
H Handle Operation (HN DL)	5	5.19.1.8	276
HIGH-SPEED STATION	7	7.7	377
Hold Function	6	6.10.2	372
Holding Registers (4××××)	4	4.3(8)	27
Horizontal Open	5	5.2.1(4)	54
Horizontal Shunt	5	5.2.1(3)	53
I Independent Multi-axis Program Operation (MOV M)	5	5.19.1.3	266
I/O ALLOCATION	7	7	373
I/O ALLOCATION AND HIGH-SPEED STATION	7	7.8	377
I/O ALLOCATION REFERENCE NO.	7	7.6	377
I/O BUFFER MODULE	3	3.5	18
I/O Buffer Module (Type JAMSC-B2110A)	APPENDIX C (9)		458
I/O CABLE	3	3.8	21
I/O Cable (Type JZMSZ-W20-1, -2)	APPENDIX C (13)		460
I/O CONFIGURATION	7	7.1	373
I/O MODULE	3	3.6	18
I/O MODULE LAYOUT	7	7.2	374
I/O MODULE LOCATION	7	7.4	375
I/O Module (Type JAMSC-B2501)	APPENDIX C (14)		460
I/O Module (type JAMSC-B2603)	APPENDIX C (15)		461
I/O NUMBERS	7	7.3	374
I/O Processor Module Check	9	9.8(3)	431
I/O Section Check	9	9.8(7)	433
IMPORTANT MACHINE CONCEPTS	4	4	24
INDEXED BLOCK MOVE	5	5.10	167
Indication of Data of Timer Register 50 ×××	9	9.6.2(5)	414
Indication of Link Register Data	9	9.6.2(6)	414
Inductive Load	8	8.3.1(1)	381
Initial Step	6	6.3.1	307
Initialization Function	6	6.5.1	319
Input Dummy Resistor	8	8.3.1(2)	381
Input Module	8	8.3.1	381
Input Module Check	9	9.8(8)	435
Input of Square Root	5	5.7.4(2)	130
Input of Trigonometric Function	5	5.8.4(2)	136
Input Pulse Width	5	5.4.4(7)	80
Input Registers (30×××)	4	4.3(7)	27
Input Relays (1××××)	4	4.3(3)	26
Inputs of Arithmetic Logic	5	5.5.10(2)	101
Inputs of Signed Arithmetic Logic	5	5.6.9(2)	125
Install a new module.	9	9.4(5)	406
Install the terminal block and connector.	9	9.4(6)	406
Installation of I/O Signal Lines Cables	8	8.4.4(1)	397
Installation of Modules	9	9.1.2	398

Subject	Chapter	Par.	Page
Installation of Mounting Bases	9	9.1.1	398
Interface Cable (Type JZMSZ-W1015-T)	APPENDIX C (17)		462
Interlock	8	8.1.2	379
INTRODUCTION	1	1	1
IOP AND COMM SETTING AND ERROR INDICATION	9	9.2	402
IOP Module (Type JAMSC-IF60)	APPENDIX C (4)		455
J Jog Operation (JOG)	5	5.19.1.7	274
L Large-capacity Counter	5	5.4.5(1)	81
Latched Coil	5	5.2.1(8)	56
Latched Relay Circuit	5	5.2.4(5)	66
Leakage Current from the Output Module	8	8.3.2(6)	386
Leakage Current in Input Equipment	8	8.3.1(3)	382
Limit of Count	5	5.3.1(3)	68
Limit of Count	5	5.4.1(3)	75
Limiting Contents of Holding Register	5	5.5.11(3)	104
Link	6	6.3.3	309
Link Coil	5	5.2.1(9)	57
Link Coils (D××××)	4	4.3(2)	26
Link Line	6	6.3.3	310
Link Registers (R××××)	4	4.3(11)	30
Local Station Network Processing	8	8.1.4	379
Local Station Scan Time	4	4.5.2(1)	36
Long-time Timer	5	5.4.5(2)	82
Loop	6	6.3.3	310
Loop Connection	6	6.4.6	316
Loosen the screws fastening the module. [Fig. 9.10 (b)]	9	9.4(3)	405
M M Relay, N Relay	5	5.19.3	293
Macro Entry and Macro Return	6	6.3.3	311
Macro Step	6	6.3.1	308
Macro Step Evolution	6	6.4.7	317
Main Power Supply Module (Type JRMSI-PS60)	APPENDIX C (2)		454
MATRIX	5	5.12	200
Matrix	APPENDIX B (10)		449
Maximum Load Current	8	8.3.2(2)	385
MBUS	5	5.18.1	243
MB21 Mounting Base (Type JRMSI-MB21)	APPENDIX C (11)		459
MB60 Mounting Base (Type JRMSI-MB60)	APPENDIX C (10)		458
MB70 Mounting Base (Type JRMSI-MB70)	APPENDIX C (12)		457
Memocon-SC GL60H, GL70H LAYOUT AND DRILLING PLAN	APPENDIX D		463
METHOD OF EVOLUTION IN VARIOUS CIRCUMSTANCES	6	6.9	348
Modify (MBIT)	5	5.12.6	211
MODULE REPLACEMENT	9	9.4	405
Monitor (MON)	5	5.19.1.9	278
Most Simple Connection	6	6.4.2	313
MOTION COMMAND	5	5.19	261
MOUNTING BASE	3	3.7	20
Mounting Bases and I/O Cables	8	8.4.2(1)	392
MOUNTING PRECAUTIONS	APPENDIX D		463
Move of Bit Pattern (Application of BLKM)	5	5.9.13(2)	165
Multi-Rotate (MROT)	5	5.12.9	220
Multiply (MUL)	5	5.5.6	92

INDEX (Cont'd)

Subject	Chapter	Par.	Page
N Network Number	4	4.4(4)	33
NETWORKS	4	4.2	25
NO, NC Contacts	5	5.2.1(1)	52
Normally-ON Circuit	5	5.2.4(1)	65
NOTES ON APPLICATIONS	8	8.1	379
O OFF-delay Timer	5	5.3.5(2)	74
ON/OFF Conditions of DC Input Module	8	8.3.1(4)	383
ON-delay Timer	5	5.3.5(1)	74
One-shot Timer	5	5.3.5(3)	73
Operation	9	9.6.2	411
Operation	5	5.9.8(3)	154
OPERATION FUNCTION REFERENCE LIST	APPENDIX B		441
Operation Keys	9	9.6.1	409
Operations of Square Root and Trigonometric Function	APPENDIX B (6)		445
ORed Outputs of Arithmetic Operations	5	5.5.10(5)	102
Oscillator	5	5.2.4(3)	65
Outline of Motion Commands	5	5.19.1	262
OUTLINE OF SFC	6	6.1	300
Output Fuse	8	8.3.2(4)	386
Output Module	8	8.3.2	385
Output Module Check	9	9.8(9)	436
Output of Square Root	5	5.7.4(3)	131
Outputs of Arithmetic Logic	5	5.5.10(3)	102
Outputs of Signed Arithmetic Logic	5	5.6.9(3)	126
Outputs of Trigonometric Function	5	5.8.4(3)	136
P P150 PROGRAMMING PANEL	3	3.9	22
Parameter Setting (PRM)	5	5.19.1.11	282
PEER	5	5.18.2	245
Pointer (DATA MOVE)	5	5.9.1(3)	137
Pointer (MATRIX)	5	5.12.2(4)	201
POLL	5	5.18.5	252
Power Supply Circuit	8	8.4.2(4)	394
POWER SUPPLY MODULE	3	3.3	6
Power Supply Module Check	9	9.8(1)	429
Power-up Sequence	4	4.4(1)	32
Precautions	6	6.6.2	333
Precautions for Presetting and Resetting	6	6.5.4	323
PRECAUTIONS FOR USING I/O MODULES	8	8.3	381
Precautions when Installing I/O Module	8	8.3.5	389
Preset Value (TIMERS)	5	5.3.2(2)	69
Preset Value (COUNTERS)	5	5.4.2(2)	76
Preset Value of Counter	5	5.4.4(5)	80
Preset Value of Timer	5	5.3.4(5)	72
Presetting and resetting inside Simultaneous Divergence and Simultaneous Convergence	6	6.5.4	323
Presetting Function	6	6.5.2	320
Prevention of Double Reservation (Application of SRCH)	5	5.9.13(4)	166
Program Example (1)	6	6.8.1	338
Program Example (2)	6	6.8.2	341
Program Example (3)	6	6.8.3	342
Program Example (4)	6	6.8.4	343
Program Example (5)	6	6.8.5	346
Programming Arithmetic Logic	5	5.5.10(1)	101

Subject	Chapter	Par.	Page
Programming Arithmetic Logic	5	5.6.9(1)	125
Programming Arithmetic Logic and Precautions	5	5.5.10	101
Programming Counter Circuit	5	5.4.4(1)	79
Programming Counter Circuit and Precautions	5	5.4.4	79
Programming Data Move Circuit and Precautions	5	5.9.12	163
PROGRAMMING FUNCTION LIST	5	5.1	47
PROGRAMMING FUNCTIONS	5	5	47
Programming Indexed Block Move Circuit and Precautions	5	5.10.6	182
Programming Panel P150 (Type DISCT-P150)	APPENDIX C	(16)	461
Programming Signed Arithmetic Logic and Precautions	5	5.6.9	125
Programming Square Root Circuit	5	5.7.4(1)	130
Programming Square Root Circuit and Precautions	5	5.7.4	130
Programming Timer Circuit	5	5.3.4(1)	71
Programming Timer Circuit and Precautions	5	5.3.4	71
Programming Trigonometric Function Circuit	5	5.8.4(1)	135
Programming Trigonometric Function Circuit and Precautions	5	5.8.4	135
Pulse Generating Circuit	5	5.3.5(4)	74
Pulse Generating Circuit for Initialization	5	5.2.4(2)	65
R Range of Reference Numbers	4	4.3(12)	30
RCV	5	5.18.8	259
Read Command (READ)	5	5.16.1	233
Reference Number	5	5.12.2(2)	201
Reference Numbers of Source and Destination	5	5.9.1(4)	138
REGISTER ACCESS PANEL	9	9.6	409
Register Access Panel: RAP (Type DISCT-IF69)	APPENDIX C	(8)	457
Register-to-Table Move (R→T)	5	5.9.4	142
Relationship of Preset and Current Values	5	5.3.4(6)	72
Relationship of Preset and Current Values	5	5.4.4(6)	80
Relay Logic Elements	5	5.2.1	51
Relay, Coil	APPENDIX B	(1)	441
RELAYS	5	5.2	51
Remote I/O Adapter (Type T-02698)	APPENDIX C	(18)	462
Remote I/O Processing Time	4	4.5.2(2)	37
REMOTE LINE SETTING	9	9.3	404
Remote Station Network Processing	8	8.1.5	380
Remove the module. [Fig. 9.10 (c)]	9	9.4(4)	405
Remove the terminal block and connector. [Fig. 9.10 (a)]	9	9.4(2)	405
Reservation (Application of FIN and FOUT)	5	5.9.13(3)	166
Resetting Function	6	6.5.3	322
RIOD Module Check	9	9.8(5)	432
RIOD Module (Type JAMSC-IF62)	APPENDIX C	(6)	456
RIOR Module Check	9	9.8(6)	432
RIOR Module (Type JAMSC-IF70)	APPENDIX C	(7)	457
Rotate (BROT)	5	5.12.8	217
S Sample Application Circuits of Relays	5	5.2.4	65
Sample Down Counter	5	5.4.3.2(3)	78
Sample Timer	5	5.3.3(3)	70
Sample Up Counter	5	5.4.3.1(3)	77
Saving I/O Modules	5	5.5.11(6)	106
Scaling	5	5.5.11(5)	106
Scan Cycle	4	4.4(2)	32
Scan Time	4	4.5.2	36
SCANNING	4	4.5	34
Segments	4	4.5.3(1)	41

INDEX (Cont'd)

Subject	Chapter	Par.	Page
Self-Holding Circuit (Memory Circuit)	5	5.2.4(4)	66
Sense (SENS)	5	5.12.7	213
Servo ON (SVON)	5	5.19.1.14	288
Set and Reser Function	5	5.2.1(11)	58
Setting and Reading Data (Application of R→T and T→R)	5	5.9.13(1)	164
Setting Constant or Contents of a Register to Holding Register	5	5.5.11(2)	104
Setting or Altering of Communication Parameters	9	9.6.2(11)	417
Setting or Altering of Constant Register, Holding Register	9	9.6.2(4)	413
SFC Evolution Prohibited Items	6	6.6.2	331
SFC Flow	6	6.2.1	301
SFC FUNCTIONS	6	6	298
SFC Mode Condition Setting	6	6.2.4	305
SFC MODE PROCESSING FUNCTION	6	6.5	318
SFC PROGRAM ELEMENTS	6	6.3	307
SFC PROGRAMMING PROHIBITED ITEMS	6	6.6	324
SFC Simulated Operation Functions	6	6.2.5	305
SFC SIMULATED OPERATION FUNCTIONS	6	6.10	372
SFC Storage Prohibited Item	6	6.6.1	324
SFC VIEW FORMAT	6	6.7	336
Sign Representation	5	5.6.2	108
Signed Addition (SADD)	5	5.6.3	109
SIGNED ARITHMETIC FUNCTIONS	5	5.6	108
Signed Arithmetic Operation	APPENDIX B (5)		444
Signed Divide (SDIV)	5	5.6.8	122
Signed Double-precision Addition (SDAD)	5	5.6.4	112
Signed Double-Precision Subtraction (SDSB)	5	5.6.6	117
Signed Multiply (SMUL)	5	5.6.7	120
Signed Subtraction (SSUB)	5	5.6.5	115
Simultaneous Convergence	6	6.3.3	310
Simultaneous Convergence Connection	6	6.4.5	315
Simultaneous Divergence	6	6.3.3	310
Simultaneous Divergence Connection	6	6.4.4	314
Sine (SIN)	5	5.8.2	132
SKIP	5	5.13	227
Skip, Subroutine	APPENDIX B (10)		453
SND	5	5.18.7	257
Solving a Ladder Circuit	4	4.5.1	34
Solving by a Two-Level Scan	4	4.5.3(2)	42
Solving Sequence of Active Steps Created by Presetting	6	6.5.4	323
Sort (SORT)	5	5.11.5	191
Source and Destination	5	5.9.1(2)	137
SPARE PARTS	8	8.5	397
SQUARE ROOT	5	5.7	127
Square Root (SQRT)	5	5.7.2	127
Status	5	5.19.2	292
Status Indication of Link Coil (Relay)	9	9.6.2(7)	415
Status Indication of SFC Step	9	9.6.2(8)	415
Status LED Indicator for AC Output Module	8	8.3.2(5)	386
Status LED Lamp	9	9.6.4	420
STEP EVOLUTION RULE	6	6.4	312
Step Relays (S×××)	4	4.3(4)	26
Stepping Relays (2XY××)	4	4.3(5)	26
STEPPING SWITCH	5	5.15	230
Stepping Switch Functions and Operation	5	5.15.1	230

Subject	Chapter	Par.	Page
Stepping Switch (Having a few Steps)	5	5.5.11(4)	105
Steps	6	6.3.1	307
Storing Counter Current Value	5	5.4.4(4)	80
Storing Timer Current Value	5	5.3.4(4)	72
SUBROUTINE	5	5.14	228
Subtraction (SUB)	5	5.5.4	88
Swap (SWAP)	5	5.11.4	189
System Errors	9	9.6.3(2)	420
T Table Search (SRCH)	5	5.9.9	157
Table Set (TSET)	5	5.9.10	160
Table Size	5	5.12.2(3)	201
Table-to-Register Move (T→R)	5	5.9.5	145
Table-to-Table Move (T→T)	5	5.9.6	148
Timer	APPENDIX B	(2)	441
Timer Configuration	5	5.3.2	68
Timer Error	5	5.3.4(7)	72
Timer Function	5	5.3.3(1)	69
Timer Inputs	5	5.3.4(3)	71
Timer Operation	5	5.3.3(2)	69
Timer Outputs	5	5.3.4(3)	71
Timer Registers (5××××)	4	4.3(10)	30
TIMERS	5	5.3	68
TRACE BACK FUNCTION	4	4.8	45
Transition	6	6.3.2	309
Transition Coil	5	5.2.1(10)	58
Transition Condition Circuit	6	6.2.3	304
Transitional Contact Circuit	5	5.2.4(6)	67
Transitional Contacts	5	5.2.1(2)	52
TRIGONOMETRIC FUNCTION	5	5.8	132
TROUBLESHOOTING	9	9.8	429
Turn off supply power	9	9.4(1)	405
Turn on power	9	9.4(7)	406
Two-Level Scan	4	4.5.3	41
Types	5	5.3.1(1)	68
Types	5	5.4.1(1)	75
Types of Arithmetic Functions	5	5.5.1	84
Types of Counters	5	5.4.1	75
Types of Data Conversion	5	5.11.1	183
Types of Data Move	5	5.9.2	138
TYPES OF I/O MODULES AND I/O ALLOCATION	7	7.5	376
Types of Indexed Block Move	5	5.10.1	167
Types of Matrix	5	5.12.1	200
Types of Signed Arithmetic Functions	5	5.6.1	108
Types of Square Root	5	5.7.1	127
Types of Timers	5	5.3.1	68
Types of Trigonometrix Function Operations	5	5.8.1	132
U Unit of Count	5	5.4.1(2)	75
Unit of Operand	5	5.8.4(4)	136
Unit of Time	5	5.3.1(2)	68
Up Counter	5	5.4.3.1	76
Up Counter Function	5	5.4.3.1(1)	76

INDEX (Cont'd)

Subject	Chapter	Par.	Page
Up Counter Operation	5	5.4.3.1(2)	76
Up/Down Counter	5	5.4.5(5)	83
USER PROGRAM CONFIGURATION	4	4.1	24
V Variable Value Setting (VAR)	5	5.19.1.12	284
Vertical Open	5	5.2.1(6)	54
Vertical Shunt	5	5.2.1(5)	54
W Watching Timer	4	4.4(3)	33
Weight (Table 8.5)	8	8.4.2(2)	393
Wiring in Panel	8	8.4.2(5)	396
Wiring of Modules	9	9.1.4	402
Write Command (WRIT)	5	5.16.2	236
Wrong Operation	9	9.6.3(1)	419

SECTION 1

INTRODUCTION

1.1 GENERAL

A Programmable Controller (PLC) is a solid-state device designed to perform logic decision making for industrial control applications. The PLC can be used as a direct replacement for relays or solid-state electronics in an industrial environment. Features of a PLC compared to standard industrial control devices include the following:

- Solid-state for maximum reliability.
- Programmed with simple ladder diagram language.
- Easily reprogrammed with a programming panel if requirements change.
- Controller is reusable if no longer required for original application.
- Indicator lights provided at major diagnostic points to simplify troubleshooting.
- Maintenance is simple, based upon module replacement, and insures minimum downtime and maximum production.
- Communicates with a central computer for machine monitoring, and data gathering and reporting.

The GL60H/GL70H Controllers are the finest example of PLC technology available today. These controllers are small-size, but large-capacity programmable controllers including the following some of the advantages in addition to all of the above features and the functions belonging to the Memcon-SC GL60S (GL60S) series. These PLCs are very suitable for creating your Factory Automation, as master models of the GL60S series.

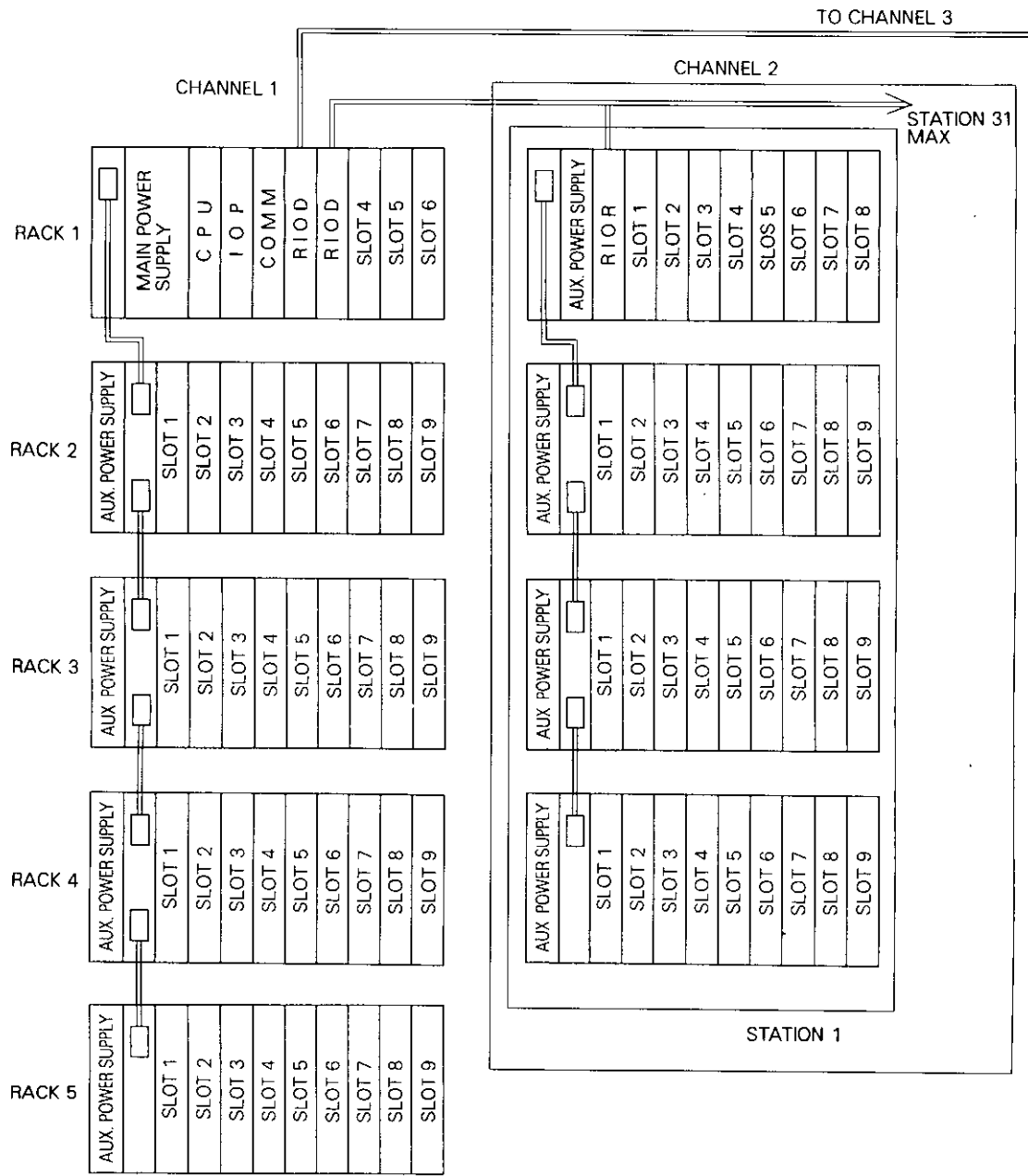
- Low cost — hardware cost less than installed relays.
- Compact — smaller cabinet size and less floor space required.
- Upgraded Sequential Function Chart (SFC)
- Higher speed control
- Reinforcement of motion control functions
- Larger-capacity ladder program memory (only for GL70H)
- Real-Time, On-Line Programming — a YASKAWA standard of excellence for maximum flexibility

SECTION 2

GL60H, GL70H CONFIGURATION

Table 2.1 GL60H, GL70H Configuration

Component	Description
CPU Module (CPU)	<p>The CPU module includes a logic solver and memory. The ladder circuits are stored in the memory and solved according to input data sent from an I/O driver. The results are output to the I/O driver.</p> <p>The program memory is available in 32k words (GL60H), 64k words (GL70H). The CPU module GL60S is capable of dealing with discrete inputs/outputs (ON/OFF signals) of up to 4,096 points and up to 512 register inputs/outputs (5-digit decimal or 16-bit binary data).</p>
Main Power Supply Module	The main power supply module supplies DC power to CPU module, optional modules and I/O modules in rack 1 of CH1.
Auxiliary Power Supply Module	The auxiliary power supply module supplies DC power to I/O buffer module and I/O modules in each expanding I/O rack.
I/O Processor Module (IOP)	The I/O processor module includes two RS-232C ports (MEMOBUS) for communication with the Programming Panel and a computer. By operating the front Register Access Panel (RAP), it is possible to display the status of the coils and input relays, to perform simulation (forced ON/OFF), to display and alter the contents of the registers, and to set and display communication parameters. I/O driver is incorporated communication module.
Expanding Communication Module (COMM)	This module is used to expand the communication ports. For communication with the programming panel and a computer, two RS-232C ports (MEMOBUS) are also available.
I/O Buffer Module (IOB)	The I/O buffer module is used for rack 2 or higher.
RAP (Register Access Panel) Module	<ul style="list-style-type: none"> • For coil or input relay: Monitoring ON/OFF status, disabling (forced ON/OFF). • Displaying or altering register contents. • Setting or displaying communication parameters. • Monitoring ON/OFF status of coil or input relay by status indication LED. • I-scan pulse monitoring available.
I/O Modules (2000 Series)	<ul style="list-style-type: none"> • Discrete signal modules: One module is provided with inputs or outputs of 16, 32 or 64 circuits. It is usable for numeric signals (by I/O allocation). • Numeric signal modules: One module is provided with eight numeric inputs or outputs of 16 bits. • Analog modules: An A/D converter module has eight circuits and a D/A converter module of two circuits. • Other modules: Counter module, positioning module
Mounting Base	The CPU module, power supply module, peripheral modules, and I/O modules are mounted on a mounting base. The type of the mounting base (three types) varies with the type of module. The modules mounted on the base are connected to each other via a built-in mother board. Connections between mounting bases are made with cables.
Programming Panel	The programming panel permits storing a program, altering or deleting the stored program, monitoring status, and printing out a ladder diagram through a connected printer.



2

Note:

1. Slots 1 to 3 are available in rack 1 of channel 1 (local). If COMM and RIOD are not used, up to 6 slots are available. And up to 31 stations can be installed for channels 2 and 3 (remote) each; up to 4 racks can be installed for each station.
2. Up to 256 modules each for discrete input, discrete output, register input and register output, up to 1024 total I/O modules, can be installed. Although the combination of input and output modules can be arranged freely, there are the following limitations:
 - Discrete input + discrete output \leq 4096
 - Register input + register output \leq 512

SECTION 3 GL60H, GL70H SPECIFICATIONS

3.1 BASIC GL60H, GL70H SPECIFICATIONS

Table 3.1 Basic GL60S Specifications

Items	Specifications
Power Supply	Single-phase 85 to 132 VAC, 47.5 to 63Hz
Consumed Power	150 VA (main power supply module), 70 VA (aux. power supply module)
Holding Time	10 ms
Ambient Temperature	0 to + 55°C (excluding peripheral devices)
Storage Temperature	-20°C to + 85°C (excluding lithium battery)
Humidity	30% to 95% relative (non-condensing)
Vibration-Resistance	In compliance with JIS* C 0911 (excluding peripheral devices)
Shock-Resistance	10 G max (excluding peripheral devices)
Environmental Condition	Free from explosive, inflammable, corrosive gases
Grounding	Grounding resistance: 100Ω or less
Dielectric Strength	1500 VAC for 1 minute
Insulation Resistance	100 MΩ or more at 500 VDC
Noise Immunity	1500 Vp-p, pulse width: 1 μs, rising time: 1 ns

*Japanese Industrial Standard

3.2 CPU MODULE

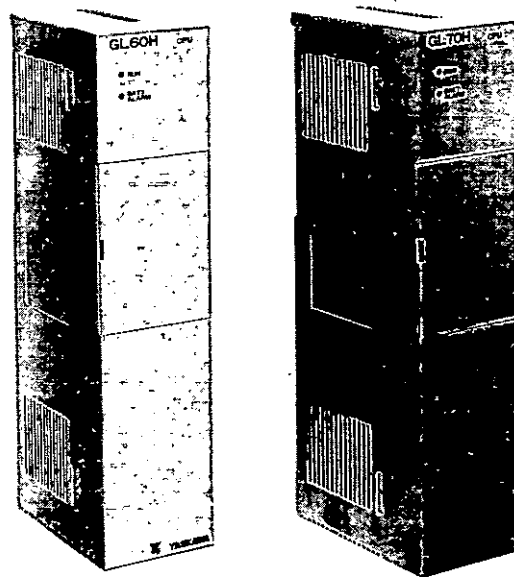


Fig. 3.1 CPU Module

Table 3.2 CPU Module Specifications

Items	Specifications	
Type	• DDSCR-GL60H (32k), DDSCR-GL70H (64k)	
Control Method	Stored program and scan control	
Programming	Relay ladder diagram symbology, SFC (Sequential Function Chart)	
Program Memory Size	32k CMOS RAM with battery back-up (24-bit per word) 32K words (GL60H), 64K words (GL70H)	
Data Memory Size	9999 words holding registers, 4096 words constant registers, expanding register, CMOS RAM with battery back-up (16-bit per word)	
Scan Time	0.083 μ s per word (basic instruction)	
Logic Function	Relay	<ul style="list-style-type: none"> • Normally open contact, normally closed contact • Transitional contact (OFF to ON), or (ON to OFF) • Horizontal shunt, vertical shunt, vertical open • Coil, latched coil, stepping relay
	Timer	<ul style="list-style-type: none"> • Type: Seconds, tenths of seconds, hundredths of seconds • Maximum preset value: 4-digit decimal • Setting available from external device
	Counter	<ul style="list-style-type: none"> • Up counter, down counter • Maximum preset value: 4-digit decimal • Setting available from external device
	Arithmetic	• (Double-precision) addition, (double-precision) subtraction, (double-precision) multiply, (double-precision) divide (in 4- or 8-digit decimal)
	Arithmetic with Sign	(Double-precision) addition with sign, (double-precision) subtraction with sign, multiply with sign, divide with sign (in 4- or 8-digit decimal)
	Square Arithmetic	Square root (SQRT), Double square root (DSQR)
	Trigonometric Function	Sine, Cosine
	Move	R→T, T→R, T→T, BLKM, FIN, FOUT, SRCH, STAT, TSET
	Move with Index	<ul style="list-style-type: none"> • DIBT, DIBR • SIBT, SIBR
	Data Convert	BIN, BCD, SWAP, SORT, BYSL, BYCM, BADD
	Matrix	AND, OR, XOR, COMP, CMPR, MBIT, SENS, BROT, MROT, TWST, BCNT, FRED, FWRT, motion Command, COMM Command
	Special Function	GOSUB, SKIP
ASCII	READ, WRITE	
Input/Output Points	<ul style="list-style-type: none"> • Discrete I/O points: Input+Output \leq 4096 points (512 bytes) • Register I/O points: Input+Output \leq 512 registers (1024 bytes) • No. of local channels: 1 (42 I/O modules max in use per channel) • No. of remote channels: 2 (31 stations per channel, No. of I/O points per station: IN \leq 512 bytes (DI + (16 \times RI) \leq 4096) OUT \leq 512 bytes (DO + (16 \times RO) \leq 4096) 	
Diagnostic Function	<ul style="list-style-type: none"> • Checksum of memory • Watchdog timer checking • Battery monitoring • Internal code checking • Reference number checking • I/O allocation checking • Memory diagnostic 	
Backed-up Memory	<ul style="list-style-type: none"> • Type: 1-lithium battery • Battery life: 5 years, at 25°C • Memory contents holding time: 1 year, at 25°C 	
Indicating Lamp	<ul style="list-style-type: none"> • RUN: Lights when CPU module is proper in operation. • BATT ALARM: Lights when the output voltage of CMOS RAM back-up battery is low level, with AC power supply turned on. 	
Mounting Location	On mounting base MB60 (CPU base)	
Dimensions in mm	60 (W) \times 250 (H) \times 100 (D)	
Approx Mass	0.6kg	

3

3.3 POWER SUPPLY MODULE

(1) Main Power Supply Module

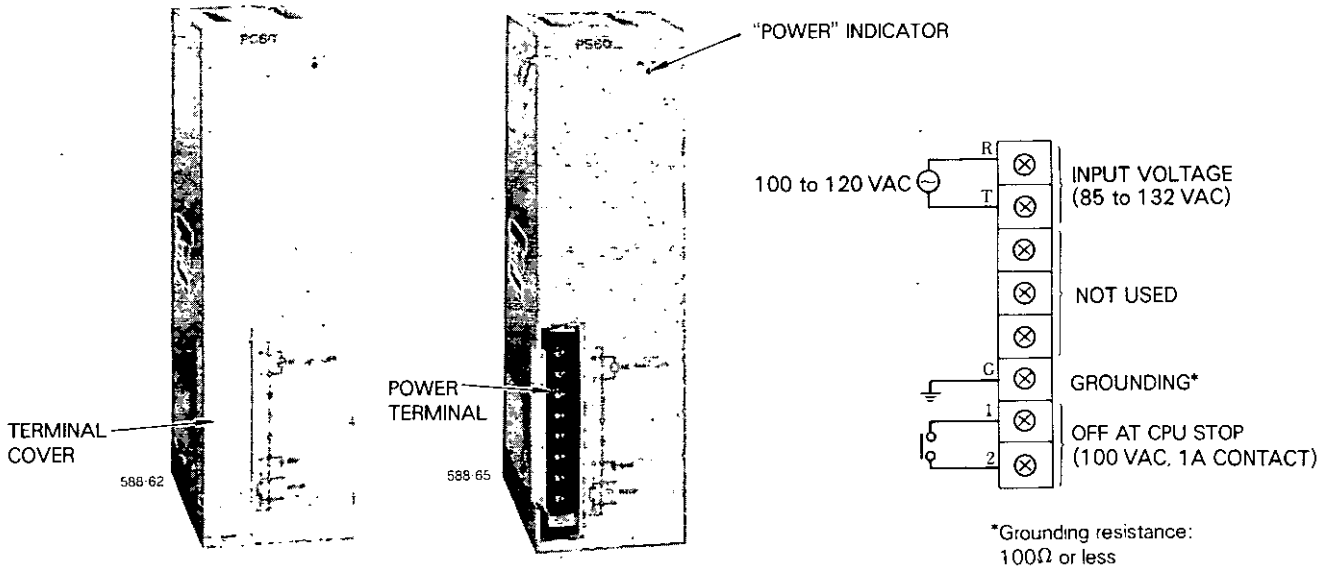


Fig. 3.2 Main Power Supply Module

Table 3.3 Main Power Supply Module Specifications

Items	Specifications
Type	JRMSP-PS60
Function	DC power supply for a CPU module, function modules (IOP, etc.), and I/O (CH1 rack 1) modules
Input Voltage	Single-phase 85-132 VAC, 47.5-63 Hz, 150 VA
Transient Input Voltage	0-154 VAC (10 ms)
Inrush Current	30 A (peak) or less
Leakage Current	1 mA or less
Fuse	Glass tube fuse (5 A)
Indicating Lamp	POWER: Lights when power supply is proper.
Monitoring Contact	STOP: ON at GL60H, GL70H running, OFF at GL60H, GL70H stop
Mounting Location	On mounting base MB60 (CPU base)
Dimensions in mm	75 (W) × 250 (H) × 94 (D)
Approx Mass	0.9kg

(2) Auxiliary Power Supply Module

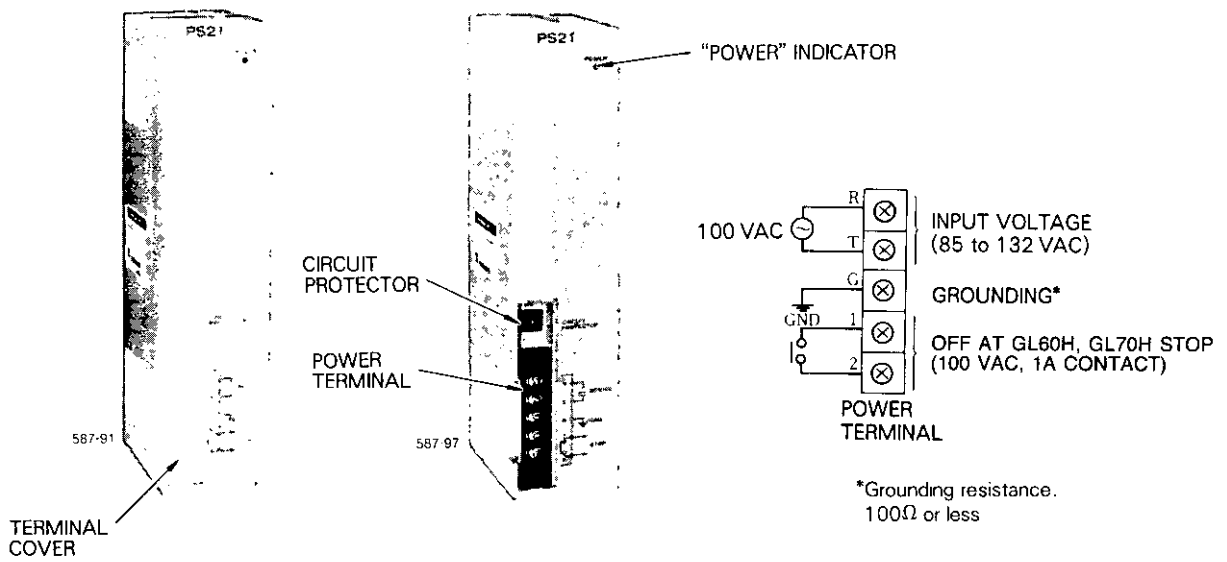


Fig. 3.3 Auxiliary Power Supply Module

Table 3.4 Auxiliary Power Supply Module Specifications

Items	Specifications
Type	JRMSP-PS21
Function	DC power supply for I/O buffer module and I/O modules.
Input Voltage	Single-phase 85-132 VAC, 47-63 HZ, 70 VA
Transient Input Voltage	0-154 VAC (10 ms)
Inrush Current	30 A (peak) or less
Leakage Current	0.2 mA or less
Fuse	Circuit protector (3A)
Indicating Lamp	POWER: Lights when power supply is proper.
Monitoring Contact	STOP : ON at GL60H, GL70H running, OFF at GL60H, GL70H stop
Mounting Location	On mounting bases MB22 and MB70
Dimensions in mm	60 (W) × 250 (H) × 94 (D)
Approx Mass	0.7kg

3.4 COMMUNICATION MODULE

(1) I/O Processor Module (IOP)

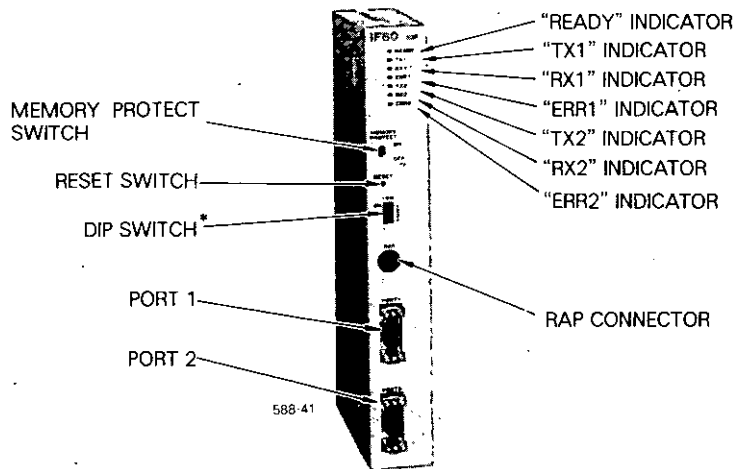


Fig. 3.4 IOP Module

Table 3.5 IOP Module Specifications

Items		Specifications
Type	JAMSC-IF60	
Function	<ul style="list-style-type: none"> • For communication with P150 programming panel and a computer using 2 MEMOBUS ports (slaves). • Discrete I/O status indication, disable operation, register contents indication, set (with register access panel) • Local I/O driver built-in 	
Communication Port	No. of Ports	2 ports per module
	Communication Specification	EIA RS-232C
	Baud Rate	19200/9600/4800/2400/1200/600/300/150
	Data Bits	7 or 8 bits
	Parity	Even, odd or non
	Stop Bits	1 or 2 bits
	Protocol	MEMOBUS protocol
	Transmission Check	CRC-16 or LRC
	Connector	D-SUB 9 pin
Indicating Lamp	<ul style="list-style-type: none"> • READY: Lights when IOP module is proper. • TX1: Lights at port 1 transmitting. • RX1: Lights at port 1 receiving. • ERR1: Lights at port 1 communication error. • TX2: Lights at port 2 transmitting. • RX2: Lights at port 2 receiving. • ERR2: Lights at port 2 communication error. 	
Mounting Location	On mounting base MB60 (CPU base)	
Dimensions in mm	37.5 (W) × 250 (H) × 94 (D)	
Approx Mass	0.6 kg	

* For DIP switch setting, refer to Par.9.2

(2) Expanding Communication Module (COMM)

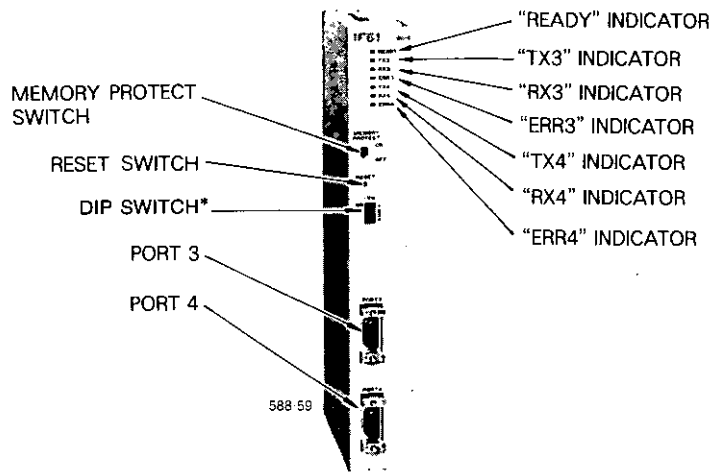


Fig. 3.5 COMM Module

Table 3.6 COMM Module Specifications

Items		Specifications
Type		JAMSC-IF61
Function		For more communication with P150 and a computer using 2 MEMOBUS ports (slaves).
Communication Port	No. of Ports	2 ports per module
	Communication Specification	EIA RS-232C
	Baud Rate	19200/9600/4800/2400/1200/600/300/150
	Data Bits	7 or 8 bits
	Parity	Even, odd or non
	Stop Bits	1 or 2 bits
	Protocol	MEMOBUS protocol
	Transmission Check	CRC-16 or LRC
	Connector	D-SUB 9 pin
Indicating Lamp (excluding RAP)		<ul style="list-style-type: none"> • READY: Lights when COMM module is proper. • TX3: Lights at port 3 transmitting. • RX3: Lights at port 3 receiving. • ERR3: Lights at port 3 communication error. • TX4: Lights at port 4 transmitting. • RX4: Lights at port 4 receiving. • ERR4: Lights at port 4 communication error.
Mounting Location		On mounting base MB60 (CPU base)
Dimensions in mm		37.5 (W) × 250 (H) × 94 (D)
Approx Mass		0.5 kg

* For DIP switch setting, refer to Par.9.2.

(3) Remote I/O Driver Module (RIOD)

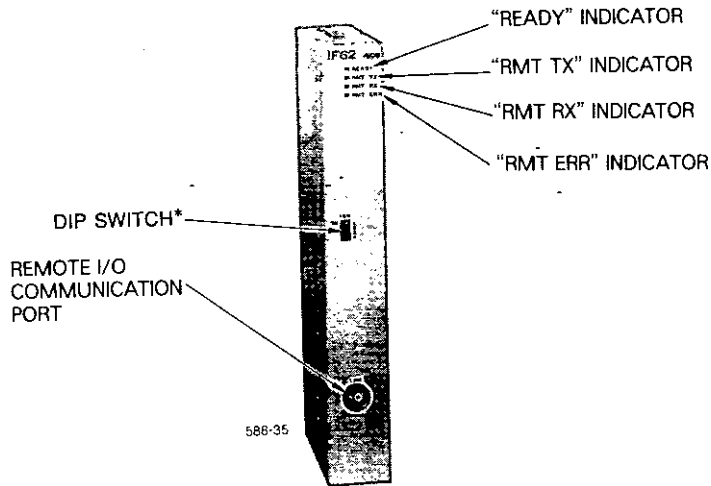


Fig. 3.6 RIOD Module

Table 3.7 RIOD Module Specifications

Items	Specifications	
Type	JAMSC-IF62	
Function	<ul style="list-style-type: none"> · For I/O modules in remote Use. · As master station in remote communication line. · For use of ASCII module. 	
Remote Communication	Topology	Bus
	Transmission media	Coaxial cable
	Transmission method	Baseband (Manchester coding)
	Data baud rate	500Kbps/1Mbps/2Mbps/4Mbps (Selected by SW)
	Max. cable length	1km (in use of IIC-FB)
	Max. No. of station	31 Stations
	Troubleshooting	<ul style="list-style-type: none"> · Failure station : Automatically disconnected from the line. · Recovered station: Automatically connected to the line.
Indicating Lamp	<ul style="list-style-type: none"> · READY : Lights when RIOD module is proper. · RMT TX : Lights at remote 1 transmitting. · RMT RX : lights at remote 1 receiving. · RMT ERR : Lights at remote 1 communication error. 	
Mounting Location	On mounting base MB60(CPU base)	
Dimensions in mm	37.5 (W) × 250 (H) × 94 (D)	
Approx Mass	0.5kg	

* For DIP switch setting, refer to Par.9.3.

(4) Remote I/O Receiver Module (RIOR)

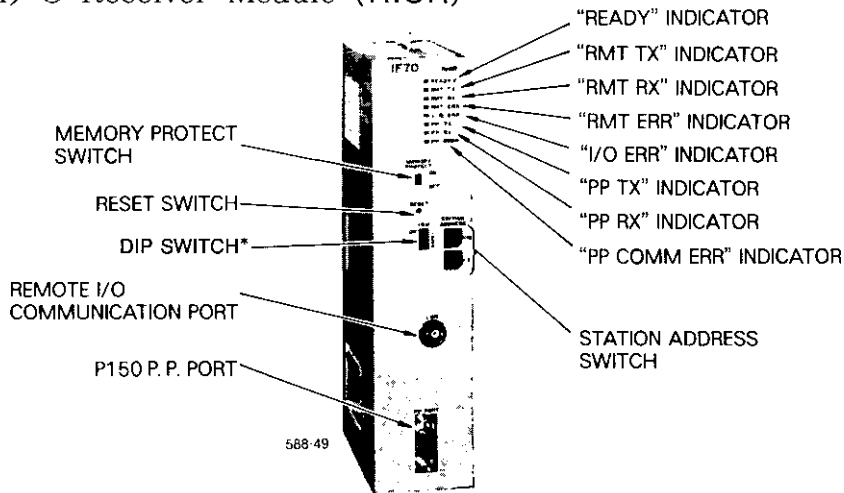


Fig. 3.7 RIOR Module

Table 3.8 RIOR Module Specifications

Items	Specifications	
Type	JAMSC-IF 70	
Function	<ul style="list-style-type: none"> • For I/O modules in remote use. • As each slave station in remote communication line. I/O modules driven via a local I/O bus. • One communication port provided for P150: For programming, monitoring. 	
Remote Communication	Topology	Bus
	Transmission media	Coaxial cable
	Transmission method	Baseband (Manchester coding)
	Data baud rate	500Kbps/1Mbps/2Mbps/4Mbps (Selected by SW)
	Max. cable length	1 km (in use of 11 C-FB)
	Max. No. of station	31 Stations
	Troubleshooting	<ul style="list-style-type: none"> • Failure station: Automatically disconnected from the line. • Recovered station: Automatically connected to the line.
Communication Port	Number of ports	1
	Transmission mode	EIA RS-232C
	Baud rate	9600 bauds
	Device address	1
	Data	8 bits
	Parity check	EVEN
	Connector	9-pin D subconnector
	Connected equipment	P150
Indicating Lamp	<ul style="list-style-type: none"> • READY: Lights when RIOR module is proper. • RMT TX: Lights at remote transmitting. • RMT RX: Lights at remote receiving. • RMT ERR: Lights at remote communication error. • I/O ERR: Lights at communication error. • PP TX: Lights at PP transmitting. • PP RX: Lights at PP receiving. • PP COMM ERR: Lights at PP communication error. 	
Mounting Location	On mounting base MB70 (RIOR base)	
Dimensions in mm	60 (W) × 250 (H) × 94 (D) mm	
Approx Mass	0.6kg	

* For DIP switch setting, refer to Par.9.9.

(5) PC Link Module

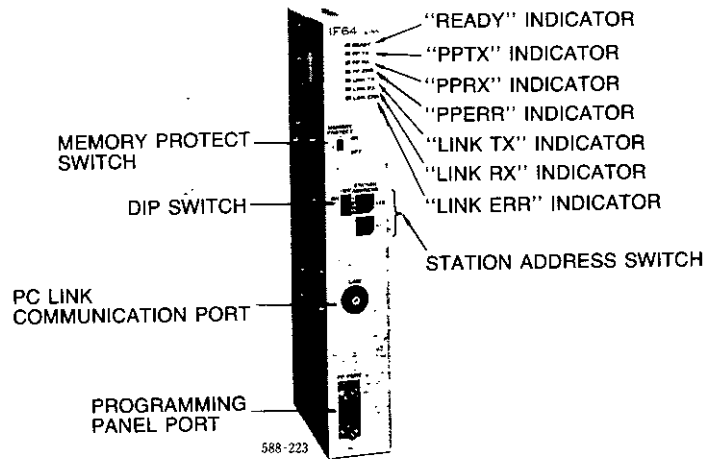


Fig. 3.8 PC Link Module

Table 3.9 PC Link Module Specifications

Items	Specifications	
Type	JAMSC-IF64	
Function	Interconnection between PLCs using exclusive-use reference	
PC Link	Topology	Bus
	Transmission media	Coaxial cable
	Transmission method	Baseband (Manchester coding)
	Data baud rate	500 kbps/1 Mbps/2 Mbps/4 Mbps (Selected by SW)
	Max. cable length	1 km (depending on transmission speed and cable used)
	Max. No. of stations	32 Stations
	Troubleshooting	<ul style="list-style-type: none"> • Failure station : Automatically disconnected from the line. • Recovered station : Automatically connected to the line.
Communication Port	Number of ports	1
	Transmission spec.	EIA RS-232C
	Transmission speed	9600 bauds
	Data	8 bits
	Stop bit	1 bit
	Parity check	EVEN
	Protocol	MEMOBUS protocol
Connector	9-pin D subconnector	
Indicating Lamp	<ul style="list-style-type: none"> • READY : Lights when module is proper. • PP TX : Lights at PP port transmitting. • PP RX : Lights at PP port receiving. • PP ERR : Lights at PP port communication error. • LINK TX : Lights at link port transmitting. • LINK RX : Lights at link port receiving. • LINK ERR : Lights at link port communication error. 	
Consumed Power	5VDC 1.3A	
Mounting Location	On mounting base MB60 (CPU base)	
Dimensions in mm	37.5 (W) × 250 (H) × 94 (D) mm	
Approx. Mass	0.6kg	

(6) 3200 IF Module (NSB)

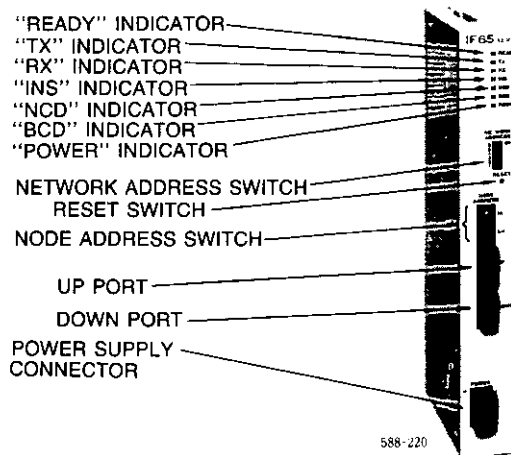


Fig. 3.9 NSB Module

Table 3.10 NSB Module Specifications

Items	Specifications	
Type	JAMSC - IF65	
Function	<ul style="list-style-type: none"> • For N : M plant communication system construction • Interconnection of programmable controller Memocon-SC series devices and personal computers, etc. via optical communication network 	
YENET	Topology	Ring
	Transmission media	2 - core fiber cable (PCF)
	Transmission method	Baseband (Manchester coding)
	Transmission speed	2 Mbps
	Max. cable length	1 km (in use of PCF cable)
	Max. No. of stations	126 stations
	RAS function	Automatic loop - back, node bypass
Indicating Lamp	<ul style="list-style-type: none"> • READY : Lights when 3200 IF module is proper. • TX : Lights at transmitting. • RX : Indicates the number of receiving. • INS : Lights when communication is possible. • NCD : } Lights when both are proper. • BCD : } Upper side loop - back when only BCD is off. • POWER : Lower side loop - back when only NCD is off. • POWER : Lights when power is supplied from power supply connector. 	
Mounting Location	On mounting base MB60 (CPU base)	
Dimensions in mm	37.5 (W) × 250 (H) × 94 (D) mm	
Approx. Mass	0.6 kg	

(7) Servo Interface Module

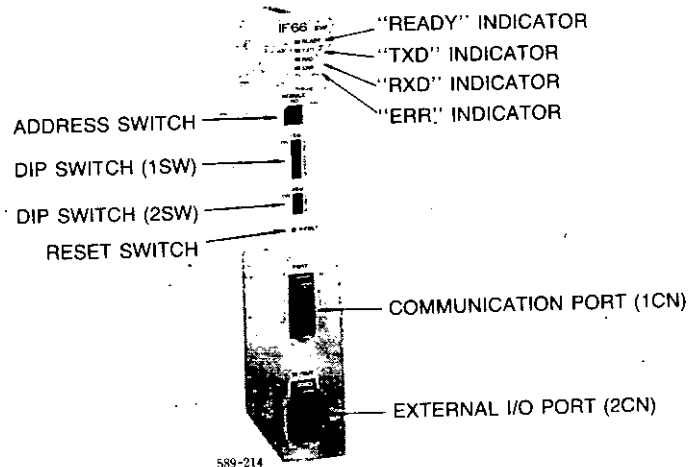


Fig. 3.10 Servo Interface Module

Table 3.11 Servo Interface Module Specifications

Items	Specifications	
Type	JAMSC-IF66	
Function	<ul style="list-style-type: none"> For motion control realized by connecting with Servopack CACR-HR □□ BA Interface between CPU module and Servopack Up to 8 CACR-HR □□ BAs possible to connect by IF66 module 	
Communication Port (1CN)	Transmission spec.	EIA standard, RS-422
	Transmission media	Semi-double asynchronous method
	Transmission speed	9600 bauds
	Bit construction	JIS 7-unit type 10 bits
	Transmission distance	Up to 30m
	Error check	Parity (EVEN), framing, BCC
External I/O (2CN)	Input	Servo alarm input (8) During servo alarm run input
	Output	IF66 proper operation input (1)
Switch Setting	Address switch	<ul style="list-style-type: none"> Designation of IF66 module address 0 to 3 possible to set Same switch cannot be used twice.
	Dip switch (1SW)	Connecting servo axis provided/not provided • 1 SW - 1 : Servo with axis No. 1 provided • 1 SW - 8 : Servo with axis No. 8 provided
	Dip switch (2 SW)	2 SW - 1 : External input signal selected by ON 2 SW - 2 : For future use 2 SW - 3 : For future use 2 SW - 4 : Test mode selected by ON
Indicating Lamp	<ul style="list-style-type: none"> READY : Lights when IF66 module is proper. TXD : Lights at data transmitting. RXD : Lights at data receiving. (Red LED lights at receiving error.) ERR : ROM, RAM check error, WDT error, system reset 	
Mounting Location	On mounting base MB60 (CPU base)	
Dimensions in mm	37.5 (W) × 250 (H) × 94 (D) mm	
Approx Mass	0.6 kg	

(8) ASCII Module (ASCII)

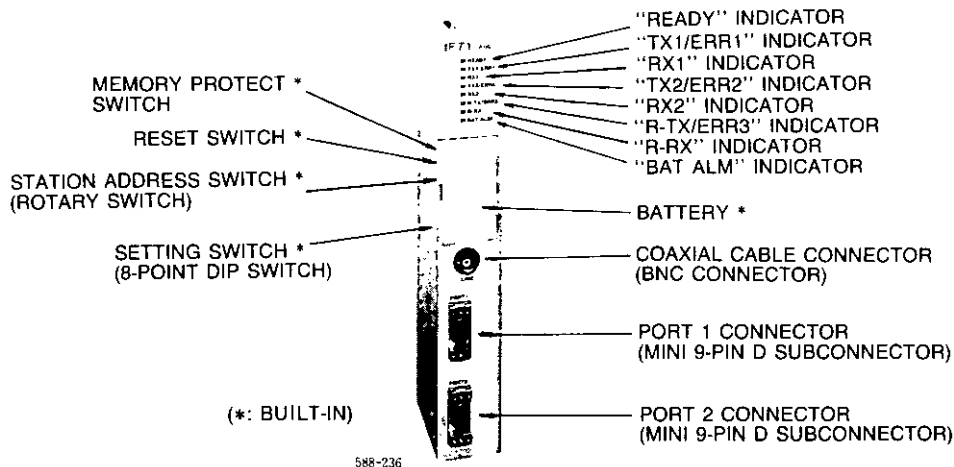


Fig. 3.11 ASCII Module

Table 3.12 ASCII Module Specifications

Items	Specifications	
Type	JAMSC - IF71	
Memory Capacity	64 k bytes	
Message	Up to 1024 messages Up to 1024 bytes/message	
Usable Number of Registers	999 registers/message	
ASCII Port	No. of ports	2 ports
	Transmission spec.	EIA RS - 232C, semi - double asynchronous method
	Transmission speed	19200/9600/4800/2400/1200/600/300bauds
	Data length	5 to 8 bits
	Parity check	EVEN, ODD or no parity
	Stop bit	1 or 2 bits
	Connector	9 - pin D subconnector
Indicating Lamp	<ul style="list-style-type: none"> • READY : Lights when ASCII module is proper. • TX1/ERR1 : Lights at port 1 transmitting. (Green) • RX1 : Lights at port 1 receiving. (Green) • TX2/ERR2 : Lights at port 2 transmitting. (Green) • RX2 : Lights at port 2 receiving. (Green) • R - TX/ERR3 : Lights at remote line transmitting. (Green) • R - RX : Lights at remote line receiving. (Green) • BAT ALM : Lights at battery voltage reduction. (Red) 	
Mounting Location	On mounting base MB60, MB70, MB71, MB21	
Dimensions in mm	37.5 (W) × 250 (H) × 94 (D) mm	
Approx. Mass	1 kg	

(9) FMGC Control Module

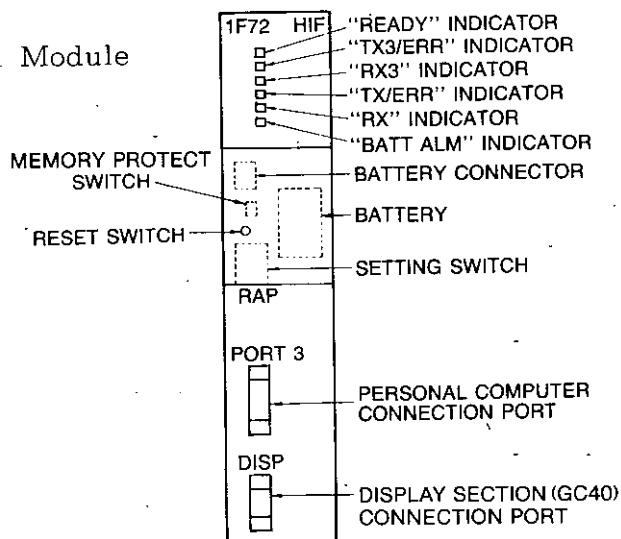


Fig. 3.12 FMGC Control Module

Table 3.13 FMGC Control Module Specifications

Items	Spacifications
Type	JAMSC - IF72
Function	To control simplified FA monitor FMGC display section
Memory Capacity	256 k bytes (CMOS - RAM battery backup)
Communication Port	Personal computer communication port (port 3) RS232C 150 to 19200 bauds
	Display section (GC40) communication port RS422 9600 bauds
	RAP port
Indicating Lamp	<ul style="list-style-type: none"> • READY : Lights when IF72 is proper. • TX3/ERR : Lights at port 3 transmitting or transmission error. • RX3 : Lights at port 3 receiving. • TX/ERR : Lights at display section port transmitting or CRT port transmission error. • RX : Lights at display section port receiving. • BATT/ALM : Lights at battery alarm.
No - power - loss Clock	Second, minute, hour, day of week, day of month, month, year
Mounting Location	On mounting base MB60 (CPU base)
Dimensions in mm	37.5 (W) × 250 (H) × 94 (D) mm
Approx. Mass	0.6 kg

Note : IF61 cannot be mounted when IF72 is mounted.

(10) Register Access Panel (RAP)

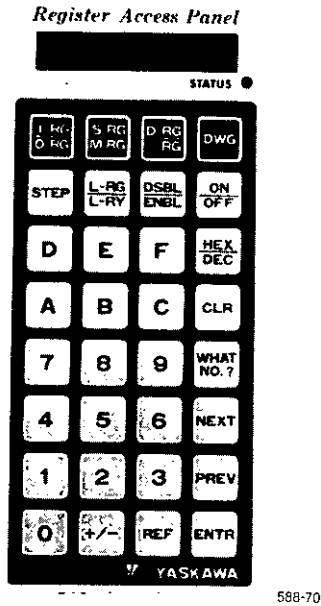


Fig. 3.13 Register Access Panel

Table 3.14 RAP Specifications

Items	Specifications
Type	DISCT-IF69
Function	<ul style="list-style-type: none"> • For coil or input relay: monitoring ON/OFF status, disabling (forced ON/OFF). • Displaying or altering register contents. • Setting or displaying communication parameters. • Monitoring ON/OFF Status of coil or input relay by status indication LED. • 1-scan pulse monitoring available.
Indication	<ul style="list-style-type: none"> • 16-segment, 8-digit alphanumeric LED • Status indication LED
Mounting Location	On I/O processor module
Dimensions in mm	70 (W) × 155 (H) × 19.5 (D)
Approx Mass	0.3kg

3.5 I/O BUFFER MODULE

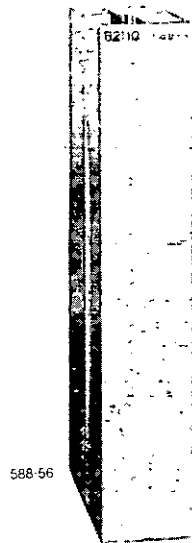


Fig. 3.14 I/O Buffer Module

Table 3.15 I/O Buffer Module Specifications

Items	Specifications
Type	JAMSC-B2110 A
Function	· I/O bus buffer · For use of rack 2, 3, 4 or 5
Connector	2 connectors for cables between racks (W20-1, -2)
Mounting Location	On mounting base MB22
Dimensions in mm	480 (W) × 250 (H) × 21 (D)
Approx Mass	0.4kg

3.6 I/O MODULE

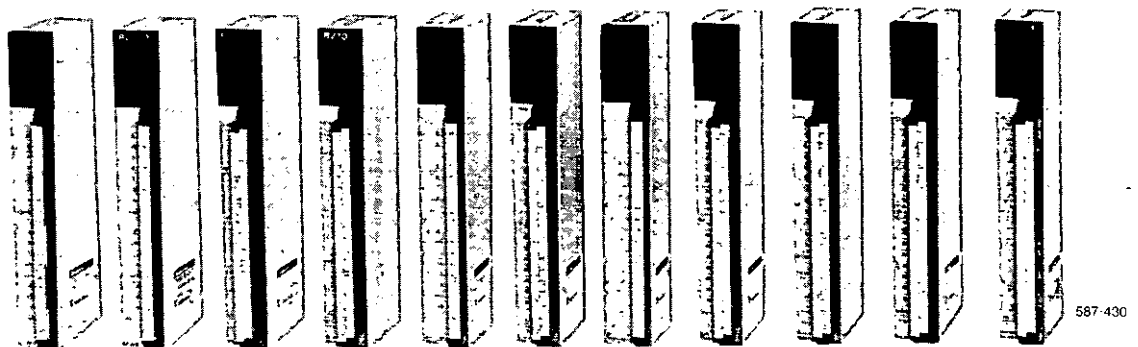


Fig. 3.15 I/O Modules

Table 3.16 I/O Module Specifications

Modules	Items	Type JAMSC -	Voltage	Current	Input Impedance	External Power Supply	Maximum Response Time	No. of I/Os	
Input	AC	100 V	B 2501	100 VAC	10 mA	10 kΩ (at 50 Hz)	-	OFF → ON 15 ms ON → OFF 25 ms	16
		200 V	B 2503	200 VAC	10 mA	20 kΩ (at 50 Hz)	-		16
		100 V	B 2505	100 VAC	10 mA	10 kΩ (at 50 Hz)	-		32
		200 V	B 2507	200 VAC	10 mA	20 kΩ (at 50 Hz)	-		32
	DC	12/24 V	B 2601	12/24 VDC	5/10 mA	2.4 kΩ (at 50 Hz)	-	OFF → ON 5 ms	16
		48 V	B 2611	48 VDC	9.4 mA	5k Ω (at 50 Hz)	-		16
		12/24 V	B 2603	12/24 VDC	5/10 mA	2.4 kΩ (at 50 Hz)	-	OFF → ON 5 ms	32
		12/24 V	B 2605	12/24 VDC	2.5/5 mA	4.7 kΩ	-	ON → OFF 10 ms	64
		5/12 V	B 2607	5/12 VDC	4/11 mA	1.2 kΩ	-	0.5 ms	32
	Register	B 2701	12/24 VDC	8 mA/24 VDC	2.4 kΩ	Terminal - type, positive logic	32/64 ms switching	8	
		B 2711	12/24 VDC	8 mA/24 VDC	2.4 kΩ	Connector - type, negative logic	32/64/192/ 320 ms switching	8	
	Analog	B 2703	0 to 10 v	-	-	Refer to "Memocon-SC GL20, GL60S ANALOG MODULES 2000 Series" (SIE-C815-13.9).	-	8	
		B 2733	-10 to +10 V	-	-		-	8	
		B 2734	1 to 5 V	4 to 20 mA	1MΩ or more		-	8	
	Instrumentation	B 2705	-	-	-	Contact your Yaskawa representative.	-	4	
Output	AC	100/200 V	B 2500	100/200 VAC	1 A per output 3 A Per 8 outputs	-	With CR, varistor. Fuse : 7.5 A per 8 outputs	OFF → ON 1 ms ON → OFF 1/2 cycle + 1 ms	16
		100/200 V	B 2504	100/200 VAC	0.3 A per output 1.2 A per 8 outputs	-	With CR. Fuse : 5 A per 16 outputs		32
	DC	12/24 V	B 2600	12/24 VDC	2 A per output 5 A per 8 outputs	-	Fuse : 7.5 A per 8 outputs	1 ms	16
		12/24 V	B 2602	12/24 VDC	0.3 A per output 0.6 A per 4 outputs	-	-	1 ms	32
		12/24 V	B 2602 A	12/24 VDC	0.3 A per output 0.6 A per 4 outputs	-	With fuse	1 ms	32
		12/24 V	B 2604	12/24 VDC	0.1 A per output 0.4 A per 8 outputs	-	-	1 ms	64
		5/12 V	B 2606	5/12 VDC	20 mA per output 640 mA per 32 outputs	-	-	1 ms	32
	Relay Contact	B 2902	• 220 VAC, 1 A (Induction load, PF : 0.4) • 110 VAC, 1.2 A (Induction load, PF : 0.4) • 24 VDC, 1 A (Induction load, TM ↑ : 15 ms)	-	-	Relay coil voltage : 24 VDC ±10% Min. operating voltage, current : 5 V, 10mA	OFF → ON 10 ms ON → OFF 15 ms	32	
		B 2904	• 220 VAC, 0.5 A (Induction load, PF : 0.6) • 110 VDC, 0.3 A (Induction load, TM ↑ : 40 ms)	-	-	Relay coil voltage : 24 VDC ±5% Min. operating voltage, current : 5 V, 1 mA	5 ms	16	
		B 2914	• 220 VAC, 0.5 A (Induction load, PF : 0.6) • 110 VDC, 0.3 A (Induction load, TM ↑ : 40 ms)	-	-	Relay coil voltage : 24 VDC ±5% Min. operating voltage, current:24VDC, 10mA	5 ms	16	
	Register	B 2700	12/24 VDC	100 mA per output	-	Terminal - type negative logic	32/64 ms switching	8	
		B 2710	12/24 VDC	100 mA per output	-	Connector - type, negative logic	32/64/192/ 320 ms switching	8	
	Analog	B 2702	0 to 10 V	-	-	Refer to "Memocon - SC GL20, GL 60 S ANALOG MODULES 2000 Series" (SIE - C 815 - 13.9).	-	2	
		B 2712	0 to 5 V	-	-		2		
		B 2722	-5 to + 5 V	-	-		2		
B 2732		-10 to + 10 V	-	-	2				
B 2742		4 to 20 mA	-	-	2				
Motion Control	Reversible Counter	B 2801	-	-	-	With coincidence output	-	2	
	Preset Counter	B 2802	-	-	-	-	-	1	
	Positioning	B 2803	-	-	-	-	• Analog output • For absolute	-	1
		B 2813	-	-	-	-	Pulse output	-	1
		B 2823	-	-	-	-	For pulse motor, servomotor	-	1
	PID Module	B 2800	-	-	-	PID control	-	1	
	Instrumentation Module	B 2705	-	-	-	Instrumentation input	-	4	
	MEMOLINK Master	B 2804	-	-	-	Connecting to I/O MEMOLINK of 1000 series is possible	-	1	
MEMOLINK Slave	B 2805	-	-	-	-		1		

7 ↑ : Time constant

3.7 MOUNTING BASE

(1) MB60 Mounting Base

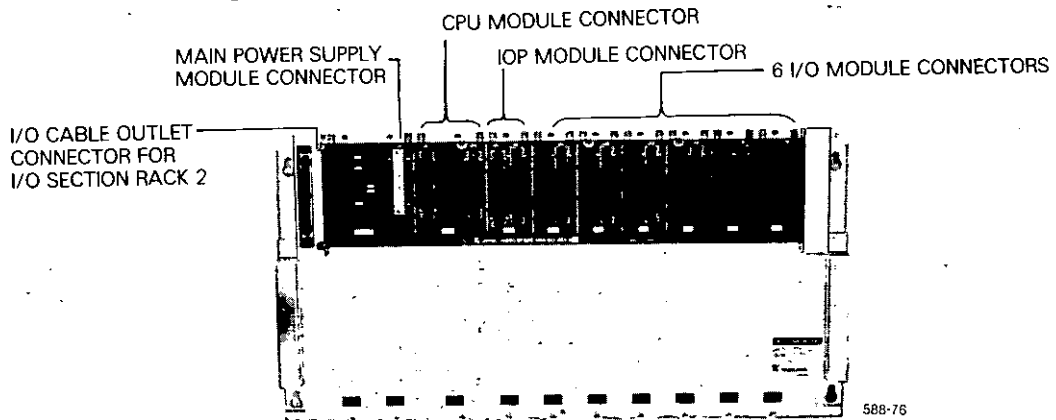


Fig. 3.16 MB60 Mounting Base

Table 3.17 Mounting Base MB60 Specifications

Items	Specifications
Type	JRMSI-MB60
Application	For mounting main power supply module, CPU module, I/O processor module, remote I/O driver module and up to 6 I/O modules.
Dimensions in mm	480 (W) × 250 (H) × 21 (D)
Approx Mass	1.4 kg

(2) MB22A Mounting Base

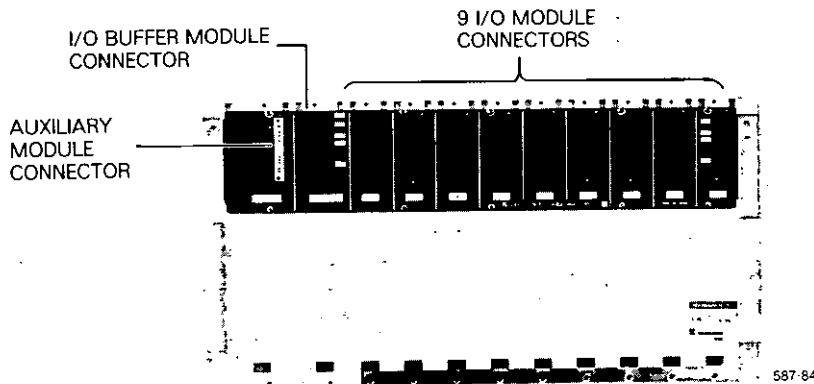


Fig. 3.17 MB22A Mounting Base

Table 3.18 Mounting Base MB22A Specifications

Items	Specifications
Type	JRMSI-MB22A
Application	<ul style="list-style-type: none"> For I/O expansion For mounting auxiliary power supply module, I/O buffer module and up to 9 I/O modules.
Dimensions in mm	480 (W) × 250 (H) × 21 (D)
Approx Mass	1.3 kg

(3) MB70 Mounting Base

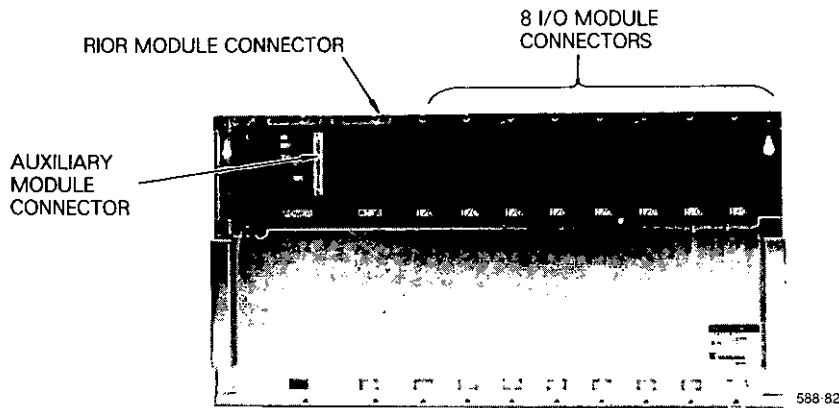


Fig. 3.18 MB70 Mounting Base

Table 3.19 Mounting Base MB70 Specifications

Items	Specifications
Type	JRMSI-MB70
Applicaition	For mounting auxiliary power supply module, remote I/O receiver module, and up to 8 I/O modules.
Dimensions in mm	480 (W) × 250 (H) × 21 (D)
Approx Mass	1.3 kg

3.8 I/O CABLE

The following types of cables are available for connection across each mounting base, respectively.

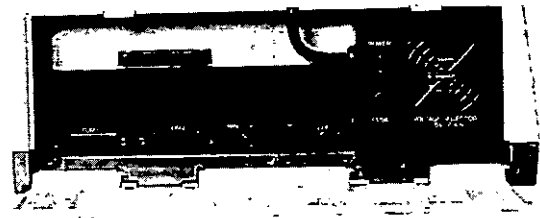
Table 3.20 I/O Cable Specifications

Items	Specifications	
Type	JZMSZ-W20-1	JZMSZ-W20-2
Length	0.5 m	1.5 m
Application	Used for connecting across each mounting base, respectively.	

3.9 P150 PROGRAMMING PANEL



586 243



586 251

(a) Front

(b) Rear

Fig. 3.19 P150 Programming Panel

Table 3.20 P150 Programming Panel Specifications

Items		Specifications
Type		DISCT-P150-10 (Standard keyboard—raised key) DISCT-P150-11 (Membrane keyboard—flush key)
Functions *		<ul style="list-style-type: none"> • Programming, adding, altering and deleting of logic and data • Logic entering, logic display • Load, dump and verify functions • Definition of system configuration • I/O allocation • Various monitorings, file control
Attachments	Graphic Display	Plasma display: 640 dot × 400 dot
	Keyboard	Label keys, function keyboard, numeric keyboard, ASCII keyboard, cursor control keys
	Floppy Disk Drive	Two 3.5-inch floppy disk drive
	Video	• Monitor type: Black and white, raster scan CRT (Type PC 8841 made by NEC Corp. is available.)
	Communication Port †	• One parallel port (made by Centronics Data Computer Corp.) • Two EIA RS-232C ports
Connection of P150 to Communication Module		Type W1015-T1 cable (length: 5 m) or W1015-T2 cable (length: 15 m)
Standard Specifications	Power	85 to 132 VAC/195 to 265 VAC, single phase, changeover at 50/60 Hz (47.5 to 63 Hz)
	Dissipated Power	120 VA
	Ambient Temperature	+5°C to +45°C
	Storage Temperature	-20°C to +60°C
	Humidity	20% to 80% relative (non-condensing)
	Atmosphere	Free from explosive, inflammable, corrosive gases, and dust.
	Grounding	Chassis grounding line is connected to mainframe grounding line via connecting cable of CPU module.
	Approx Mass	9 kg

* A control program in a 3.5-inch floppy disk must be loaded in the P150.

† A ladder diagram can be printed out through a printer connected to the P150.

<3.5-inch Floppy Disk>

Before using the P150 programming panel, a control program in a 3.5-inch floppy disk must be loaded in the P150. Select one of the following 3.5-inch floppy disks suitable for the intended use.

(1) **GL60H, GL70H Programmer FD (F70H-E001)**

This is used to make a program and control load, dump, verify and file functions with P150, and also used to perform I/O allocation.

(2) **GL60H, GL70H Ladder Lister FD (F70H-E002)**

This is used to print out a ladder diagram through a printer connected to the P150.

(3) **Blank FD (F150-000)**

A blank floppy disk is used to record the ladder circuits and/or SFC stored in the CPU module memory by using a programmer floppy disk. The contents of the registers and I/O allocation are recorded together.

SECTION 4 IMPORTANT MACHINE CONCEPTS

4.1 USER PROGRAM CONFIGURATION

A user program consists of a ladder program and an SFC flow. The area of a ladder program, which is 32k words (GL60H), 64k words (GL70H) in size, is divided into the ladder, action circuit, transition condition circuit and subroutine circuit according to the purpose of the user program. The size of each area can be defined by the user. The size of the SFC flow is fixed and cannot be altered.

The ladder circuit contains programs which cannot be described in SFC or programs to be processed continuously regardless of the system state.

The subroutine circuit is used to contain subroutines. When the same lines repeatedly appear in a program in the ladder circuit to perform the same processing several times, these lines are made into a subroutine to save memory.

For the action circuit and the transition condition circuit, see Section 6, SFC Functions.

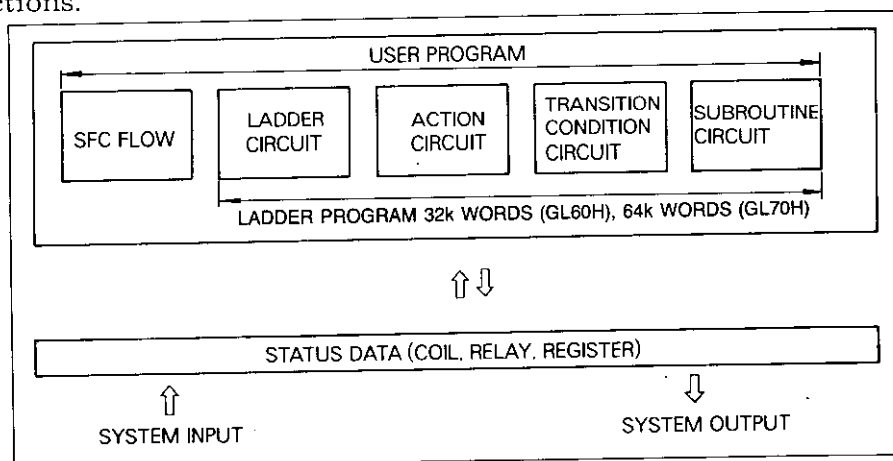


Fig. 4.1 User Program Configuration

Table 4.1 shows the relationships between the user program and the status data.

Table 4.1 Relationships between the User Program and the Status Data

Status Data	Ladder and Subroutine Circuits	Action Circuit	Transition Condition Circuit	SFC Flow
Input Relay	#	#	#	×
Input Register	#	#	#	×
Coil	○	○	#	×
Holding Register	○	○	○	×
Constant Register	#	#	#	×
Link Relay*	○*	○*	○*	×
Link Register*	○*	○*	○*	×
Step Relay	#	×	×	○
Stepping Relay	○	○	○	×
Transition Coil	×	×	○	×
Timer Register	#	×	×	×
M, N Code Relay	#	#	#	×

○: Can be updated. #: Can be referenced. ×: Cannot be accessed.

*Only the status data allocated at the local station can be updated.

4.2 NETWORKS

The GL60H, GL70H program is composed in units of network (Only one network for transition condition circuit). The multi-node format allows for up to ten elements of the program in each horizontal rung of the ladder diagram. Up to seven of these rungs can be combined into a network of relay contacts and other programming elements (timers, counters, etc.); each network can have up to seven coils placed at the extreme right of the network.

The networks are stored in the memory of the CPU module in the order of the network numbers.

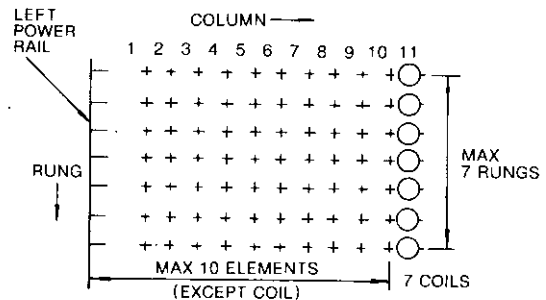


Fig. 4.2 Networks

4.3 CONTROLLER REFERENCE NUMBERS

Throughout the programming of the GL60H/GL70H Controllers, five-digit reference numbers (four-digit for only step relay) are utilized to build the user's logic.

These references are divided into two broad categories: discretives and registers. Discrete references are used for individual items that can be either ON or OFF, such as limit switches, pushbuttons, relay contacts, motor starters, relay coils, solenoid valves, etc. Register references are used to store numerical values such as counts, times, analog values, etc.; all register references are three BCD digits long (maximum value 9999). Since there are four bits per BCD digit, registers can also be 16 bits of data.

Only nine types of references are required to program the GL60H/GL70H Controllers. Any specific reference can be used as many times as required by the particular application; there are no limitations on the number of times a reference is used. References are defined as shown in Table 4.2.

Table 4.2 The Number of Reference

Reference Number	Elements
0 × × × ×	Coils (including latched coil) and their contacts (including transitional contact)
D × × × ×	Link coils and their contacts
1 × × × ×	Input relays (including transitional contact)
S × × ×	Step relays
2YY × ×	Stepping relays
M × × YZ	M code relays
N00 YZ	N code relays
30 × × ×	Input registers
4 × × × ×	Holding registers
3 × × × ×	Constant registers
5 × × × ×	Timer registers
R × × × ×	Link registers
A × × × ×	Extended registers

(1) Coils (0 × × × ×)

Other than the coils used in a relay circuit, coils can be used for any results of processing such as timer, counter, arithmetic operations, data move, data convert, matrix, etc.

Coils are divided into two types: normal coil (not retentived when power is OFF) and latch coil (retentived even after power is OFF). They are divided into two groups from another standpoint. One is the internal coil (auxiliary relay) which is used only in a ladder circuit but not as an external output. The other is the output coil which drives an external device via an output module.

The coils are identified by specific coil numbers. The coil contacts (having the same reference numbers as the coils) may be used repeatedly in any network.

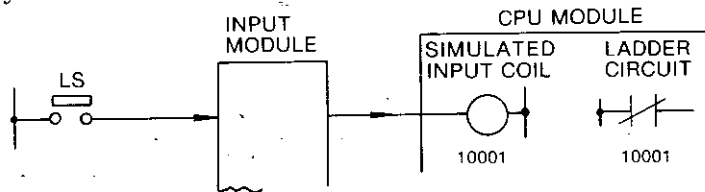
NOTE Coil 08192 cannot be used in a program since it is used as a battery monitor coil. However, its contact may be used repeatedly in a program.

(2) Link Coils (D × × × ×)

The GL60H, GL70H can link PCs (up to 32 PCs) by its PC link system to perform high speed data link. The ON/OFF state of the coil output by the link coil which is allocated to the local system by a GL60H, GL70H can be referenced by another GL60H, GL70H, GL60S1, GL60S2, GL60S3 or GL40S at a remote station through a link relay.

(3) Input Relays (1 × × × ×)

An input relay is an ON/OFF signal (discrete input signal) entered through an input module. It may be considered as the contacts of a simulated input coil stored in the CPU module. The contacts of an input coil (input relay) may be used repeatedly in any network. An input relay may be referenced repeatedly but cannot be altered.



(4) Step Relays (S × × ×)

When both the ladder program and the SFC program are being used, the ON/OFF state of the step used in the SFC program can be referenced from the ladder circuit through a step relay.

(5) Stepping Relays (2YY × ×)

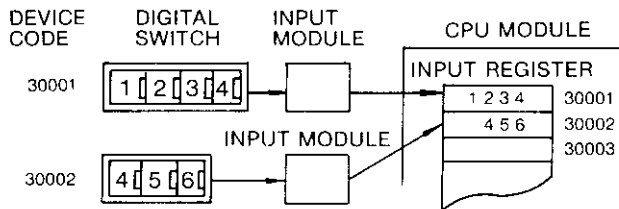
32 Stepping relay contacts (ON, OFF signals) are indicated. According to the contents of the relevant registers (42001 to 42032), a simulated coil is turned on or off.

(6) **M - code Relays** ($M50 \times Y$, $M51 \times Y$), **N - code Relays** ($NOO \times Y$)

These relays operate when the servo interface module IF66 is used. M50 relays are for FIN relays, M51 notch signal relays and NOO is for NEAR relays. They function as input signal contacts from the relevant axes. X (1 to 4) corresponds to the IF66 module number and Y (1 to 8) to the axis number.

(7) **Input Registers** ($30 \times \times \times$)

An input register is memory (16 bits) for temporarily storing a numeric signal sent from an external device such as a digital switch, card reader, A/D converter, or computer. Each input register is identified by a reference number beginning with 30. The reference number may be considered as a device code given to the associated external device.



An input register stores 16-bit numeric data. Binary data input from an external device can be used as is, but BCD data must be converted to binary data by the operation function BIN. If BCD data are used for an internal operation, the result will be incorrect.

Fig. 4.3 shows connection for numeric data input from the digital switch.



The contents of any input register is binary in the memory. It is possible to refer to the contents of any input register but impossible to alter them in the ladder circuit.

(8) **Holding Registers** ($4 \times \times \times \times$)

The holding registers (of 16 bits each) hold the preset values and current values of the timer and counter, constants and results of arithmetic operations, transferred data, data convert matrix data, or other constants needed for processing. Each holding register is identified by a reference number beginning with 4.

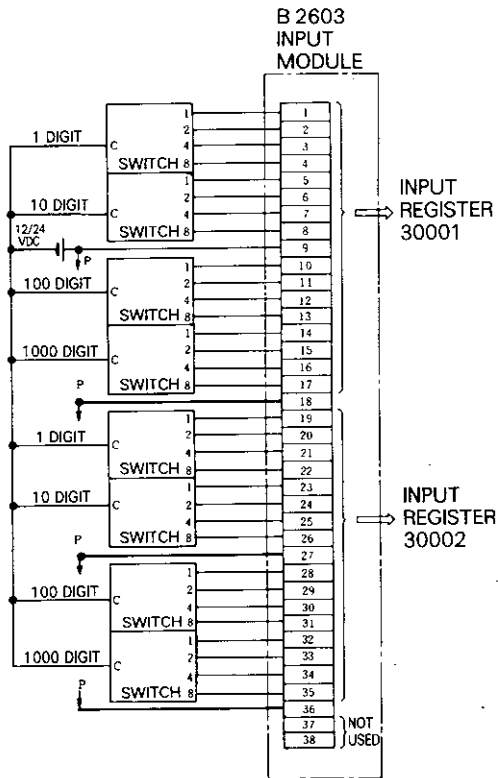
A holding register holds numeric data of 16 bits in binary form.

The contents of the holding register can be output to an external device via an output module if necessary. At this time, the holding register may be called an output register. If the external device to which data are to be output handles BCD data, the output data must be converted from binary to BCD.

Fig. 4.4 shows connection for numeric data output to a BCD digital display device.

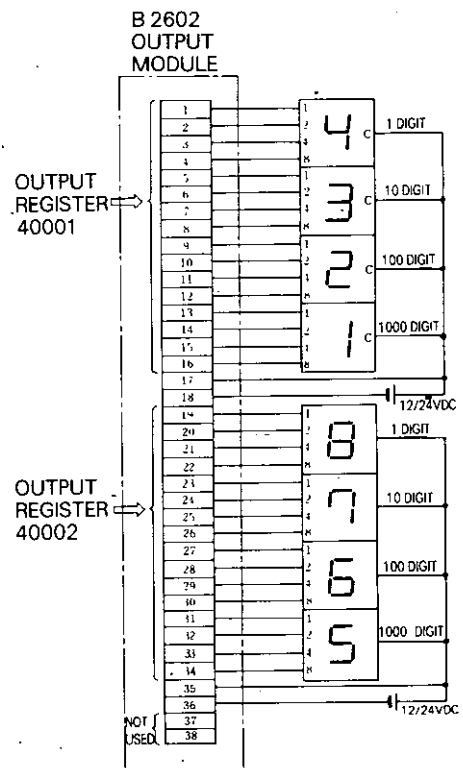


1. The contents of all the holding registers are binary.
2. The contents of the holding registers can be referenced or altered by the ladder program.



NOTE To use for an operation, the data in the input register must be converted to binary.

Fig. 4.3 Sample Connection for Numeric Data Input



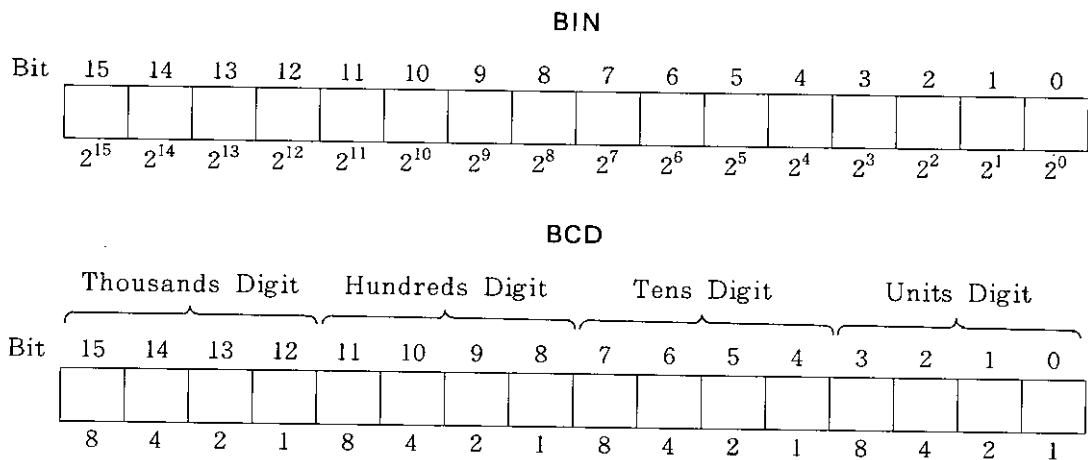
NOTE 1. Data must be converted to BCD before output.
2. The digital display device is assumed to be of 12/24VDC interface, negative logic, and BCD.

Fig. 4.4 Sample Connection for Numeric Data Output

Binary and BCD

Numeric data is represented by "1" and "0", which stand for the ON and OFF states, respectively, in binary notation. The internal data in a computer are generally represented in binary notation. In BCD (abbreviated form of "BINARY CODED DECIMAL") notation, a BCD digit is represented by four bits or binary four digits. Thus a decimal number is represented as a combination of binary four-digit strings. When the GL60S communicates with an external device which uses BCD codes for I/O signals, data conversion is necessary. Data must be converted from BCD to binary when transmitted from an external device to the GL60H/GL70H, and vice versa. The GL60H/GL70H have operation functions BIN (BCD → BIN) and BCD (BIN → BCD) for data conversion.

Fig. 4.5 shows data representation in binary and BCD forms.



Binary and BCD Representations of Decimal Numbers

Decimal	Binary (BIN)	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
2000	0000 0111 1101 0000	0010 0000 0000 0000

Fig. 4.5

(9) Constant Registers (3 × × × ×)

Constant registers (16 bits per register) store constants, such as system initial values, function tables and type code tables, necessary for operation. Each constant register is identified by a five-digit number beginning with 3 (3 × × × ×), and stores binary 16-bit data.

The contents of the constant registers can be referenced by the ladder program but cannot be altered. It is secure even in the case of a power failure.

(10) Timer Registers (5 × × × ×)

The timer registers (16 bits per register) measure time when the steps in an SFC program are active and store the obtained data. Each register is identified by a five-digit number beginning with 5, and stores binary 16-bit data. The timer registers can be used for data transfer and matrix data processing.

The contents of the timer registers can be referenced by the ladder program but cannot be altered.

When an active step becomes inactive, the stored time will be retained until this step becomes active again.

(11) Link Registers (R × × × ×)

The link registers, as well as the link coils, are used when a PC link system is built. Data stored in the link registers which are allocated at the local station by GL60H, GL70H can be referenced by another GL60H, GL70H or GL60S, GL40S at a remote station. That is, the link registers allocated at the local station can be used as "write-enabled registers" and those allocated at a remote station can be used as "read register."

(12) Range of Reference Numbers

Table 4.3 shows classification of reference numbers.

Table 4.3 Range of Reference Numbers

Reference Number	Elements	Remarks
00001 - 04096	Output coils and their contacts	Available as internal coils.
04097 - 08191	Internal coils and their contacts	
08192	Battery monitoring contact	ON when battery voltage is proper.
D0001 - D1024	Link coils and their contacts	
10001 - 14096	Input relays	
S001 - S512	Step relays	
2YY × ×	Stepping relays (YY : 01 to 32, × × : 01 to 99)	
M × × YZ	M - code relays (× × : 50 or 51, Y : IF66 module Nos. 1 to 4, Z : axis Nos. 1 to 8)	
NOOYZ	N - code relays (Y : IF66 module Nos. 1 to 4, Z : axis Nos. 1 to 8)	
30001 - 30512	Input registers	
40001 - 40512	Output registers	Available as holding registers
40513 - 49999	Holding registers	
31001 - 35096	Constant registers	
R0001 - R1024	Link registers	
49998	Stores constant sweep set value	Available as holding registers when not used for constant sweep.
49999	Stores actual scan time	
50001 - 50512	Timer registers	
A0000 - A7FFF	Extended registers	

4.4 GL60H, GL70H INTERNAL PROCESS

Fig. 4.6 shows the GL60H, GL70H internal processing flow chart.

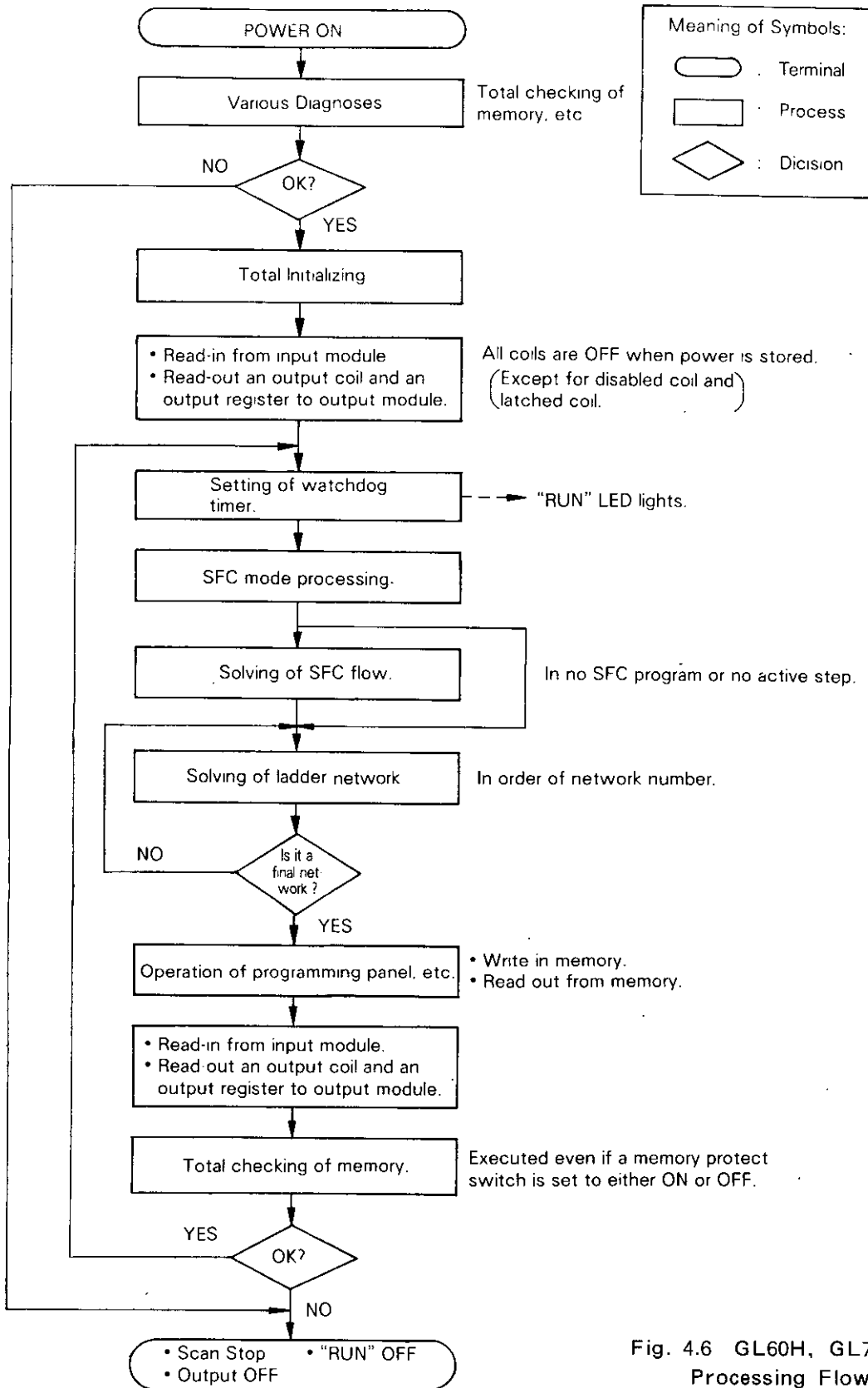


Fig. 4.6 GL60H, GL70H Internal Processing Flow Chart

(1) Power-up Sequence

When power is turned on, the system checks the contents of the memory and, if normal, initializes all input relays and coils, etc.

Table 4.4 shows the initialized statuses of input relays and coils.

Table 4.4 Initialized Statuses of Input Relays and Coils

Elements	Initialized Status
Coils, Link Coils *	OFF except for latched coils and disabled coils
Input Relays, Input Registers, Link Registers †, Link Coils †	Latest status except for disabling input relays
Latched Coils, Disabling Inputs, Disabling Coils	Status held immediately before power failure
Holding Registers, Constant Registers, Link Registers *	Status held immediately before power failure

* Local station

† Remote station

The power-up sequence takes approximately 5 to 10 seconds.

(2) Scan Cycle

Upon completion of a power-up sequence, the GL60H/GL70H perform mode processing. Then, the GL60H/GL70H perform SFC flow solving. After every active step is solved, the GL60H/GL70H solve the ladder circuit except the action circuit and the transition condition circuit. Solving the ladder circuit is performed in order of network numbers beginning at network 1. The processing related to the programming panel and I/O processing are completed after the last network is solved, then the GL60H/GL70H return to SFC mode processing, and repeat the same procedure.

This is called a scan cycle operation, and the time required for one cycle is called scan time.

NOTE The GL60H or GL70H has a two-level scan function. All the ladders are not always scanned during one scan cycle.

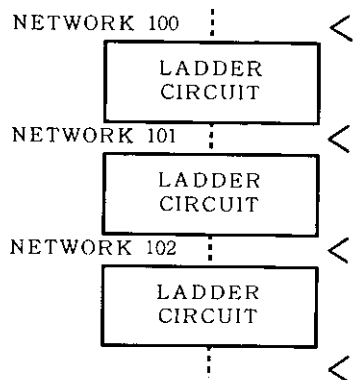
(3) Watchdog Timer

A watchdog timer is set at the beginning of every scanning cycle. The timer remains ON for approximately 500ms after it has been set. If the timer has not been set within a certain time interval, the system stops scanning, determining that there is something wrong with the scanning. At this time, the RUN LED on the CPU module comes OFF and all outputs become OFF. This is one of the self-diagnostic functions.

(4) Network Number

The ladder program is stored in the program memory in units of the network. The number of elements in a network must be in a range of 7 lines \times 10 columns + 1 column (coil). The designer should designate the boundary of the network. The networks are numbered serially (1, 2, 3, and so on). As the designer designates the boundaries of the networks, the system assigns network numbers to them automatically. There are no limitations adding or deleting elements in a network, adding or deleting networks, or inserting a new network between adjacent networks. When some networks are added or deleted, the system controls the network numbers automatically.

NOTE Network numbers may increase until the program memory becomes full.



NOTE The designer designates the boundaries of each network (shown with <). It is actually performed by pressing the START NEXT key on the programming panel.

Fig. 4.7 Networks and Their Delimitations

(5) Checksum

A serious error might occur if some contents of the memory, such as a user program or major system constants have been changed during operation and the changes cannot be detected. To prevent such problems, the system is provided with a self-diagnostic function called "Checksum." The total sum of the contents of the memory which should not be changed has been set. In every scanning cycle, the sum is calculated and compared with the previously set sum. If a discrepancy occurs, the CPU stops scanning as with any error. Simultaneously, all outputs connected to the output module are turned off. Register and analog outputs hold value before stopping the scan.

Total check covers the following.

- Ladder circuits
- SFC programs
- Various allocation table

NOTE

Total check is performed regardless of the memory protect switch position. When in OFF position, some ladder circuits may be altered through the programming panel and it will not be detected as an error by total check. Rather the total check sum is renewed and comparison will be made with reference to this new value.

4.5 SCANNING

The GL60H or GL70H has a two-level scan function which performs high speed and low speed scans at the same time. The two-level scan function reduces the time required for one scan cycle by allocating the elements requiring higher speed control to the high speed segments and the elements not requiring high speed control to the low speed segments.

This subsection describes ladder scanning by the conventional one-scan function first, and then explains the two-level scan function.

4.5.1 Solving a Ladder Circuit

Each network is examined (solved) in order of column one to column ten and then to the coils. Within each column, the logic is solved from the top rung to the bottom rung of that column. The new results from each network (either coil status or register content) are immediately available for use by the next network or column. The scan is done by network number not by output coil number. Fig. 4.8 shows the sequence the GL60S solves networks. The scanning technique is essential to the operation of the GL60S controller and should be understood before proceeding. Table 4.5 shows the status of each element during scanning.

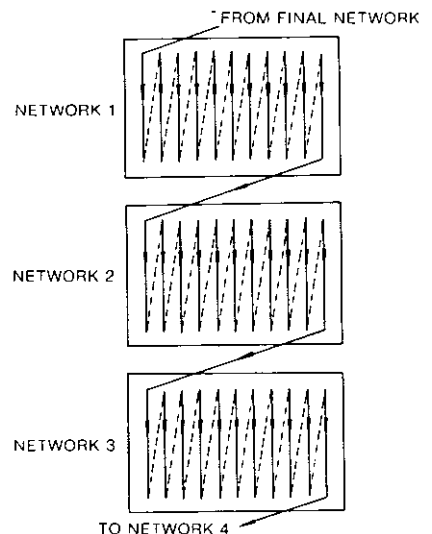


Fig. 4.8 Sequence of Solving Networks

Table 4.5 Status of Each Element during Scanning

Element	Status
Input Relays (Except for Disabling Operation Inputs), Input Registers	Latest status after power-up sequence. Then input status, updated just when inputs are read in, remain unchanged until the next scanning.
Coils (Except for Latch Coils and Disabling Operation Coils)	OFF after power-up sequence. A coil is turned on or off according to the result solved in the first scanning status remains until the next scanning.
Latched Coils, Step Relays	After power-up sequence, status before power failure are restored. During scanning, the same as with coils.
Disabling Operation Inputs, Disabling Operation Coils, Constant Registers	After power-up sequence, status before power failure are restored. Statuses are updated only when changed through the programming panel at the end of scanning.
Holding Registers, Link Registers	After power-up sequence, status before power failure are restored. During scanning, the contents vary according to changing the register contents using a ladder diagram.

NOTE Input and output are actually performed when they are updated, as described in Par. 4.4.

As the CPU solves a ladder circuit by scanning, the ON/OFF status of coils, latched coils or link coils, and the contents of the holding registers or the link registers may change and the new status is referred to, not only in the subsequent networks, but in the same network for the elements which are to be solved later.

In Fig. 4.9, the status of a coil solved in network N does not change between network N + 1 and the final network nor between network 1 and network N-1. But the contents of a holding register or a link register may be changed at any time during a scanning cycle and the revised contents remain held until they are changed again.

For example, a network is composed as shown in Fig. 4.10 (a). This ladder diagram is held in the memory as shown in Fig. 4.10 (b). The GL60S solves the network as shown below where the elements are identified by the reference numbers.

10001 → 10003 → 10002 → 10004 → 00101 (coil) → 00101 (contact) → 00050 (coil).

In this example, a change in coil 00101 will be reflected immediately to line 2.

NOTE If an element has been inserted between the contact 10002 and the coil 00101 on line 1, the change in coil 00101 will be reflected to line 2 during the next scanning cycle but not during the same scanning cycle as that of the change. If such a delay is a problem, remove line 2 from the network and place it in the next one.

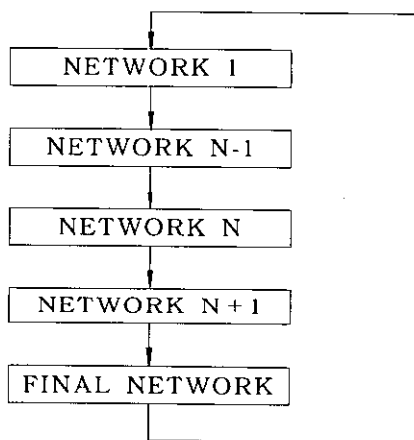
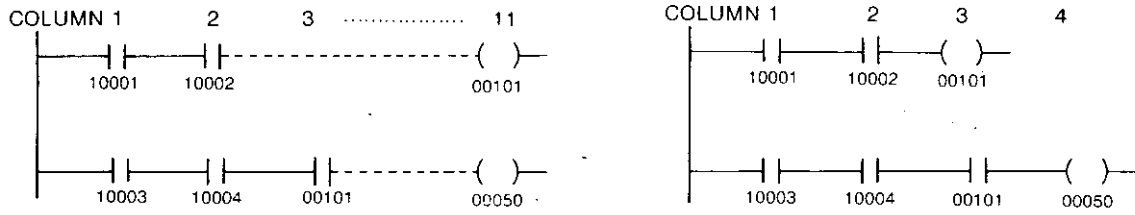


Fig. 4.9 Example of Sequence of Solving Networks



(a) Display on the Screen of the Programming Panel

(b) The Network Stored in the Memory

NOTE Columns 4-10 are not shown above. They are indicated with dotted lines as shown above also on the programming panel.

Fig. 4.10 Example of Sequence of Solving in a Network

4.5.2 Scan Time

(1) Local Station Scan Time

The scan time T (ms) of one scanning cycle is determined roughly by the following formula.

$$T = (\text{Basic Time}) + (\text{Additional Time}) + \{(\text{SFC Solving Processing Time}) + (\text{Network Processing Time}) + (\text{I/O Processing Time})\}$$

NOTE Remote I/O processing time overrides T , where remote I/O processing time is longer than T .

Where

- Basic time = $\frac{0.7 \text{ ms}}{\quad}$ Processing time for self diagnosis
- Additional time = $\frac{2 \text{ ms}}{\quad}$ Processing time for communication module
- Network processing time = $\frac{\{(\text{number of networks}) \times 3 + \sum (\text{processing time of each element})\}}{1000} \text{ (ms)}$
See Table 4.6.
- I/O processing time = $\frac{\{(\frac{\text{number of discrete inputs}}{8} \times 20) + (\frac{\text{number of discrete outputs}}{8} \times 15) + (\text{number of binary register inputs} \times 35) + (\text{number of binary register outputs} \times 28)\}}{1000} \text{ (ms)}$

The processing time of each element varies during execution time and non-execution time as shown in Table 4.6. Therefore, the scan time usually varies. The GL60H or GL70H is provided with a "constant sweep" function which keeps the scan time fixed. This function uses two holding registers as follows.

- Holding register 49998: stores the preset value of constant sweep given in the unit of 10 ms between 10 and 200 ms.
- Holding register 49999: stores the actual scan time when constant sweep is executed. The value varies in the unit of 10 ms.

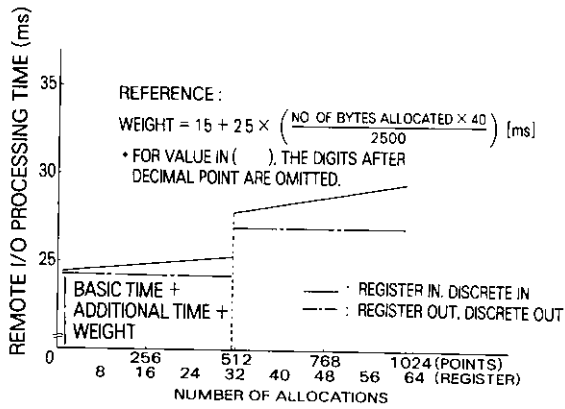
NOTE If the preset value is smaller than the actual scan time, the actual time overrides the preset value.

(2) Remote I/O Processing Time

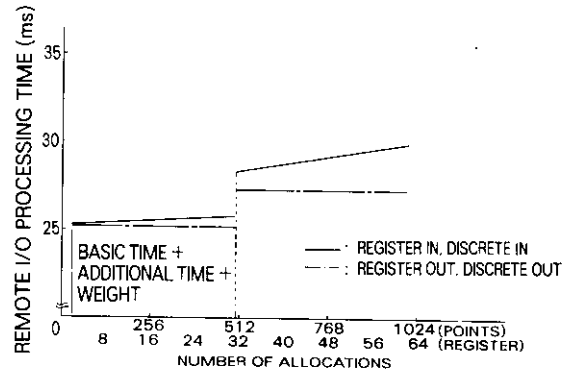
The scan time in a system without a remote station can be calculated using Equation for T described in (1). If a remote station is connected, the remote I/O processing time or T will be the scan time, whichever is longer.

The remote I/O processing time cannot be formulated. The following data can be used as a reference.

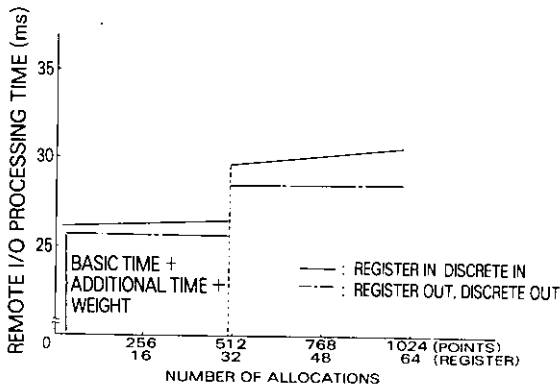
• One-Level Scan



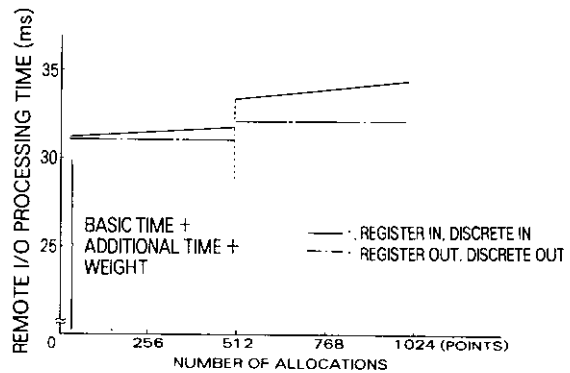
① Number of Stations: 1
 Transmission Speed: 4 Mbps



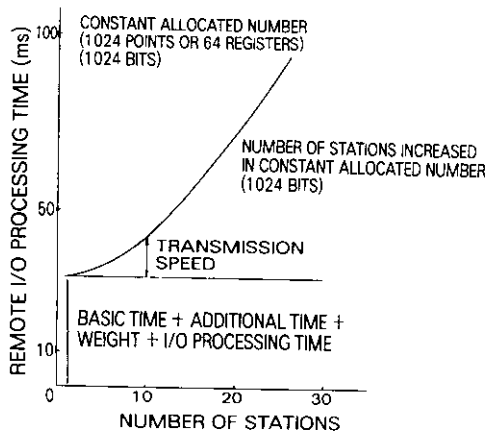
② Number of Stations: 1
 Transmission Speed: 2 Mbps



③ Number of Stations: 1
 Transmission Speed: 1 Mbps

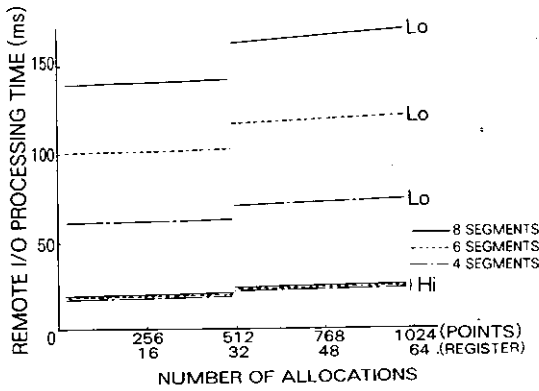


④ Number of Stations: 1
 Transmission Speed: 0.5 Mbps

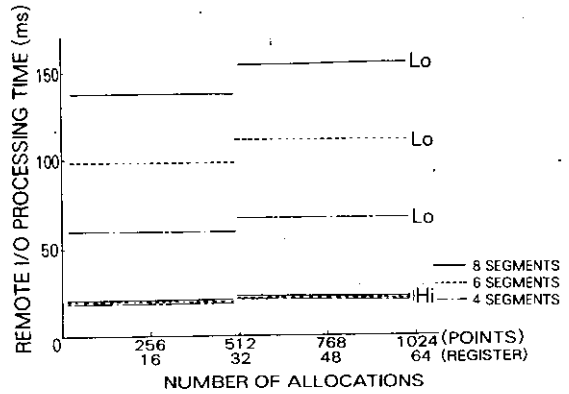


⑤ Number of I/O Allocations: 1024 bits constant

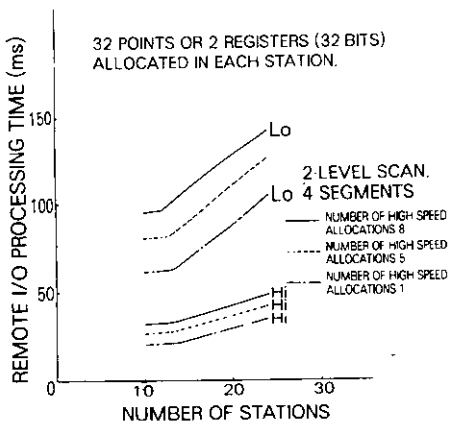
• Two-Level Scan



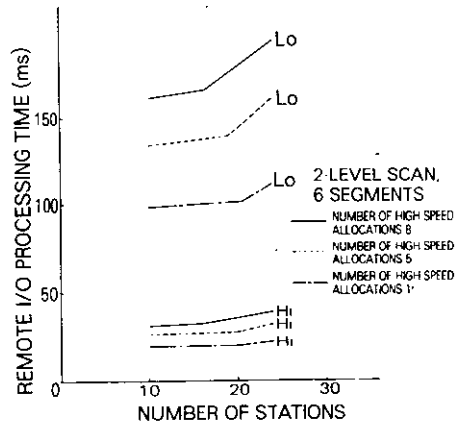
⑥ Number of Stations: 1
Transmission Speed: 4 Mbps
Input Processing



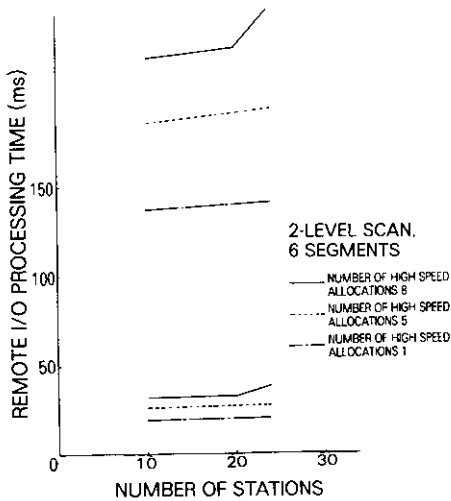
⑦ Number of Stations: 1
Transmission Speed: 2 Mbps
Output Processing



⑧ Number of I/O Allocations: 1024 bits constant
Transmission Speed: 4 Mbps
Number of Segments: 4



⑨ Number of I/O Allocations: 1024 bits constant
Transmission Speed: 4 Mbps
Number of Segments: 6



⑩ Number of I/O Allocations: 1024 bits constant
Transmission Speed: 4 Mbps
Number of Segments: 8

Table 4.6 Processing Time of Elements

Element (Function)	Condition	Processing Time (μ s)		Remarks
		Non Execution	Execution	
Coil, Latched Coil	—	—	1.0	
Contact, Horizontal Open/Shunt	—	—	0.08	
Transitional Contact	—	—	0.3	
Timer	—	14	16	
Counter	—	14	24.5	
Addition	—	16.5	22.5	
Double-precision Addition	—	16.5	30.5	
Subtraction	—	16.5	22.5	
Double-precision Subtraction	—	16.5	33.5	
Multiply	0×0	16.5	22	
	9999×9999		30	
Double-precision Multiply	0×0	16.5	27	
	99999999×99999999		76	
Divide	Quotient overflow	16.5	28	
	For remainder		28	
	For decimal part		32	
Double-precision Divide	Quotient overflow	16.5	20	
	For remainder		224	
	For decimal part		412	
Signed Addition	—	16.5	24	
Signed Double-precision Addition	$(-0) + (+0)$	16.5	35	
	$(-9999) + (-9999)$		34	
Signed Subtraction	—	16.5	25	
Signed Double-precision Subtraction	$(-0) + (+0)$	16.5	34	
	$(-9999) - (-9999)$		36	
Signed Multiply	$(-0) \times (-0)$	16.5	22	
	$(-9999) \times (-9999)$		33	
Signed Divide	Quotient overflow	16.5	24	
	For remainder		30	
	For decimal part		34	
Square Root	$\sqrt{0}$	14	16	
	$\sqrt{9999}$		135	
Double-precision Square Root	$\sqrt{0}$	14	16	
	$\sqrt{99999999}$		214	
Sine	—	14	186	
Cosine	—	14	189	
R-to-T Move	—	16.5	24	
T-to-R Move	—	16.5	26	
T-to-T Move	—	16.5	25	
FIN	—	16.5	$24.5 + 1.15 \times n$	$(1 \leq n \leq 100)$
FOUT	Coil as destination	16.5	26	—
	Register as destination		25	
BLKM	Coil as destination	16.5	$20.5 + 2.3 \times n$	$(1 \leq n \leq 100)$
	Register as destination		$22 + 1.1 \times n$	

4

Table 4.6 Processing Time of Elements (Cont'd)

Element (Function)	Condition	Processing Time (μ s)		Remarks
		Non Execution	Execution	
STAT	Coil as destination	14	$17+1.5 \times n$	$(1 \leq n \leq 100)$
	Register as destination		$16+0.3 \times n$	
SRCH	Compare	16.5	$22.5+1.3 \times n$	$(1 \leq n \leq 100)$
	Non compare			
TSET	—	16.5	$18+0.8 \times n$	$(1 \leq n \leq 100)$
DIBT	—	16.5	$23+1.3 \times n$	$(1 \leq n \leq 100)$
DIBR	—	16.5	$24.5+1.1 \times n$	
SIBT	—	16.5	$24+1.3 \times n$	
SIBR	—	16.5	$22+1.1 \times n$	
BCD→BIN	—	16.5	$20+15 \times n$	$(1 \leq n \leq 100)$
BIN→BCD	—	16.5	$20+10 \times n$	$(1 \leq n \leq 100)$
SWAP	—	16.5	$18+1.5 \times n$	
SORT	—	16.5	$23+12 \times (n-1)$	Max value depending on data
BYSL	—	16.5	$19+2.5 \times n$	$(1 \leq n \leq 100)$
BYCM	—	16.5	$18+2.1 \times n$	$(1 \leq n \leq 100)$
BADD	Word added	16.5	$22+0.8 \times n$	$(1 \leq n \leq 100)$
	Byte added		$23+2.1 \times n$	
AND, OR, XOR	Coil as destination	16.5	$19+1.5 \times n$	$(1 \leq n \leq 100)$
	Register as destination		$20+2.6 \times n$	
COMP	Coil as destination	16.5	$20+2.5 \times n$	$(1 \leq n \leq 100)$
	Register as destination		$19+1.3 \times n$	
CMPR	Miscompare	16.5	$1.8 \times n+3.5 \times m+26$	m: Bit No. in miscompare $(0 \leq m \leq 15)$
	Non miscompare		$26+1.8 \times n$	
MBIT	Coil as destination	16.5	32	—
	Register as destination		28	
SENS	—	16.5	29	—
BROT	Coil as destination (right)	16.5	$26+3.5 \times n$	$(1 \leq n \leq 100)$
	Coil as destination (left)		$25+2.6 \times n$	
	Register as destination (right)		$25+2.3 \times n$	
	Register as destination (left)		$24+1.5 \times n$	
MROT	Shift	16.5	$24+1.5 \times n$	$(1 \leq n \leq 100)$
	Rotate		$24+1.5 \times n$	
TWST	—	16.5	$14+11 \times n$	$(1 \leq n \leq 100)$
BCNT	—	16.5	$21+11.8 \times n$	$(1 \leq n \leq 100)$

NOTE

1. The processing time for a vertical short is zero.
2. n shows table size.
3. The data given above simply provide you with a basis for calculating processing time. It is recommended to measure the actual processing time by the method shown in Fig. 5.27.

4.5.3 Two-Level Scan

When a two-level scan is not specified, the GL60H or GL70H performs a conventional single scan in the following sequence:

[I/O processing → SFC solving → Ladder solving → Processing related]
to the programming panel

When the scan level is set to 2 on the P150 programming panel and the ladder circuit is stored in memory in segment units, the two-level scan is performed. In this case, the segments are divided into two groups, and these two groups are solved with different frequencies. For example, one group is solved three times while the other is solved once. The segment which is solved with a high frequency is called a high speed segment, and the segment which is solved with a low frequency is called a low speed segment.

(1) Segments

When the scan level is set to 2, the ladder circuit, which has been stored in network units, is stored in segment units. (A segment is larger than a network as a unit.) Segment 1 is a high speed segment, and segments 2 to 8 are low speed segments.

Fig. 4.11 illustrates the relationship between networks and segments.

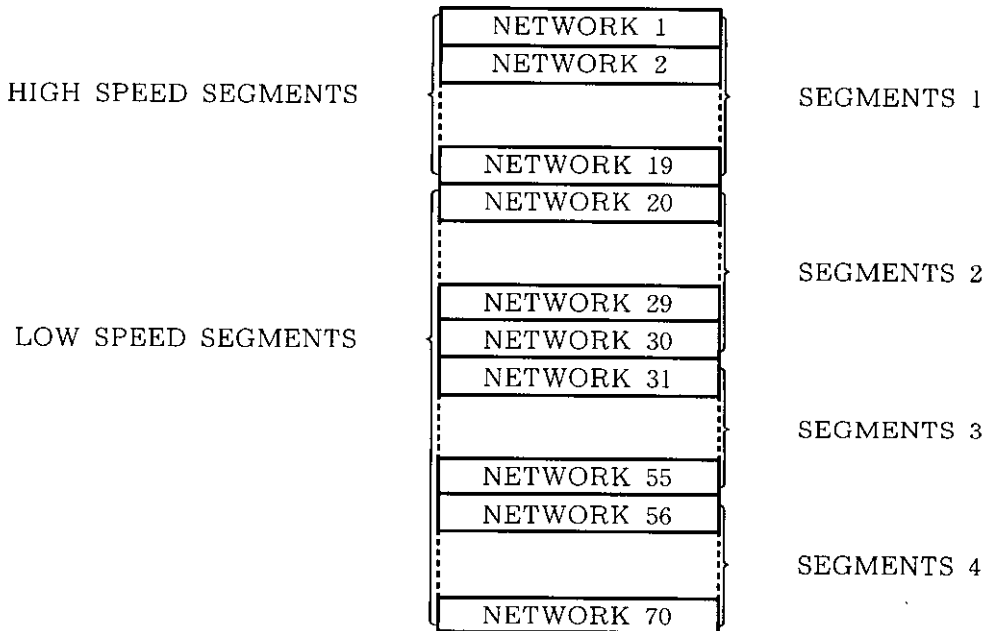


Fig. 4.11 Relationship between Networks and Segments (Using up to Segment 4)

Networks can be allocated to segments when the ladder circuit is stored. After the ladder circuit is stored, the networks within some range can be moved from the current segment to another segment. For the details on segment allocation, refer to the manual of the P150 Programming Panel.

The action circuit and the transition condition circuit of the SFC program cannot be allocated to the low speed segments. The SFC program is solved before segment 1. Thus, only the ladder circuit can be allocated to segments.

(2) Solving by a Two-Level Scan

When networks are allocated to a high speed segment and three low speed segments, the cycle of scanning can be illustrated as below:

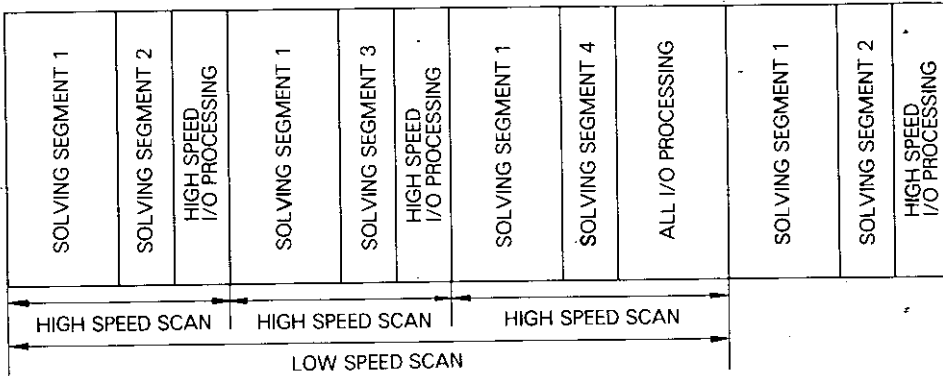


Fig. 4.12 Cycle of a Two-Level Scan

A cycle in which solving of a high speed segment (networks 1 to 19) and a low speed segment and high speed I/O processing are performed is called a high speed scan, while a cycle from the beginning of the first high speed scan to the end of all I/O processing is called a low speed scan. In a low speed scan, all the low speed segments must be solved. Since only one low speed segment is solved per high speed scan, a low speed scan includes several high speed scans.

Accordingly, the ladder circuit in segment 1 (networks 1 to 19) in Fig. 4.11 is solved three times during a low speed scan. On the other hand, segments 2, 3 and 4 (networks 20 to 30, 31 to 55, and 56 to 70), which are low speed segments, are solved only once. This rule also applies to I/O processing. In this example, high speed I/O processing is performed three times while all I/O processing is performed only once at the end.

Fig. 4.12 shows a conventional one-level scan cycle and a two-level scan cycle for comparison.

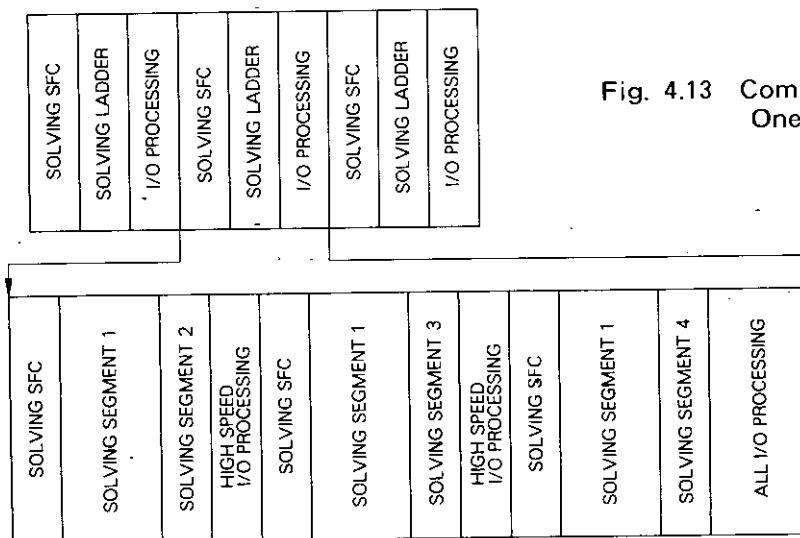


Fig. 4.13 Comparison between One-Level Scan and Two-Level Scan

NOTE

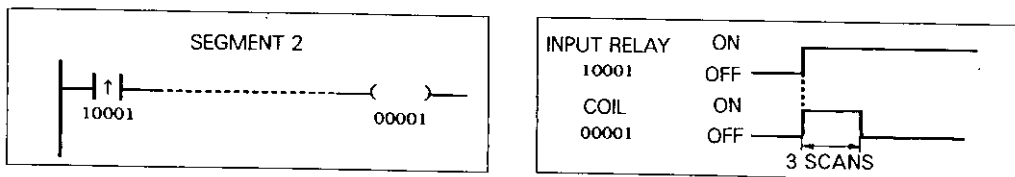
High speed I/O processing means I/O processing performed at a local station and at the stations allocated to the high speed station (see Section 7, I/O Allocation) within a remote station. All I/O processing means the processing performed for all the I/O devices for which I/O numbers are allocated.

The ladder program stored in n (the number of low speed segments) low speed segments are solved only once during each low speed scan. During that time, the high speed scan is performed n times. The maximum number of low speed segments are seven, and if seven low speed segments are used, each of them is solved only once each low speed scan (or the high speed scan is performed n times). Low speed segments are solved in ascending order of the segment numbers. When only one low speed segment is used, solving of segments is performed in the same way as when the entire ladder circuit is stored into a high speed segment. (Using only one low speed segment is useful to control the ladder circuit by dividing it into two blocks.)

The two-level scan is performed to reduce the time required for a high speed scan and provide greater control precision. For this purpose, the ladder circuit for the control that does not require high precision (e.g., ON-OFF control of an indicator lamp) is divided into several low speed segments. The high speed scan is different from the low speed scan in frequency of solving. The two-level scan also reduces α (see Par 4.5.2) which must be added to the scan time when remote I/O processing is performed.

In the case of the following example, note that the differential contact remains ON while a high speed scan is performed three times.

(Example)



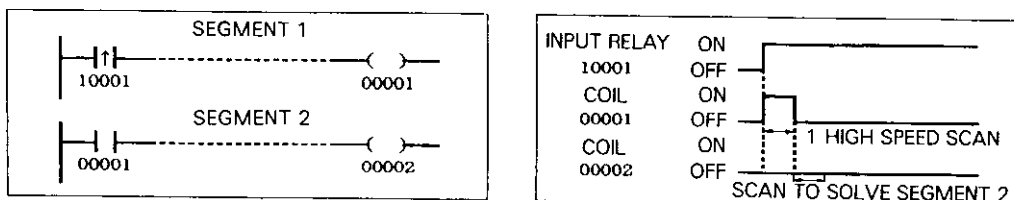
NOTE Assume that three low speed segments are allocated.

Therefore, care must be exercised when the coils within the ladder circuit stored into a low speed segment are to be used in the ladder circuit stored in a high speed segment (e.g., the differential contact must be used again).

On the other hand, when the coils within the ladder circuit stored in a high speed segment are to be referenced in the ladder circuit stored in a low speed segment, the transition of their status cannot be seen correctly since the low speed segment is not solved every time a high speed scan is performed.

In the example below, assume that input relay 10001 is turned ON by a scan. If segment 2 is solved during this scan, coil 00002 is turned ON, but if segment 3 or 4 is solved, coil 00002 is not turned ON.

(Example)



NOTE Assume that three low speed segments are allocated.

When the ladder circuit is not contained in segment 2 but in segments 3 and 4, a high speed scan is performed three times while a low speed scan is performed once.

4.5.3 Two-Level Scan (Cont'd)

According to high speed or low speed scanning of the ladder circuit, inputs and outputs are processed at either a high speed or a low speed. If I/O allocation has been performed but high speed station allocation has not been performed at a remote station, high speed I/O processing is not available. However, only I/O allocation is required for high speed I/O processing at a local station. I/O processing is closely related to solving. For details, refer to Section 7, I/O Allocation.

NOTE Since the watchdog timer for I/O modules is set to 500 ms, the I/O modules go down if the low speed scan time exceeds 500 ms. Accordingly, proper values must be specified for the constant sweep time (effective for a high speed scan) and the high speed scan time so that the low speed scan time may not exceed 500 ms.

4.6 ALLOWABLE NUMBER OF MEMORY WORDS

The following formula gives the memory capacity in words needed for a network.

$$\text{Memory Capacity (in Words)} = 1 + (\text{Number of Columns in Use}) + (\text{Total Number of Elements})$$

As an example, the detail of the network memory capacity is described using the ladder circuit of Fig. 4.14.

- Number of Columns in Use = 9 ... Columns 1-9

NOTE The coils A, B, and C are stored at the locations of *1, *2, and *3 in the memory as shown in Fig. 4.10(b)

- Total Number of Elements = $\frac{7}{\text{Line 1}} + \frac{5}{\text{Line 2}} + \frac{6}{\text{Line 3}} + \frac{4}{\text{Line 4}} = 22$

NOTE On line 2, the locations of *4 and *5 (blank) and the vertical shunt do not occupy any memory. Horizontal shunts are counted as elements. As explained later, the timer and addition are counted as 2 and 3, respectively.

As a result,

- Memory Capacity = $1 + 9 + 22 = 32$ (words)

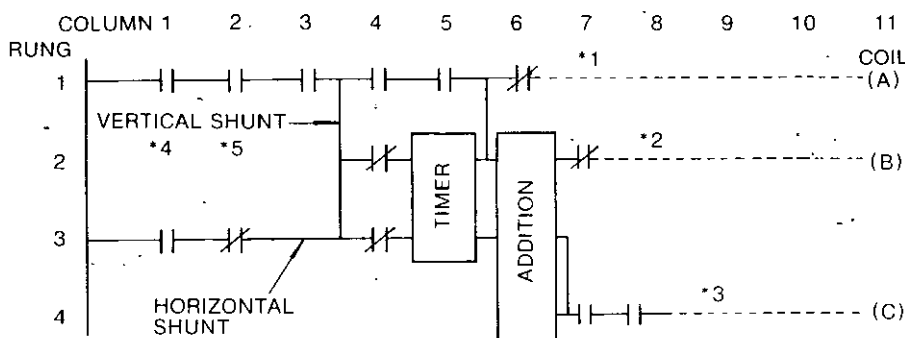
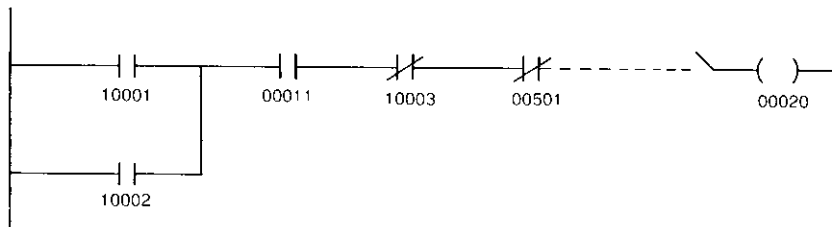


Fig. 4.14 An Example of Ladder Diagram for Calculating Memory Capacity Needed for Network

4.7 DISABLE FUNCTION

To simplify the checkout and maintenance of a control system using the GL60S Controller, a special feature is incorporated into the Controller. This feature is called the Disable function. The Disable status is alterable only if Memory Protect is OFF. Any logic coil selected by the cursor can be disconnected from its logic by depressing the DISABLE pushbutton. If the coil was OFF when the pushbutton was depressed, it will remain OFF; if it was ON, it will remain ON. The coil is no longer controlled by the program in the Controller, but is now controlled by the operator via the P150 Programming Panel or RAP of an I/O processor module. The coil can be toggled ON/OFF/ON/OFF by successively depressing the FORCE pushbutton.

Fig. 4.15 shows an example of a disabled coil display on the screen of the programming panel.



NOTE

The disabled condition remains unchanged even if power is turned off and on.

Fig. 4.15 Sample Display of a Disabled Coil

4.8 TRACE BACK FUNCTION

The GL60H or GL70H has a trace back function in addition to a simulation function to be used when debugging is performed. The trace back function traces the ON/OFF states of the discrete signals (coils and relays) and the transition of the register values within the specified number of scan cycles.

The following parameters must be specified:

- Sampling scan cycle: The number of scans to be performed for a single sampling.
- Trigger point: The number of points to be sampled after the trigger condition is established.
- Discrete: 8 discrete points to be traced and their trigger conditions (ON/OFF or no condition).
- Register: A register to be traced and its trigger condition.

The trace back function traces 1024 points and displays all of their states on the screen.

Fig. 4.16 is a sample of the trace back condition setting screen.

TRACE BACK CONDITIONS			UNIT;001	PROGRAM MODE
SAMPLING ; DISABLE				
SAMPLING CYCLE; 1 SCAN/POINT				
TRIGGER ; 512 POINT				
NO.	DISCRETE REF#	TRIGGER	REGISTER REF#	TRIGGER
1	10001	ON	40035	500
2	S005	ON		
3	00020	*		
4	---	--		
5	---	--		
6	---	--		
7	---	--		
8	---	--		

Fig. 4.16 Trace Back Condition Setting

In the example above, a trigger condition is established when the scan is performed in the following conditions: the states of input relay 10001 and step relay S005 are ON and the contents of holding register 40035 is 500. Then 512 points are traced and their states are displayed. Coil 00020 does not affect the trigger because no condition needs to be set (indicated by "*"). Thus, a trigger condition is established when all the conditions set for the specified reference are satisfied. As the trace back function displays the states of 1024 points, the states of 512 points before the establishment of the trigger condition are displayed on the same screen. If a trigger condition is established at the point before the 512th point, 0 is displayed for the value before starting tracing.

When the trace back function is executed by specifying the two-level scan, sampling is performed based on the cycle of a high speed scan. For example, if the sampling cycle is set to 2, a single sampling is performed during two high speed scans. Thus, the trace back function is used to display the transition of the status of coil and relays but not used to display the I/O status.

SECTION 5 PROGRAMMING FUNCTIONS

5.1 PROGRAMMING FUNCTION LIST

GL60H or GL70H is provided with the programming functions as shown in Table 5.1.

Table 5.1 Programming Function List

No.	Name	Description	Page																																				
1	Relay	<ul style="list-style-type: none"> For programming a relay circuit using relay elements. There are 10 types of relay elements as follows: <table border="1"> <thead> <tr> <th>No</th> <th>Type</th> <th>Symbol</th> <th>No</th> <th>Type</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Normally Open Contact</td> <td></td> <td>6</td> <td>Vertical Shunt</td> <td> </td> </tr> <tr> <td>2</td> <td>Normally Closed Contact</td> <td></td> <td>7</td> <td>Vertical Open</td> <td>Non</td> </tr> <tr> <td>3</td> <td>Transitional Contact (OFF to ON)</td> <td></td> <td>8</td> <td>Coil</td> <td></td> </tr> <tr> <td>4</td> <td>Transitional Contact (ON to OFF)</td> <td></td> <td>9</td> <td>Latched Coil</td> <td></td> </tr> <tr> <td>5</td> <td>Horizontal Shunt</td> <td></td> <td>10</td> <td>Stepping Coil</td> <td></td> </tr> </tbody> </table>	No	Type	Symbol	No	Type	Symbol	1	Normally Open Contact		6	Vertical Shunt		2	Normally Closed Contact		7	Vertical Open	Non	3	Transitional Contact (OFF to ON)		8	Coil		4	Transitional Contact (ON to OFF)		9	Latched Coil		5	Horizontal Shunt		10	Stepping Coil		51
		No	Type	Symbol	No	Type	Symbol																																
		1	Normally Open Contact		6	Vertical Shunt																																	
		2	Normally Closed Contact		7	Vertical Open	Non																																
		3	Transitional Contact (OFF to ON)		8	Coil																																	
		4	Transitional Contact (ON to OFF)		9	Latched Coil																																	
		5	Horizontal Shunt		10	Stepping Coil																																	
2	Timer	<ul style="list-style-type: none"> For counting a time while any signal is ON There are 3 types of timers as follows: <table border="1"> <thead> <tr> <th>No</th> <th>Type</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Timer (Seconds)</td> <td>T1.0</td> </tr> <tr> <td>2</td> <td>Timer (Tenths of Seconds)</td> <td>T0.1</td> </tr> <tr> <td>3</td> <td>Timer (Hundredths of Seconds)</td> <td>T.01</td> </tr> </tbody> </table>	No	Type	Symbol	1	Timer (Seconds)	T1.0	2	Timer (Tenths of Seconds)	T0.1	3	Timer (Hundredths of Seconds)	T.01	68																								
		No	Type	Symbol																																			
		1	Timer (Seconds)	T1.0																																			
		2	Timer (Tenths of Seconds)	T0.1																																			
3	Timer (Hundredths of Seconds)	T.01																																					
3	Counter	<ul style="list-style-type: none"> For counting the number when any signal is changed from OFF to ON. There are 2 types of counters as follows: <table border="1"> <thead> <tr> <th>No.</th> <th>Type</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Up Counter</td> <td>UCTR</td> </tr> <tr> <td>2</td> <td>Down Counter</td> <td>DCTR</td> </tr> </tbody> </table>	No.	Type	Symbol	1	Up Counter	UCTR	2	Down Counter	DCTR	75																											
		No.	Type	Symbol																																			
		1	Up Counter	UCTR																																			
2	Down Counter	DCTR																																					
4	Arithmetic	<ul style="list-style-type: none"> Arithmetic operation for numerical values in 4-digit decimal form or 8-digit. There are 8 types of arithmetic as follows: <table border="1"> <thead> <tr> <th>No.</th> <th>Type</th> <th>Symbol</th> <th>Functions</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Addition</td> <td>ADD</td> <td>Addition in 4-digit decimal</td> </tr> <tr> <td>2</td> <td>Subtraction</td> <td>SUB</td> <td>Subtraction in 4-digit decimal</td> </tr> <tr> <td>3</td> <td>Multiply</td> <td>MUL</td> <td>Multiply in 4-digit decimal</td> </tr> <tr> <td>4</td> <td>Divide</td> <td>DIV</td> <td>Divide in 4-digit decimal</td> </tr> <tr> <td>5</td> <td>Double-precision Addition</td> <td>DADD</td> <td>Addition in 8-digit decimal</td> </tr> <tr> <td>6</td> <td>Double-precision Subtraction</td> <td>DSUB</td> <td>Subtraction in 8-digit decimal</td> </tr> <tr> <td>7</td> <td>Double-precision Multiply</td> <td>DMUL</td> <td>Multiply in 8-digit decimal</td> </tr> <tr> <td>8</td> <td>Double-precision Divide</td> <td>DDIV</td> <td>Divide in 8-digit decimal</td> </tr> </tbody> </table>	No.	Type	Symbol	Functions	1	Addition	ADD	Addition in 4-digit decimal	2	Subtraction	SUB	Subtraction in 4-digit decimal	3	Multiply	MUL	Multiply in 4-digit decimal	4	Divide	DIV	Divide in 4-digit decimal	5	Double-precision Addition	DADD	Addition in 8-digit decimal	6	Double-precision Subtraction	DSUB	Subtraction in 8-digit decimal	7	Double-precision Multiply	DMUL	Multiply in 8-digit decimal	8	Double-precision Divide	DDIV	Divide in 8-digit decimal	84
		No.	Type	Symbol	Functions																																		
		1	Addition	ADD	Addition in 4-digit decimal																																		
		2	Subtraction	SUB	Subtraction in 4-digit decimal																																		
		3	Multiply	MUL	Multiply in 4-digit decimal																																		
		4	Divide	DIV	Divide in 4-digit decimal																																		
		5	Double-precision Addition	DADD	Addition in 8-digit decimal																																		
		6	Double-precision Subtraction	DSUB	Subtraction in 8-digit decimal																																		
7	Double-precision Multiply	DMUL	Multiply in 8-digit decimal																																				
8	Double-precision Divide	DDIV	Divide in 8-digit decimal																																				
5	Signed Arithmetic	<ul style="list-style-type: none"> Square root operation for numerical values in 4-digit decimal form or 8-digit. There are types of square root as follows: <table border="1"> <thead> <tr> <th>No</th> <th>Type</th> <th>Symbol</th> <th>Functions</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Signed Addition</td> <td>SADD</td> <td>Signed Addition in 4-digit decimal</td> </tr> <tr> <td>2</td> <td>Signed Subtraction</td> <td>SSUB</td> <td>Signed Subtraction in 4-digit decimal</td> </tr> <tr> <td>3</td> <td>Signed Multiply</td> <td>SMUL</td> <td>Signed Multiply in 4-digit decimal</td> </tr> <tr> <td>4</td> <td>Signed Divide</td> <td>SDIV</td> <td>Signed Divide in 4-digit decimal</td> </tr> <tr> <td>5</td> <td>Signed Double-precision Addition</td> <td>SDAD</td> <td>Signed Addition in 8-digit decimal</td> </tr> <tr> <td>6</td> <td>Signed Double-precision Subtraction</td> <td>SDSB</td> <td>Signed Subtraction in 8-digit decimal</td> </tr> </tbody> </table>	No	Type	Symbol	Functions	1	Signed Addition	SADD	Signed Addition in 4-digit decimal	2	Signed Subtraction	SSUB	Signed Subtraction in 4-digit decimal	3	Signed Multiply	SMUL	Signed Multiply in 4-digit decimal	4	Signed Divide	SDIV	Signed Divide in 4-digit decimal	5	Signed Double-precision Addition	SDAD	Signed Addition in 8-digit decimal	6	Signed Double-precision Subtraction	SDSB	Signed Subtraction in 8-digit decimal	108								
		No	Type	Symbol	Functions																																		
		1	Signed Addition	SADD	Signed Addition in 4-digit decimal																																		
		2	Signed Subtraction	SSUB	Signed Subtraction in 4-digit decimal																																		
		3	Signed Multiply	SMUL	Signed Multiply in 4-digit decimal																																		
		4	Signed Divide	SDIV	Signed Divide in 4-digit decimal																																		
5	Signed Double-precision Addition	SDAD	Signed Addition in 8-digit decimal																																				
6	Signed Double-precision Subtraction	SDSB	Signed Subtraction in 8-digit decimal																																				

5

Table 5.1 Programming Function List (Cont'd)

No.	Name	Description	Page																														
6	Square Root	<ul style="list-style-type: none"> • Square root operation for numerical values in 4-decimal form or 8-digit. • There are two types of square root as follows: 	127																														
		<table border="1"> <thead> <tr> <th>No.</th> <th>Type</th> <th>Symbol</th> <th>Functions</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Square Root</td> <td>SQRT</td> <td>In 4-digit decimal</td> </tr> <tr> <td>2</td> <td>Double-precision Square Root</td> <td>DSQR</td> <td>In 8-digit decimal</td> </tr> </tbody> </table>		No.	Type	Symbol	Functions	1	Square Root	SQRT	In 4-digit decimal	2	Double-precision Square Root	DSQR	In 8-digit decimal																		
		No.		Type	Symbol	Functions																											
1	Square Root	SQRT	In 4-digit decimal																														
2	Double-precision Square Root	DSQR	In 8-digit decimal																														
7	Trigonometric Function	<ul style="list-style-type: none"> • Trigonometric function operation for numeric data of up to four decimal places within 360 degrees. • There are two types of trigonometric function operations. 	132																														
		<table border="1"> <thead> <tr> <th>No.</th> <th>Type</th> <th>Symbol</th> <th>Functions</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Sine</td> <td>SIN</td> <td>Sine operation for numeric data within 360°</td> </tr> <tr> <td>2</td> <td>Cosine</td> <td>COS</td> <td>Cosine operation for numeric data within 360°</td> </tr> </tbody> </table>		No.	Type	Symbol	Functions	1	Sine	SIN	Sine operation for numeric data within 360°	2	Cosine	COS	Cosine operation for numeric data within 360°																		
		No.		Type	Symbol	Functions																											
1	Sine	SIN	Sine operation for numeric data within 360°																														
2	Cosine	COS	Cosine operation for numeric data within 360°																														
8	Move	<ul style="list-style-type: none"> • A variety types of data move in multi-data groups. • There are 9 types of data move as follows: 	137																														
		<table border="1"> <thead> <tr> <th>No.</th> <th>Type</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Register-to-Table</td> <td>R→T</td> </tr> <tr> <td>2</td> <td>Table-to-Register</td> <td>T→R</td> </tr> <tr> <td>3</td> <td>Table-to-Table</td> <td>T→T</td> </tr> <tr> <td>4</td> <td>Block Move</td> <td>BLKM</td> </tr> <tr> <td>5</td> <td>First In</td> <td>FIN</td> </tr> <tr> <td>6</td> <td>First Out</td> <td>FOUT</td> </tr> <tr> <td>7</td> <td>Table Search</td> <td>SRCH</td> </tr> <tr> <td>8</td> <td>Get Controller System Status</td> <td>STAT</td> </tr> <tr> <td>9</td> <td>Table Set</td> <td>TSET</td> </tr> </tbody> </table>		No.	Type	Symbol	1	Register-to-Table	R→T	2	Table-to-Register	T→R	3	Table-to-Table	T→T	4	Block Move	BLKM	5	First In	FIN	6	First Out	FOUT	7	Table Search	SRCH	8	Get Controller System Status	STAT	9	Table Set	TSET
		No.		Type	Symbol																												
		1		Register-to-Table	R→T																												
		2		Table-to-Register	T→R																												
		3		Table-to-Table	T→T																												
		4		Block Move	BLKM																												
		5		First In	FIN																												
		6		First Out	FOUT																												
		7		Table Search	SRCH																												
8	Get Controller System Status	STAT																															
9	Table Set	TSET																															
9	Block Move with Index	<ul style="list-style-type: none"> • For specifying a data group transferred by an index, or a destination. • There are 4 types of block move with index as follows: 	167																														
		<table border="1"> <thead> <tr> <th>No</th> <th>Type</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Block Move 1 with Destination Index</td> <td>DIBT</td> </tr> <tr> <td>2</td> <td>Block Move 2 with Destination Index</td> <td>DIBR</td> </tr> <tr> <td>3</td> <td>Block Move 1 with Source Index</td> <td>SIBT</td> </tr> <tr> <td>4</td> <td>Block Move 2 with Source Index</td> <td>SIBR</td> </tr> </tbody> </table>		No	Type	Symbol	1	Block Move 1 with Destination Index	DIBT	2	Block Move 2 with Destination Index	DIBR	3	Block Move 1 with Source Index	SIBT	4	Block Move 2 with Source Index	SIBR															
		No		Type	Symbol																												
		1		Block Move 1 with Destination Index	DIBT																												
		2		Block Move 2 with Destination Index	DIBR																												
3	Block Move 1 with Source Index	SIBT																															
4	Block Move 2 with Source Index	SIBR																															
10	Data Conversion	<ul style="list-style-type: none"> • For changing data pattern within the matrix. • There are 7 types of data conversion 	183																														
		<table border="1"> <thead> <tr> <th>No.</th> <th>Type</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>BCD→BIN</td> <td>BIN</td> </tr> <tr> <td>2</td> <td>BIN→BCD</td> <td>BCD</td> </tr> <tr> <td>3</td> <td>Change Upper Byte and Lower Byte</td> <td>SWAP</td> </tr> <tr> <td>4</td> <td>Rearrange in the Lifting order</td> <td>SORT</td> </tr> <tr> <td>5</td> <td>Resolve Word Data Into Byte Data</td> <td>BYSL</td> </tr> <tr> <td>6</td> <td>Combine Byte Data Into Word Data</td> <td>BYCM</td> </tr> <tr> <td>7</td> <td>Block Addition</td> <td>BADD</td> </tr> </tbody> </table>		No.	Type	Symbol	1	BCD→BIN	BIN	2	BIN→BCD	BCD	3	Change Upper Byte and Lower Byte	SWAP	4	Rearrange in the Lifting order	SORT	5	Resolve Word Data Into Byte Data	BYSL	6	Combine Byte Data Into Word Data	BYCM	7	Block Addition	BADD						
		No.		Type	Symbol																												
		1		BCD→BIN	BIN																												
		2		BIN→BCD	BCD																												
		3		Change Upper Byte and Lower Byte	SWAP																												
		4		Rearrange in the Lifting order	SORT																												
		5		Resolve Word Data Into Byte Data	BYSL																												
6	Combine Byte Data Into Word Data	BYCM																															
7	Block Addition	BADD																															

Table 5.1 Programming Function List (Cont'd)

No.	Name	Description	Page																																				
11	Matrix	<ul style="list-style-type: none"> For operating upon bit patterns within the matrix. There are 10 types of matrices as follows: <table border="1"> <thead> <tr> <th>No</th> <th>Type</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Logical AND</td> <td>AND</td> </tr> <tr> <td>2</td> <td>Logical OR</td> <td>OR</td> </tr> <tr> <td>3</td> <td>Logical Exclusive OR</td> <td>XOR</td> </tr> <tr> <td>4</td> <td>Logical Complement</td> <td>COMP</td> </tr> <tr> <td>5</td> <td>Logical Compare</td> <td>CMPR</td> </tr> <tr> <td>6</td> <td>Logical Bit Modify</td> <td>MBIT</td> </tr> <tr> <td>7</td> <td>Logical Bit Sense</td> <td>SENS</td> </tr> <tr> <td>8</td> <td>Logical Bit Rotate</td> <td>BROT</td> </tr> <tr> <td>9</td> <td>Logical Multi-Bit Rotate</td> <td>MROT</td> </tr> <tr> <td>10</td> <td>Logical Byte Rearrangement</td> <td>TWST</td> </tr> <tr> <td>11</td> <td>Logical Bit Count</td> <td>BCNT</td> </tr> </tbody> </table>	No	Type	Symbol	1	Logical AND	AND	2	Logical OR	OR	3	Logical Exclusive OR	XOR	4	Logical Complement	COMP	5	Logical Compare	CMPR	6	Logical Bit Modify	MBIT	7	Logical Bit Sense	SENS	8	Logical Bit Rotate	BROT	9	Logical Multi-Bit Rotate	MROT	10	Logical Byte Rearrangement	TWST	11	Logical Bit Count	BCNT	200
		No	Type	Symbol																																			
		1	Logical AND	AND																																			
		2	Logical OR	OR																																			
		3	Logical Exclusive OR	XOR																																			
		4	Logical Complement	COMP																																			
		5	Logical Compare	CMPR																																			
		6	Logical Bit Modify	MBIT																																			
		7	Logical Bit Sense	SENS																																			
		8	Logical Bit Rotate	BROT																																			
		9	Logical Multi-Bit Rotate	MROT																																			
		10	Logical Byte Rearrangement	TWST																																			
11	Logical Bit Count	BCNT																																					
12	Skip	<ul style="list-style-type: none"> Any networks are skipped and not solved. Symbol: SKP 	227																																				
13	ASCII	<ul style="list-style-type: none"> There are 2 types of operating functions for processing ASCII message as follows: <table border="1"> <thead> <tr> <th>No.</th> <th>Type</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Read Command</td> <td>READ</td> </tr> <tr> <td>2</td> <td>Write Command</td> <td>WRITE</td> </tr> </tbody> </table>	No.	Type	Symbol	1	Read Command	READ	2	Write Command	WRITE	233																											
		No.	Type	Symbol																																			
		1	Read Command	READ																																			
2	Write Command	WRITE																																					
<ul style="list-style-type: none"> For detail information, refer to Memocon-SC U84/U84S ASCII MODULE USER'S MANUAL (SIE-C815-10.5). 																																							
14	Communication	<ul style="list-style-type: none"> Communication can be performed via communication ports IOP and COMM (the GL60S can be used as a master). COMM is used as a communication symbol. For detail information, refer to Memocon-SC GL60S USER'S MANUAL COMM COMMAND (SIE-C815-14.5). 	-																																				
15	Subroutine	<ul style="list-style-type: none"> This function calls up subroutines A symbol consisting of a two-digit subroutine number preceded by G is used to call up a subroutine. 	228																																				
16	COMM Instruction	<ul style="list-style-type: none"> Communication possible via IOP, COMM communication port as the master COMM is used as a communication symbol. For details, refer to the COMM instruction description (SIE-C815-14.5). 	242																																				
17	Extensive Register Access Instruction	<ul style="list-style-type: none"> Writes-in/reads-out the data to /from extensive register. The following 2 instructions are available: <table border="1"> <thead> <tr> <th>No.</th> <th>Name</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>File read</td> <td>FRED</td> </tr> <tr> <td>2</td> <td>File write</td> <td>FWRT</td> </tr> </tbody> </table>	No.	Name	Symbol	1	File read	FRED	2	File write	FWRT	294																											
		No.	Name	Symbol																																			
		1	File read	FRED																																			
2	File write	FWRT																																					

5

5.1 PROGRAMMING FUNCTION LIST

Table 5.1 Programming Function List (Cont'd)

No.	Name	Description	Page																																																
18	YENET	<p>① Operates when IF65 is used.</p> <p>② The following 8 communication instructions are available:</p> <table border="1"> <thead> <tr> <th>No.</th> <th>Name</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Memobus</td> <td>MBUS</td> </tr> <tr> <td>2</td> <td>Selective simultaneous broadcasting</td> <td>PEER</td> </tr> <tr> <td>3</td> <td>All-station simultaneous broadcasting</td> <td>BROD</td> </tr> <tr> <td>4</td> <td>Booking instruction</td> <td>BOOK</td> </tr> <tr> <td>5</td> <td>Poll instruction</td> <td>POLL</td> </tr> <tr> <td>6</td> <td>Node diagnosis instruction</td> <td>DIAG</td> </tr> <tr> <td>7</td> <td>Data sending instruction</td> <td>SND</td> </tr> <tr> <td>8</td> <td>Data receiving instruction</td> <td>RCV</td> </tr> </tbody> </table> <p>③ For details, refer to the YENET-3200 optical LAN system user's manual (SIE-C815-14.16).</p>	No.	Name	Symbol	1	Memobus	MBUS	2	Selective simultaneous broadcasting	PEER	3	All-station simultaneous broadcasting	BROD	4	Booking instruction	BOOK	5	Poll instruction	POLL	6	Node diagnosis instruction	DIAG	7	Data sending instruction	SND	8	Data receiving instruction	RCV	—																					
No.	Name	Symbol																																																	
1	Memobus	MBUS																																																	
2	Selective simultaneous broadcasting	PEER																																																	
3	All-station simultaneous broadcasting	BROD																																																	
4	Booking instruction	BOOK																																																	
5	Poll instruction	POLL																																																	
6	Node diagnosis instruction	DIAG																																																	
7	Data sending instruction	SND																																																	
8	Data receiving instruction	RCV																																																	
19	Motion Instruction	<p>① Motion control instruction which operates when IF66 is used</p> <p>② The following 15 control instructions are available:</p> <table border="1"> <thead> <tr> <th>No.</th> <th>Name</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1-axis program run</td> <td>MMOVI</td> </tr> <tr> <td>2</td> <td>1-axis single-block run</td> <td>MVIS</td> </tr> <tr> <td>3</td> <td>Independent multi-axis program run</td> <td>MOVM</td> </tr> <tr> <td>4</td> <td>3-(2-) axis program run</td> <td>MOVL</td> </tr> <tr> <td>5</td> <td>3-(2-) axis single-block run</td> <td>MOVS</td> </tr> <tr> <td>6</td> <td>1-axis home position return operation</td> <td>ZRN</td> </tr> <tr> <td>7</td> <td>JOG operation</td> <td>JOG</td> </tr> <tr> <td>8</td> <td>Handle operation</td> <td>HNDL</td> </tr> <tr> <td>9</td> <td>Monitor</td> <td>MON</td> </tr> <tr> <td></td> <td>Current value setting</td> <td>POS</td> </tr> <tr> <td></td> <td>Parameter setting</td> <td>PROM</td> </tr> <tr> <td></td> <td>Variable setting</td> <td>VAR</td> </tr> <tr> <td></td> <td>Alarm reset</td> <td>ARES</td> </tr> <tr> <td></td> <td>Servo ON</td> <td>SVON</td> </tr> <tr> <td></td> <td>3-(2-) axis current value adjustment</td> <td>ADJ</td> </tr> </tbody> </table> <p>③ For details, refer to the GL40S motion control user's manual (SIE-C815-15.30).</p>	No.	Name	Symbol	1	1-axis program run	MMOVI	2	1-axis single-block run	MVIS	3	Independent multi-axis program run	MOVM	4	3-(2-) axis program run	MOVL	5	3-(2-) axis single-block run	MOVS	6	1-axis home position return operation	ZRN	7	JOG operation	JOG	8	Handle operation	HNDL	9	Monitor	MON		Current value setting	POS		Parameter setting	PROM		Variable setting	VAR		Alarm reset	ARES		Servo ON	SVON		3-(2-) axis current value adjustment	ADJ	261
No.	Name	Symbol																																																	
1	1-axis program run	MMOVI																																																	
2	1-axis single-block run	MVIS																																																	
3	Independent multi-axis program run	MOVM																																																	
4	3-(2-) axis program run	MOVL																																																	
5	3-(2-) axis single-block run	MOVS																																																	
6	1-axis home position return operation	ZRN																																																	
7	JOG operation	JOG																																																	
8	Handle operation	HNDL																																																	
9	Monitor	MON																																																	
	Current value setting	POS																																																	
	Parameter setting	PROM																																																	
	Variable setting	VAR																																																	
	Alarm reset	ARES																																																	
	Servo ON	SVON																																																	
	3-(2-) axis current value adjustment	ADJ																																																	
20	Set, Reset Instruction (for SFC)	<p>① Set/reset instruction for coil used in SFC operating circuit.</p> <p>② The following 2 instructions are available:</p> <table border="1"> <thead> <tr> <th>No.</th> <th>Name</th> <th>Symbol</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Set function</td> <td>SET</td> </tr> <tr> <td>2</td> <td>Reset function</td> <td>REST</td> </tr> </tbody> </table>	No.	Name	Symbol	1	Set function	SET	2	Reset function	REST	—																																							
No.	Name	Symbol																																																	
1	Set function	SET																																																	
2	Reset function	REST																																																	

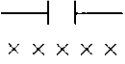
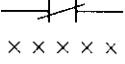

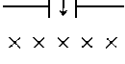




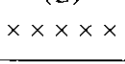
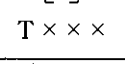
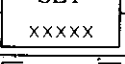
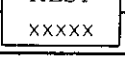
NOTE The reference numbers of coils, relays, and registers in this section are shown assuming that they are used in the ladder circuit. If they are used in the action circuit, the range may be different. To use correct numbers, refer to the reference number list at the end of this manual.

5.2 RELAYS




5.2.1 Relay Logic Elements

Table 5.2 shows relay elements for programming a relay circuit.

Table 5.2 Relay Logic Elements

Element		Symbol (XXXXX: Reference Number)	Description	Reference Number
Contact	Normally Open		Operates while reference coil* is ON.	Coil: 00001 to 08192 [†]
	Normally Closed		Operates while reference coil is OFF.	Input relay: 10001 to 14096
	Transitional Contact (OFF to ON)		Operates only in 1 scan cycle when reference coil is turned on	Step: S001 to S512
	Transitional Contact (ON to OFF)		Operates only in 1 scan cycle when reference coil is turned off	Link coil: D0001 to D1024
Connection	Horizontal Shunt		Shunts between columns.	
	Vertical Shunt		Shunts between lines.	—
	Vertical Open		Opens the vertical shunt.	
Coil	Coil		De-energized when power is restored.	Coil: 00001 to 08191
	Latched Coil		Energized coil when power is restored.	Link coil: D0001 to D1024
	Transition Coil		Transition determining coil of transition condition circuit	Transition coil T001 to T512
Function	Set Function		Function for action circuit coil setting	Coil: 00001 to 08192
	Reset Function		Function for action circuit coil reset	

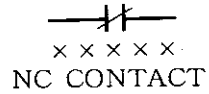
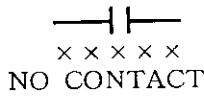
* Meaning of reference coil:

- Reference coil of  is 
0XXXX 0XXXX
- Reference coil of  is input relay 1XXXX.
1XXXX

[†]Coil 8192 cannot be programmed, because it is used as a battery monitoring coil. But its contact can be used in any network.

(1) NO, NC Contacts

① Symbol:



NOTE

x x x x x Shows reference No.

② Function

- NO contact: Operates while reference coil is on, and signals are sent from left to right.
- NC contact: Operates while reference coil is off, and signals are sent from right to left.

NOTE

Reference coils are coil, relay and step which make contact conduct or not conduct.

③ Reference No.

Table 5.3 NO, NC Contacts Reference No.

Reference Coil	Reference No.
Input relay	10001 to 14096
Coil, latched coil* ¹	00001 to 08192
Step* ²	S001 to S512
Link coil	D0001 to D1024
M - code Relays* ³	MXXYZ XX : 50 or 51, Y : 1 to 4, Z : 1 to 8
N - code Relays* ³	N00YZ Y : 1 to 4, Z : 1 to 8

- * 1 : 8192 is a battery coil.
- * 2 : Step is used in SFC program.
- * 3 : Operates when IF66 is used.

④ Operation

Example: COIL

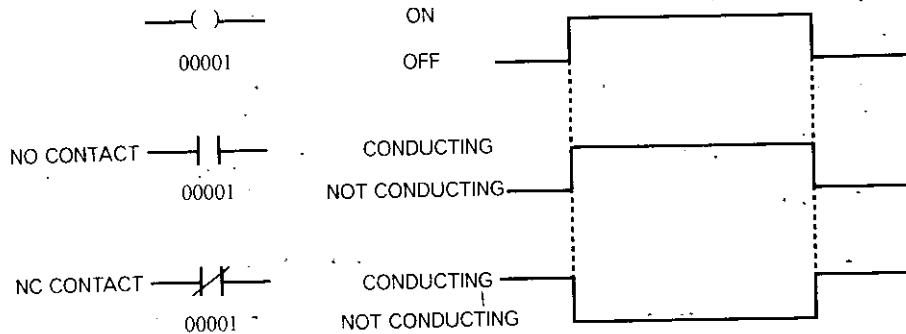
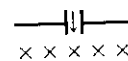
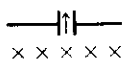


Fig. 5.1 NO, NC Contacts Operation

(2) Transitional Contacts

① Symbol:



NOTE

x x x x x Shows reference No.

② Function

Transitional contact: Operates only in 1 scan cycle when reference coil is turned on, and signals are sent from left to right.

Transitional contact: Operates only in 1 scan cycle when reference coil is turned off, and signals are sent from right to left.

③ Reference No.

Table 5.4 Transitional Contacts Reference No.

Reference Coil	Reference No.
Input relay	10001 to 14096
Coil, latched coil	00001 to 08192
Step	S001 to S512
Link coil	D0001 to D1024
M - code Relays	MXXYZ XX : 50 or 51, Y : 1 to 4, Z : 1 to 8
N - code Relays	NNOYZ Y : 1 to 4, Z : 1 to 8

④ Operation

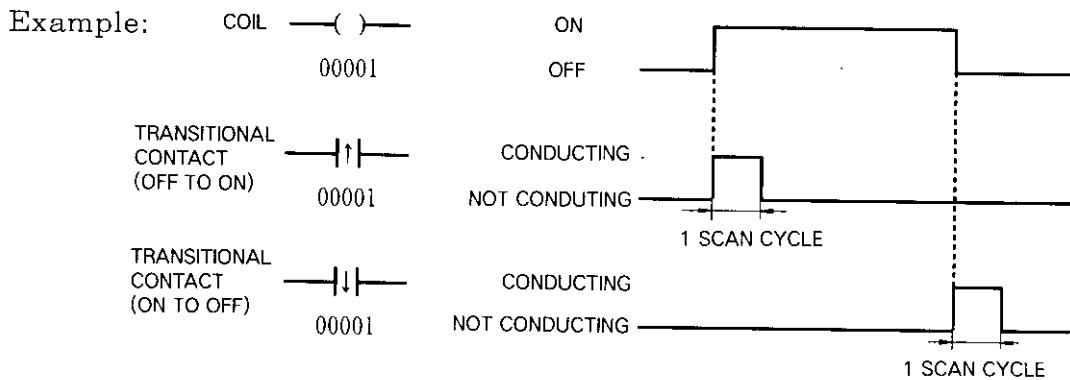


Fig. 5.2 Transitional Operation

(3) Horizontal Shunt

- ① Symbol: _____
- ② Function: Shunts between columns and signals are sent from left to right.
- ③ Reference No.: None
- ④ Application

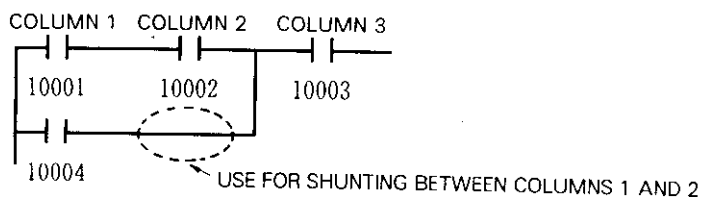


Fig. 5.3 Application for Horizontal Shunt

(4) Horizontal Open

- ① Symbol: . .
- ② Function: Opens horizontal shunt.
- ③ Reference No.: None
- ④ Application

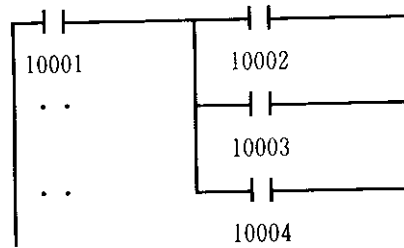


Fig. 5.4 Application for Horizontal Open

(5) Vertical Shunt

- ① Symbol: |
- ② Function: Shunts between columns, and signals are sent from top to down, down to top.
- ③ Reference No.: None
- ④ Application

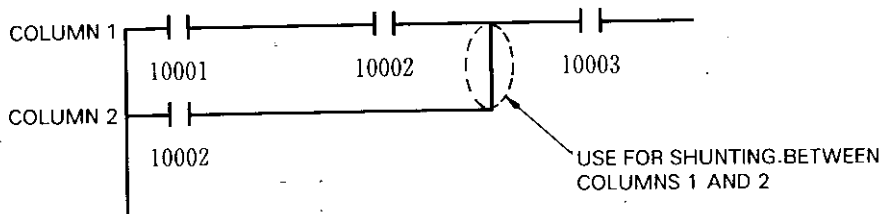


Fig. 5.5 Application for Vertical Shunt

(6) Vertical Open

- ① Symbol: None
- ② Function: Opens vertical shunt.
- ③ Reference No.: None
- ④ Application

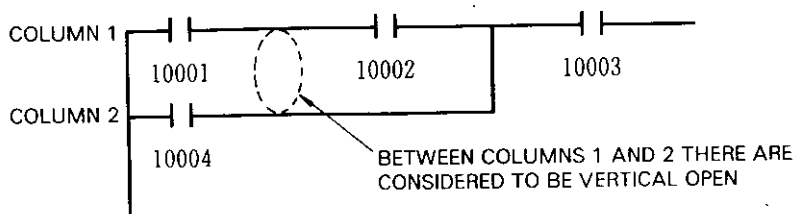


Fig. 5.6 Application for Vertical Open

(7) Coil

① Symbol: $\text{---}(\quad)\text{---}$
 $\quad\quad\quad\times\times\times\times\times$

② Type, function, reference No.

Table 5.5 Coil Type, Function, Reference No.

Type	Function	Reference No.
Output Coil	Outputs ON/OFF condition externally through output module.	00001 to 04096
Internal Coil	Used for assembling logic. Cannot output ON/OFF condition externally.	04097 to 08191
Battery Monitor Coil	Monitors memory back-up battery output voltage.	08192

③ Output coil operation

Example:

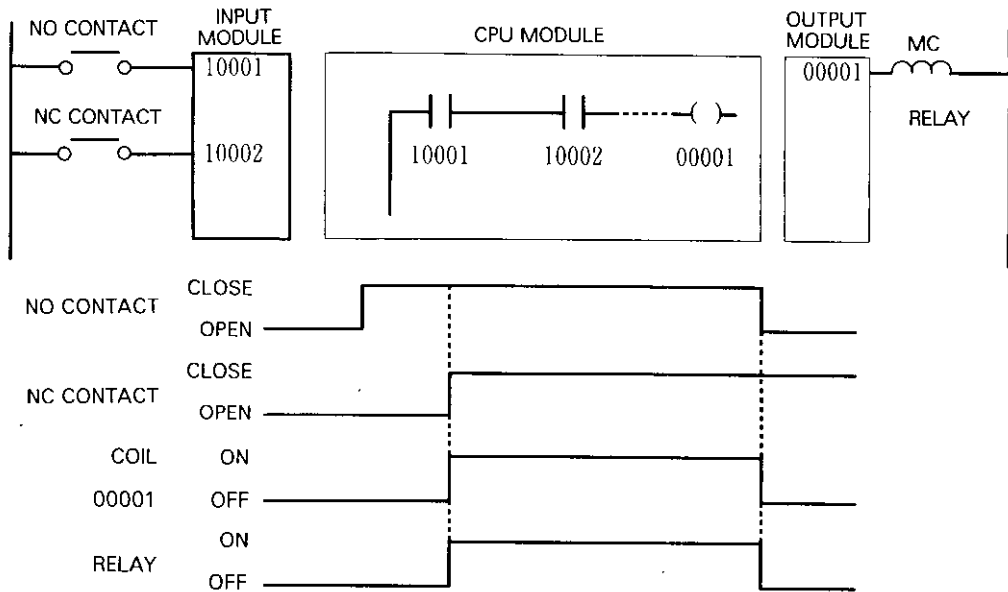


Fig. 5.7 Output Coil Operation

5.2.1 Relay Logic Elements (Cont'd)

④ Internal coil application

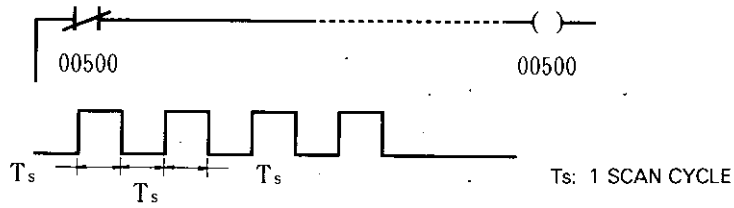


Fig. 5.8 Application for Internal Coil

⑤ Battery monitor coil operation

Example:

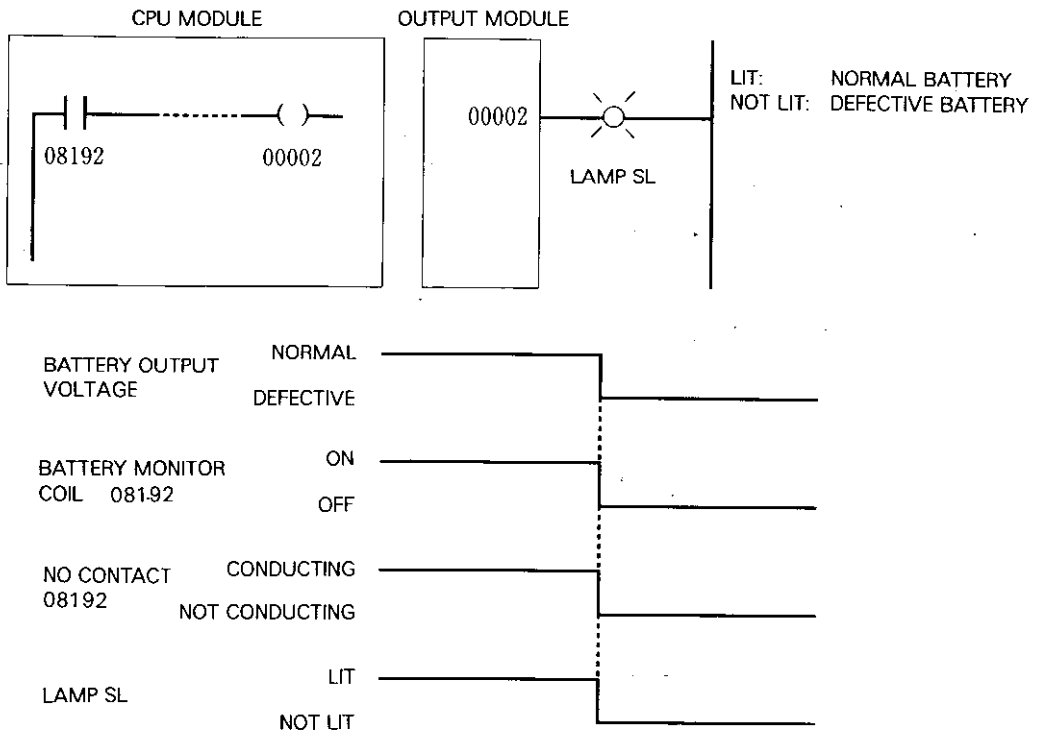


Fig. 5.9 Battery Monitor Coil Operation

(8) Latched Coil

① Symbol: $\text{---} \text{---} (L) \text{---} \text{---}$
 $\times \times \times \times \times$

② Function: The ON/OFF status before power failure will be restored after recovery.

③ Reference No: The same as coil reference No.

④ Operation

Example:

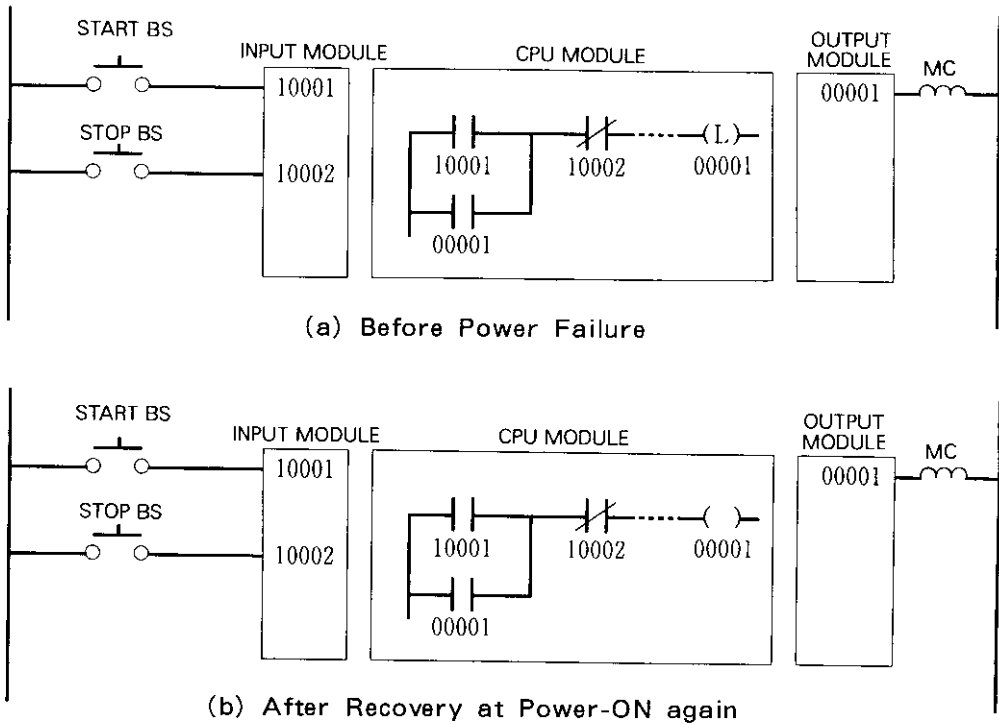


Fig. 5.10 Latched Coil Operation

When —(L)— is on and a power failure occurs;
 00001
 At recovery, without depressing start BS,
 —(L)— is on again.
 00001

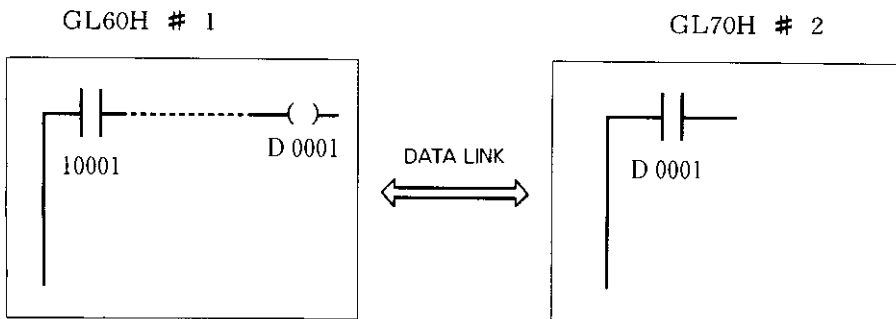
(9) Link Coil

① Symbol: $\text{---}(\quad)\text{---}$
 $D \times \times \times \times$

② Function: When some units of GL60H or GL70H are provided at high speed data link, this coil makes other GL series PC* refer to them.

③ Reference No.: D0001 to D1024

④ Operation



5.2.1 Relay Logic Elements (Cont'd)

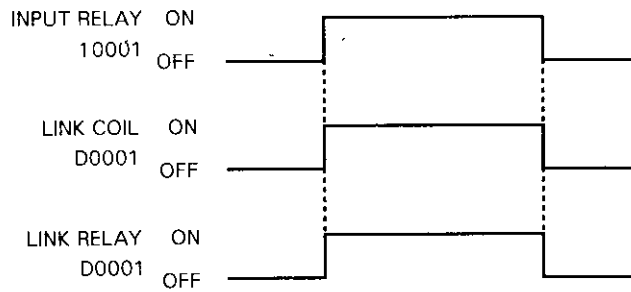
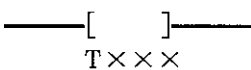


Fig. 5.11 Link Coil Operation

(10) Transition Coil

① Symbol: 

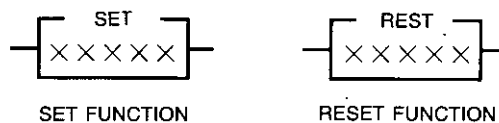
② Function: Operates evolution of steps in SFC program.

③ Reference No.: T001 to T512

④ Operation: Refer to Section 6 "SFC FUNCTION." Used only in SFC program transition condition circuit. Holds ON/OFF status only when the transition condition circuit is solved. Cannot be referred in ladder circuit.

(11) Set and Reset Functions

① Symbol :



② Function : Controls ON/OFF status of the coil that is used in the operation circuit in the SFC program. The set function turns on the specified coil and the reset function turns it off.

③ Reference No. : 00001 to 08191 (same reference can be used twice.)

④ Operation : Refer to Section 6 "SFC FUNCTION ". Used only in the SFC program action circuit. Operates in the SFC program holding mode.

NOTE

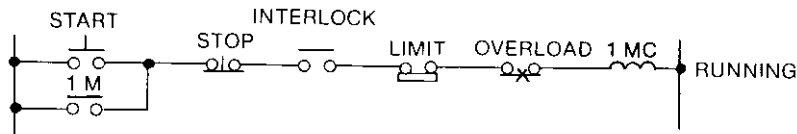
In addition to the GL60H and GL70H, the following G series PCs can be connected by PC link:

GL60SI, GL60S2, GL60S3, GL40S1, GL40S2, GL40S3

5.2.2 Example Relay Logic Circuit

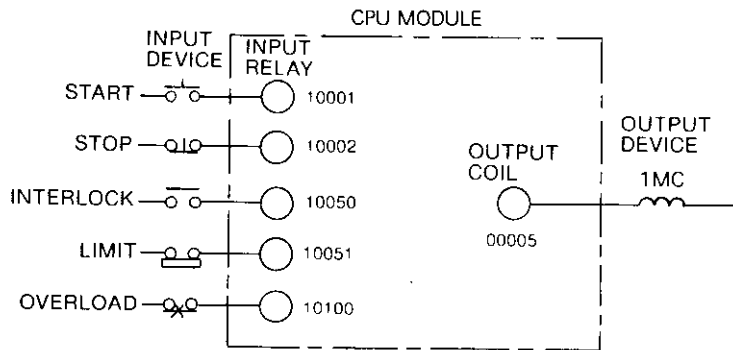
Fig. 5.12 shows an example of relay logic circuit.

- If the logic in Fig. 5.12 (a) is to be implemented in the GL60H or GL70H controller, the control elements must be connected to input circuits in the I/O configuration and outputs assigned.



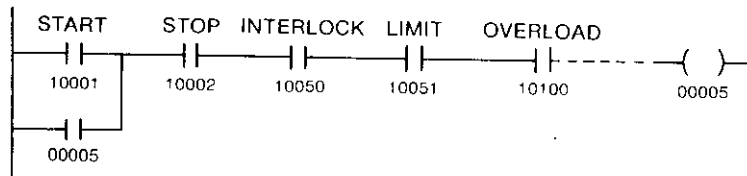
(a) Example Relay Logic

- Fig. (b) illustrates assumed input assignments and wiring details. Output number 00005 is assigned to operate the external device.



(b) Assumed I/O Wiring

- The resultant internal logic to be programmed by the user is shown in Fig. (c).



(c) Equivalent GL60H, GL70H Program

NOTE

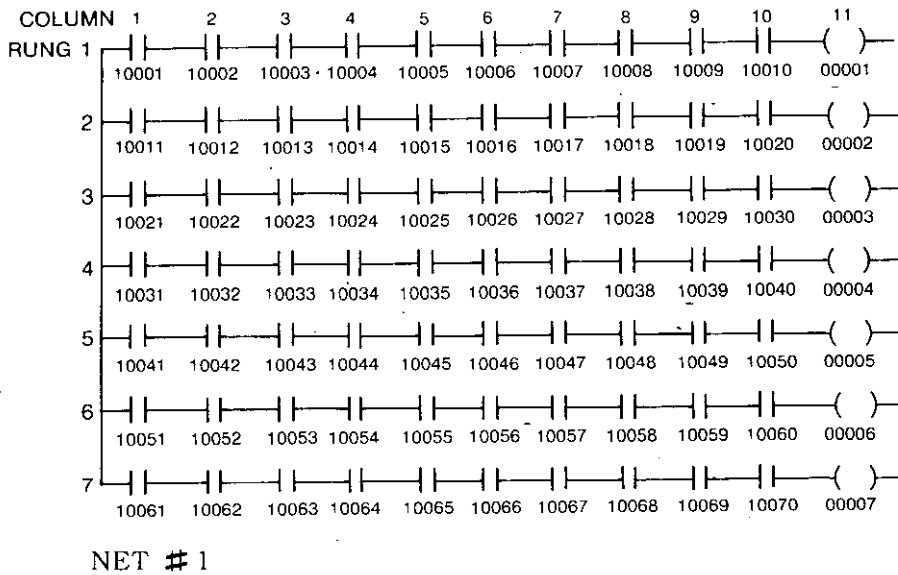
1. In this example, the stop, interlock, limit and overload are normally on for safety from failure. Thus they are normally open contacts in the ladder diagram. Whether a signal is a normally open (NO) or a normally closed (NC) contact should be decided in the design stage.
2. The reference number of a coil is given, like "00005," just as appearing on the screen of the programming panel. On the drawing, "-()" may be written instead of "00005."

Fig. 5.12 Sample Relay Logic Circuit

5.2.3 Creating of Relay Circuits

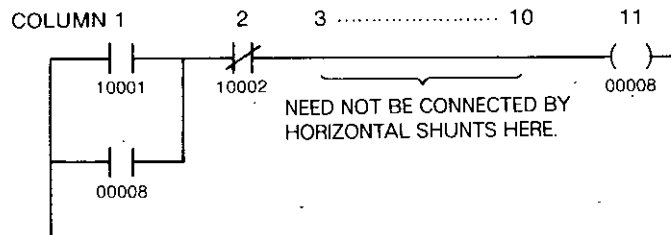
(1) In a network, contacts and horizontal short elements (shunts) may exist at any intersection of the matrix of 7 lines by 10 columns. Coils may exist on column 11. Thus up to 70 contacts and seven coils can be used in a network.

Example:



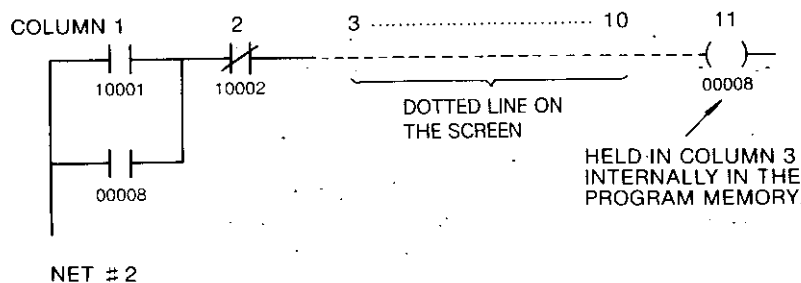
(2) If a relay circuit consisting of contacts and a uniting element ends at column 2 (see below), columns 3 to 10 need not be connected by horizontal shunts before placing a coil in column 11.

Example:



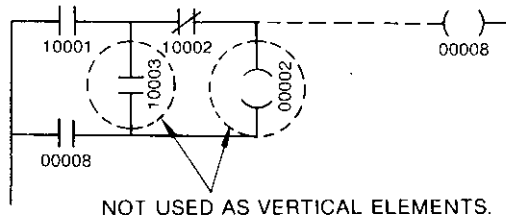
By placing a coil in column 3 on the screen of the programming panel, it will automatically appear in column 11 as shown below.

Example:



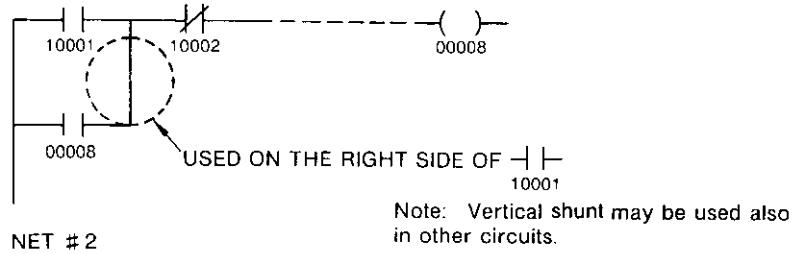
(3) Contacts and coils are used as horizontal elements, but not as vertical elements.

Example:



(4) A vertical shunt may exist on the right side of a contact or horizontal shunt element for downward connection (to the next line), but not on line 7 or with any coil.

Example:

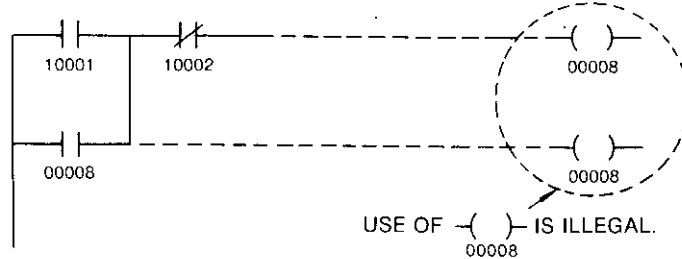


(5) A vertical open is only used to cancel a vertical shunt.

(6) Vertical shunts and opens are not placed at intersections of the 7 lines-by-10 columns matrix and therefore they occupy no memory locations.

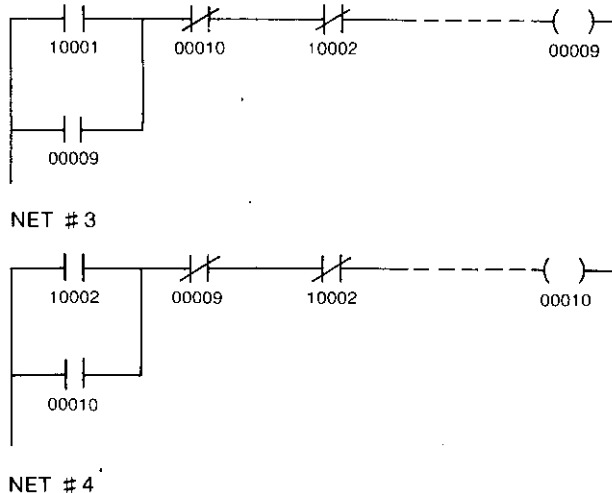
(7) A reference number cannot be given to two or more coils.

Example:



(8) The contact of a coil and input relay (contact) may be used as a NO contact, a NC contact or transitional contact many times.

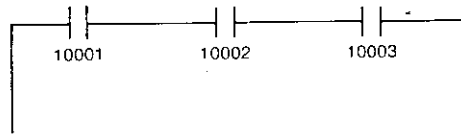
Example:



5.2.3 Creating of Relay Circuits (Cont'd)

(9) A relay circuit without coil is not useful, but not erroneous.

Example:

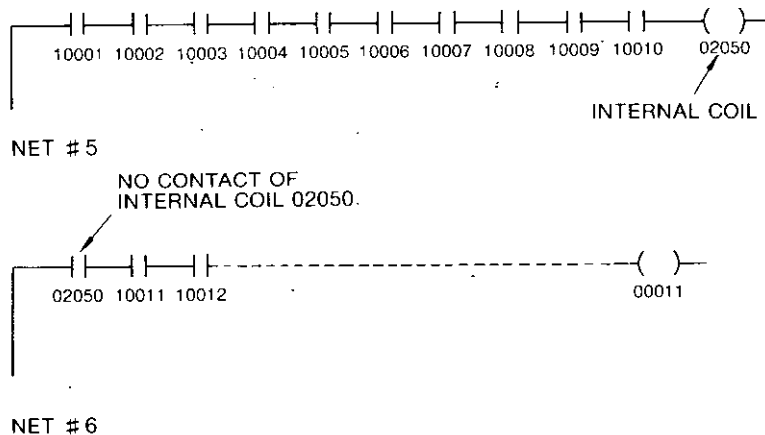


(10) An internal coil is used to extend the number of elements, if more than 10 contacts arranged series or more than seven contacts arranged in parallel are required.

① Examples of series circuit

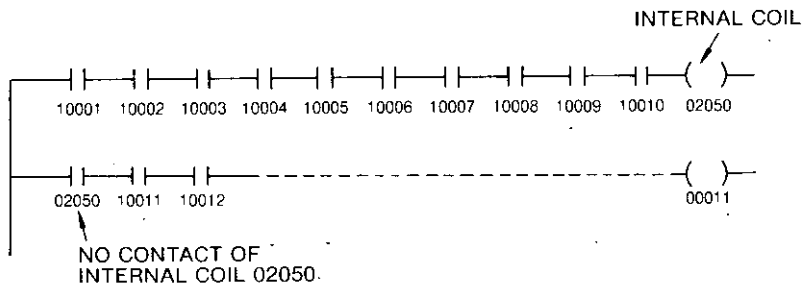
Examples 1 and 2 show how to make up coil 00011 with a series circuit of NO contacts 10001-10012.

Example 1: Where programmed in two networks



Coil 00011 is turned on during the scanning cycle when all input relays 10001-10012 all are turned on.

Example 2: Where programmed in one network

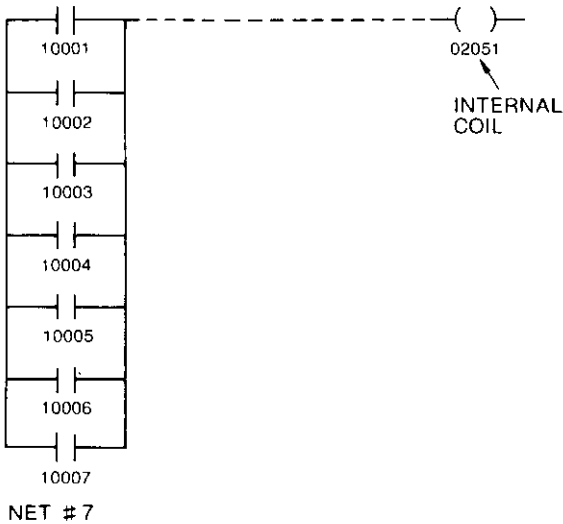


Coil 02050 is turned on during the scanning cycle when all input relays 10001-10012 are turned on, but coil 00011 is turned on during the next scanning cycle (with a delay of one scanning cycle).

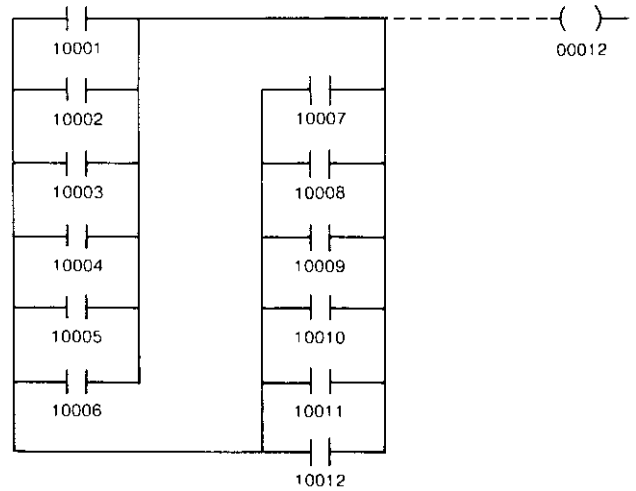
② Examples of Parallel Circuit

Examples 1 and 2 show low to make up coil 00012 with a parallel circuit of NO contacts 10001-10012.

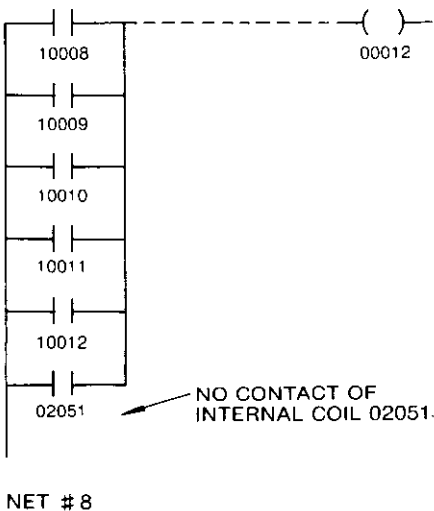
Example 1:



Example 2:

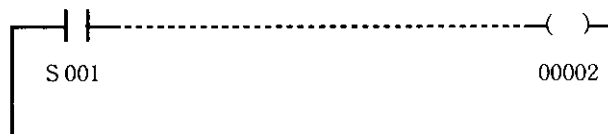


NOTE No internal coil is used in this case.



(11) A step number in the SFC can be specified as the reference number of a relay.

Example:



In the example above, step S001 becomes active when a scan is performed. And the state of coil 00002 turns ON during the same scan. This is caused because the ladder circuit is solved after the SFC.

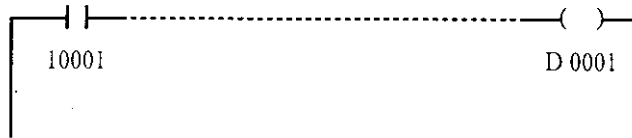
5.2.3 Creating of Relay Circuits (Cont'd)

(12) Link Relay

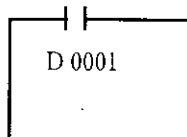
The GL60H or GL70H enables high speed data link by establishing links with other GL60H's or GL70H's. The states of link coils in other GL60H's or GL70H's can be referenced by using link relays.

Example:

GL60H, GL70H # 1



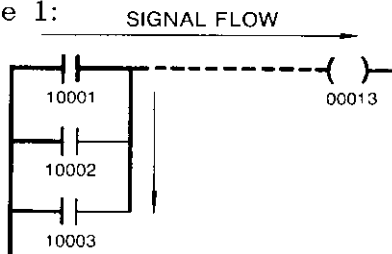
GL60H, GL70H # 2



Synchronized to the link relay D0001 of GL60H, GL70H #1.

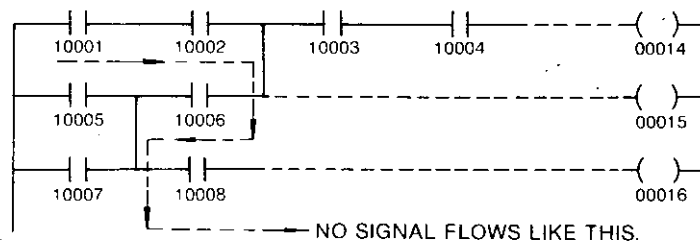
(13) In a relay circuit, a signal always flows from the power rail (left side) to the coils (right side). Vertically it flows either from top to bottom or bottom to top.

Example 1:



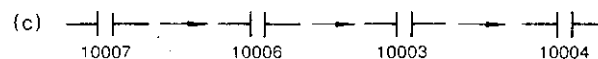
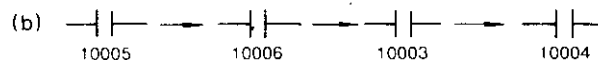
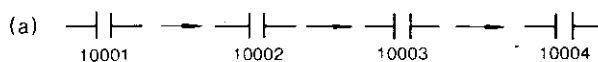
When input relay 10001 is on, a signal flows as shown by arrows and coil 00013 is turned on. On the screen of the PI50 programming panel, flows of signal (called "power flows") are indicated by bold lines as shown here.

Example 2:

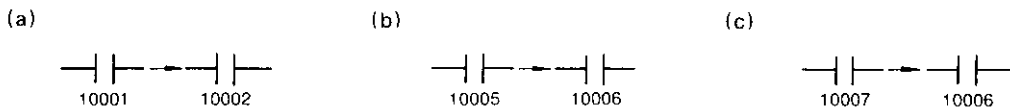


NET # 9

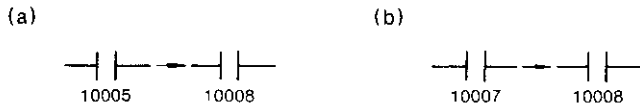
① Paths to turn on coil 00014



② Paths to turn on coil 00015



③ Paths to turn on coil 00016

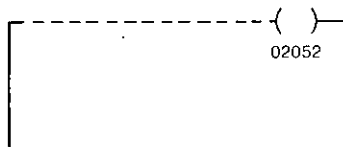


④ Coil 00016 does not turn on through a path as shown with dotted line and therefore it is not necessary to consider a roundabout path.

5.2.4 Sample Application Circuits of Relays

(1) Normally-ON Circuit

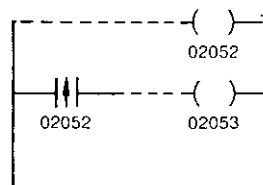
Example:



- Coil 02052 is ON as long as the GL60H, GL70H is operating properly.
- This is used, for example, in a pulse generating circuit for initialization.

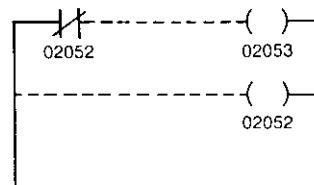
(2) Pulse Generating Circuit for Initialization

Example 1:



- Coil 02053 is turned on only during the first scanning cycle after the GL60H, GL70H power is turned on. Examples 1 and 2 are equivalent.
- These are used to set/reset memory circuits, clear the current values of timers and counters, and preset arithmetic constants for initializing the internal logic after the GL60H, GL70H power is turned on.

Example 2:

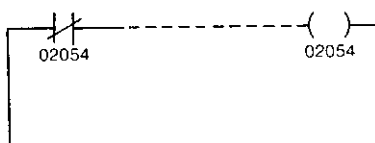


NOTE

Program the circuits of Examples 1 and 2 in a network preceding one using these circuits.

(3) Oscillator

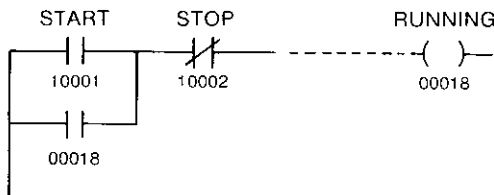
Example:



- Coil 02054 repeats turning on and off every two scanning cycles.
- This is used to perform an arithmetic operation every two scanning cycles.

(4) Self-Holding Circuit (Memory Circuit)

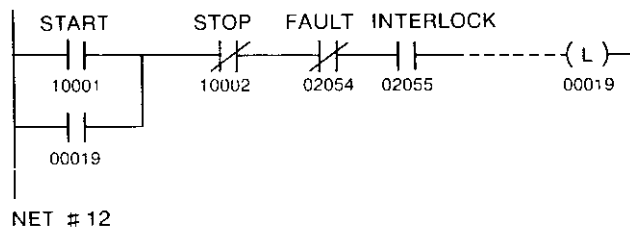
Example:



- As in an ordinary relay circuit, an NO contact of coil is placed in parallel with memory set signal.
- When memory set signal (input relay 10001) comes ON, coil 00018 is turned on and self-held. When memory reset signal (input relay 10002) comes ON, the coil is released from self-holding.
- This is used for start and stop operation with a pushbutton switch which resets automatically.

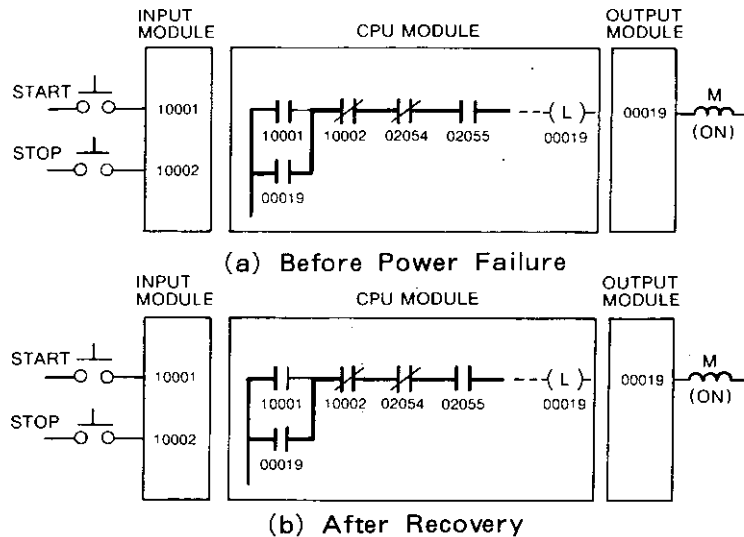
(5) Latched Relay Circuit

(Self-Holding Circuit with Memory during Power Failure)



When a latched coil is used in a self-holding circuit, the status before power failure will be restored after recovery. Any coil can be used as a latched coil.

Example:



NOTE

To use input signal as the reset (OFF) signal (10002 shown above) of a latched relay circuit, read it in as an NO contact then program it as an NC contact (to prevent a latched coil from resetting even if source power of I/O modules stop prior to that of the CPU at power failure).

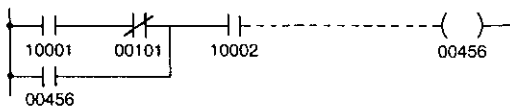
To use a coil as the OFF condition (02054 above) or holding condition (02055 above) of a latched relay circuit, check if it is ON of OFF in reference to the network number during the first scanning cycle after power-on of the GL60H, GL70H.

For example, if the network of latched coil 00019 appears later (has a greater network number) than that of coil 02054 and earlier than that of coil 02055 in the above example, the latched circuit will be reset if coil 02054 is ON during the first scanning cycle after power-on. It will also be reset unless coil 02055 is a latched coil and ON.

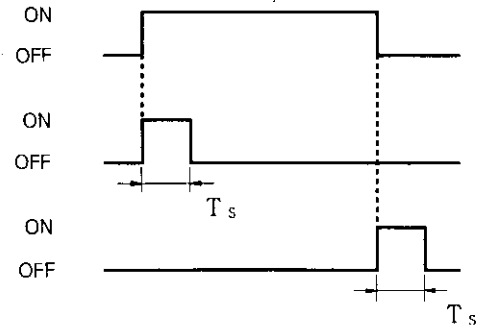
(6) Transitional Contact Circuit

The transitional contact remains on only for one scanning cycle when the associated reference coil has turned on or off. Any input relay or coil may be used as a reference coil. (For 2-level scanning operation, refer to Par. 4.5.3.2.)

NOTE The transitional contact to the associated reference coil in the skipped network is not operated correctly. (Fig. 5.13.) The contact remains ON or OFF.



COIL 00456



T_s : 1 SCAN CYCLE

5.3 TIMERS

5.3.1 Types of Timers

(1) Types

Three types of timers are available as shown in Table 5.6. As many timers as desired used in a range of the program memory capacity and the number of holding registers.

Table 5.6 Types of Timers

Type	Symbol	Unit of Time	Limit of Count
Timer (Seconds)	T1.0	1 sec	1-9,999 sec
Timer (Tenths of Seconds)	T0.1	0.1 sec	0.1-999.9 sec
Timer (Hundredths of Seconds)	T.01	0.01 sec	0.01-99.99 sec

(2) Unit of Time

A timer counts up in certain units of time. The 1-second timer, for example, counts in units of seconds.

(3) Limit of Count

A timer can count up to a certain limit of time. The 1-second timer, for example, counts in the range of 1 to 9,999 seconds. The upper limits of count are presettable.

5.3.2 Timer Configuration

(1) Form

Fig. 5.13 shows the basic form of a timer. A timer is built vertically and needs two elements (top and bottom). Specify any of constant K or reference numbers, referring to Table 5.7. T × × × identifies a specific type of timer. Specify any of T1.0, T0.1, or T.01 referring to Table 5.6.

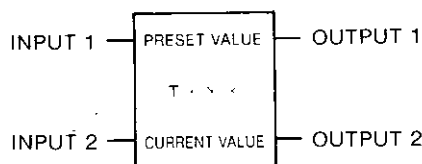


Fig. 5.13 Timer General Form

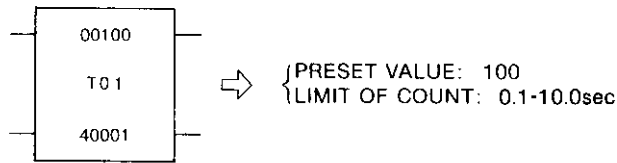
Table 5.7 Timer Elements

Element	Specified Numbers	Description
Top	Constant K (00000-09999) Any of the following: Reference No. Input register (30001-30512) Holding register (40001-49999) Constant register (31001-35096) Link register (R0001-R1024)	The preset value of timer is constant K, or contents of register reference No.
Bottom	Holding register (40001-49999) Link register (R0001-R1024)	The current value of timer is stored in the specified reference No.

(2) Preset Value

Designate a value of 1 to 9999 to determine the range (variable) of timer.

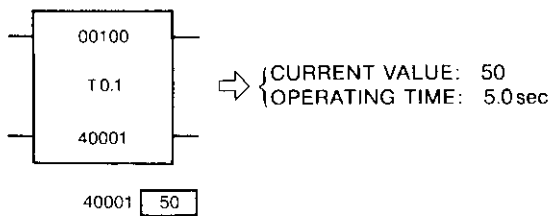
Example:



(3) Current Value

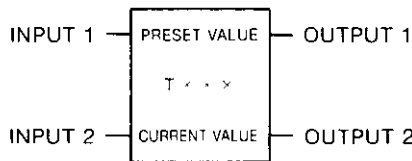
This is the value a timer has counted up.

Example:



5.3.3 Function and Operation of Timer

(1) Timer Function



When input 2 is ON, the timer adds up the time intervals while input 1 is ON. When the current value becomes equal to the preset value, it stops counting with output 1 turned ON and output 2 OFF. When input 2 is OFF, the timer does not count up regardless of the status of input 1. At this time, output 1 is kept OFF and output 2 is ON (the current value is 0).

(2) Timer Operation

Fig. 5.14 and Table 5.8 show the timing chart and operation of the timer.

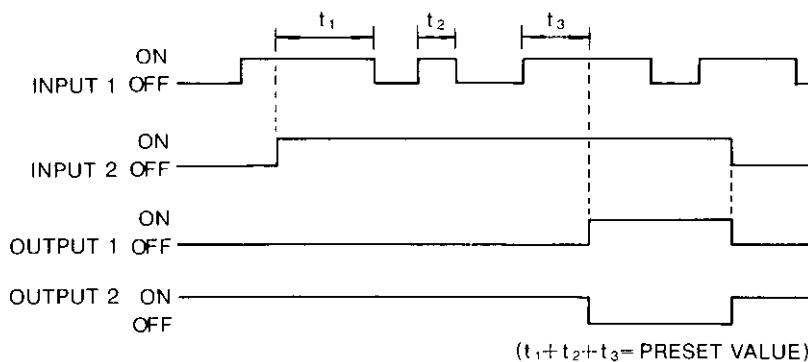


Fig. 5.14 Timer Operation

5.3.3 Function and Operation of Timer (Cont'd)

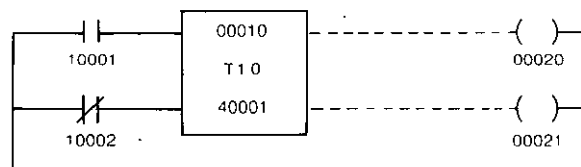
Table 5.8 Timer Operation

Input Status		Timer Status		Current Value	Output Status	
Input 1	Input 2				Output 1	Output 2
-	OFF	Reset status		0	OFF	ON
ON	ON	Operating status	Current value < preset value	Increase	OFF*	ON*
			Current value = Preset value	Preset value	ON	OFF
OFF	ON	Standstill status	Current value < Preset value	Constant	OFF	ON
			Current value = Preset value	Preset value	ON	OFF

*As a result of increasing current value, current value < preset value.

(3) Sample Timer

Example:



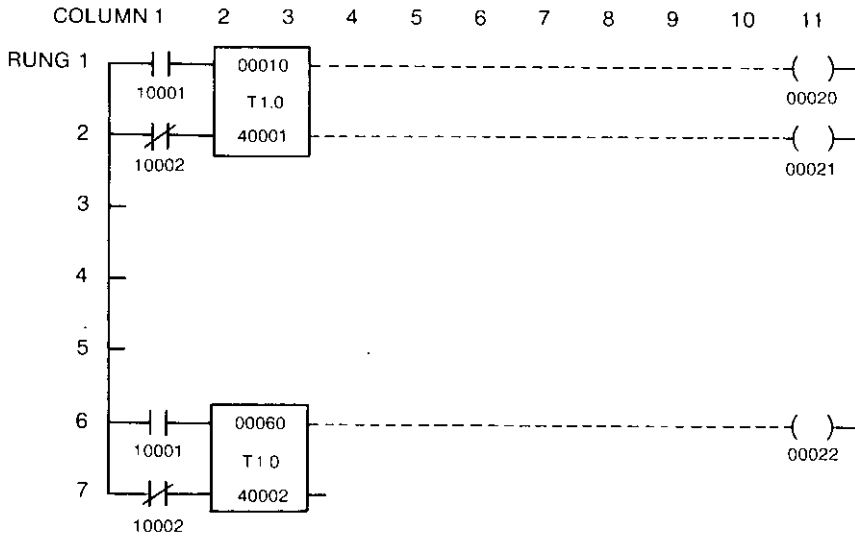
- This timer counts up only when input 2 is ON. (input relay 10002 is OFF).
- If input 1 (input relay 10001) is turned ON while input 2 is ON, the timer is operated and the current value (contents of 40001) is increased by one every second.
- When the current value is equal to the preset value (10), the timer stops and output 1 (coil 00020) is turned ON and output 2 (coil 00021) OFF.
- If input 2 is ON and input 1 is cycled ON-OFF-ON-OFF; the time intervals while input 1 is ON are added. See Fig. 5.13.
- When input 2 is OFF, the timer does not count up regardless of the status of input 1. At this time, the current value is 0, output 1 OFF, and output 2 ON.

5.3.4 Programming Timer Circuit and Precautions

(1) Programming Timer Circuit

A timer needs two elements placed vertically (top and bottom) in a network. It can be used at any intersection of the 7 lines-by-10 columns matrix, but the top element (preset value) cannot be located on line 7.

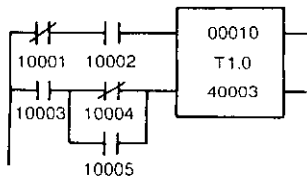
Example:



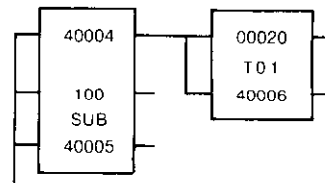
(2) Timer Inputs

Inputs to the timer may be outputs of relays, other timers, counters, arithmetic operations, or data processing circuits.

Example 1:



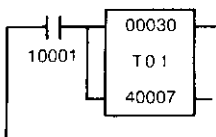
Example 2:



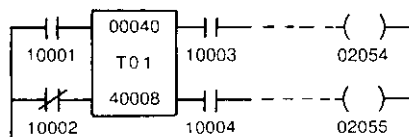
(3) Timer Outputs

Coils may not be connected to outputs 1 and 2 of a timer. It is permitted to insert a relay contact to the right of an output or to connect an output directly to an input of a logic, except relays.

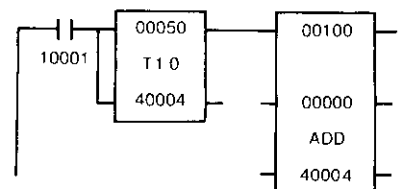
Example 1:



Example 2:



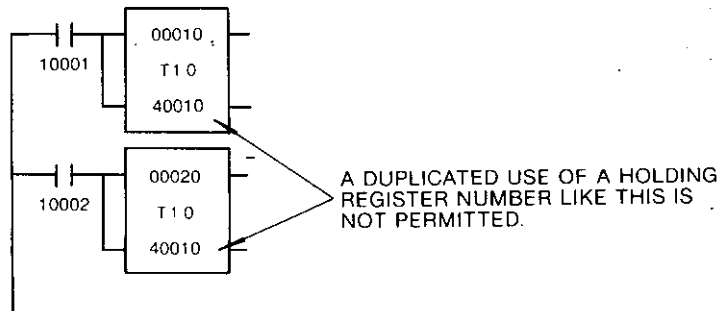
Example 3:



(4) Storing Timer Current Value

The register number of the holding register storing the current value of a timer cannot be the same as the register number of the holding register storing the current value of another timer or counter.

Example:

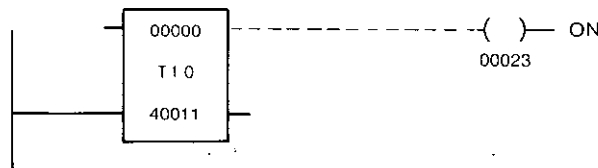


Due to current value changes, the holding register storing the current value of a timer cannot be used as the register storing arithmetic operation results. However, the register can be used as an operand.

(5) Preset Value of Timer

When the preset value of a timer is 0 and input 2 is ON, output 1 is ON (output 2 is OFF), regardless of ON/OFF operation of input 1.

Example:



(6) Relationship of Preset and Current Values

Ordinarily, the current value does not exceed the preset value, but it is possible to make the current value greater than the preset value by arithmetic operation or data transfer function. In such a case, the current value becomes equal to the preset value (output 1 is turned ON) as soon as the timer circuit is solved.

(7) Timer Error

Error of the timer is given as follows.

Maximum value of error = Unit of preset timer + 1 scan time

For example, if T1.0 is used for a 1-second timer, error may rise to 1 second (the timer may reach the limit a second earlier than the real time limit). Therefore, the use of T0.1 or T.01 is recommended in this case.

(8) Current Value of Timer at Power ON

During power failure, the timers of the GL60H/GL70H memorize the values before power failure. When the GL60H or GL70H is turned ON, the current value of the coil will be reset to 0 or remain unchanged depending on the type and status of the reference coil of contact used for input 2. Table 5.9 shows these relationships.

Table 5.9 Current Value of Timer at Power ON

Signal to Input 2			Current Value at Power ON	
Type of Reference Coil	Status	Contact		
Input Relay Latched Coil	ON	NO	Value at power failure	
		NC	0	
	OFF	NO	0	
		NC	Value at power failure	
Coil (Not including Latched Coil)	$n \leq m$	ON	NO	Value at power failure
			NC	0
	OFF	NO	0	
		NC	Value at power failure	
	$n > m$	OFF	NO	0
			NC	Value at power failure

- NOTE**
1. The number of the network including the reference coil of signal to input 2 is n and the number of the network including the timer circuit is m.
 2. If the signal to input 2 is composed of more than one contact, obtain the input 2 status from each contact status.
 3. The current value of the timer, at the CPU power ON, is the value read when the timer circuit has been solved during the first scanning cycle after power is applied to the GL60S.

5.3.5 Application Timer Circuits

On-delay and off-delay timers can be obtained by vertically shunting timer inputs 1 and 2. Various timer circuits can be realized by using different signal for inputs 1 and 2.

(1) ON-delay Timer

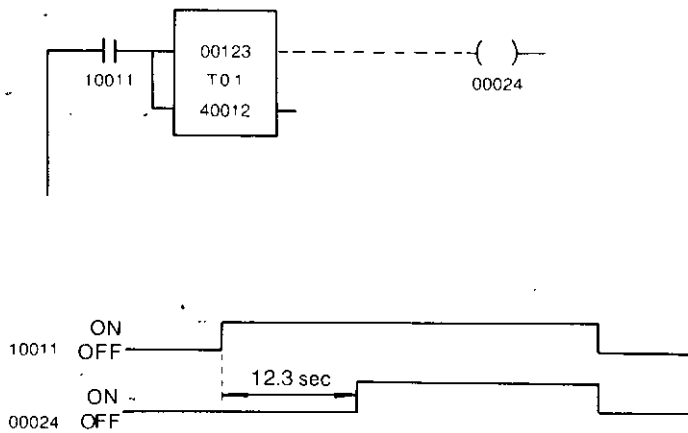


Fig. 5.15 Sample ON-delay Timer

(2) OFF-delay Timer

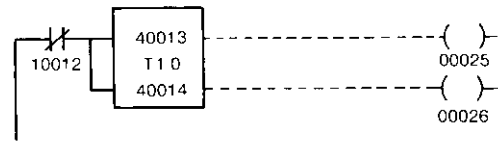
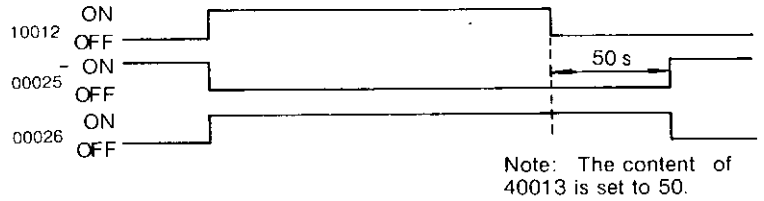


Fig. 5.16 Sample OFF-delay Timer



(3) One-shot Timer

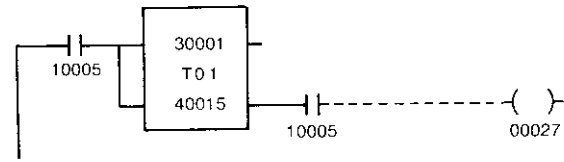
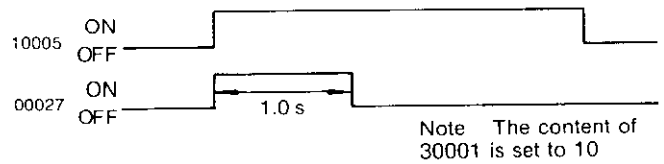


Fig. 5.17 Sample One-shot Timer



(4) Pulse Generating Circuit

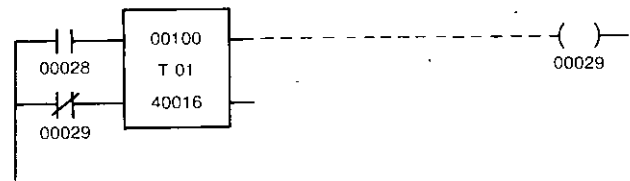
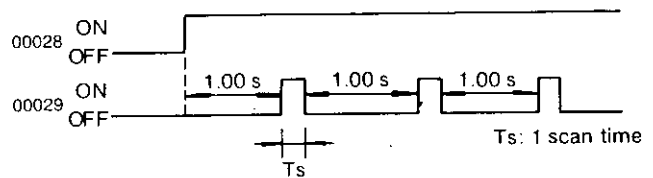


Fig. 5.18 Sample Basic Pulse Generating Circuit



(5) Flicker Relay

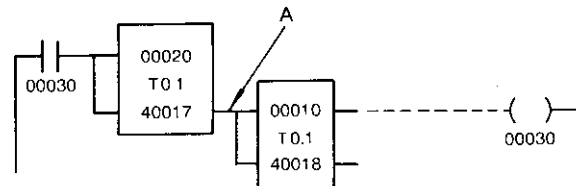
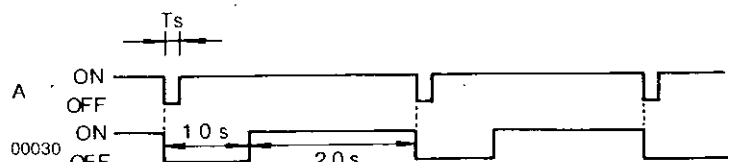


Fig. 5.19 Sample Flicker Relay



5.4 COUNTERS

5.4.1 Types of Counters

(1) Types

Two types of counters are available as shown in Table 5.10. As many counters as desired may be used, in a range of the program memory capacity and the number of holding registers.

Table 5.10 Types of Counters

Type	Symbol	Unit of Count	Limit of Count
Up Counter	UCTR	1 pulse	1-9,999 pulses
Down Counter	DCTR		

(2) Unit of Count

An up or down counter increases or decreases the current count by one pulse.

(3) Limit of Count

An up or down counter increases or decreases in a range of pulses counted (1 to 9,999 pulses). The upper limits of count are presettable.

5.4.2 Counter Configuration

(1) Form

Fig. 5.20 shows the basic form of a counter. A counter needs two elements placed vertically (top and bottom). Specify any of constant K or reference numbers referring to Table 5.18 ×CTR identifies a specific type of counter. Specify UCTR or DCTR in reference to Table 5.10.

Fig. 5.20 Counter General Form

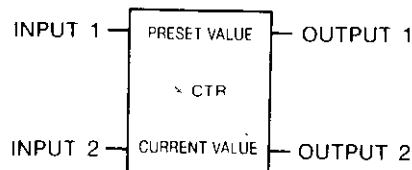


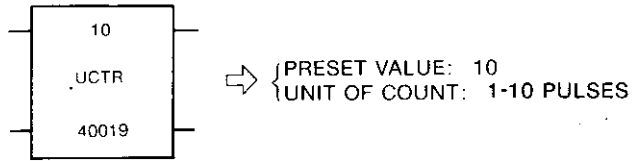
Table 5.11 Counter Elements

Element	Specified Numbers	Description
Top	Constant K (00001-09999) Any of the following: Input register (30001-30512) Holding register (40001-49999) Constant register (31001-35096) Link register (R0001-R1024)	The preset value of counter is constant K, or contents of register reference No.
Bottom	Holding register (40001-49999) Link register (R0001-R1024)	The current value of counter is stored in the specified reference.

(2) Preset Value

Specify a value of 1 to 9999 to determine the upper limit (variable) of count.

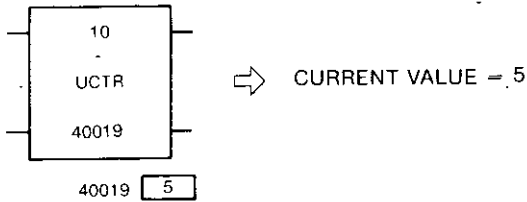
Example:



(3) Current Value

This is the value a counter has counted.

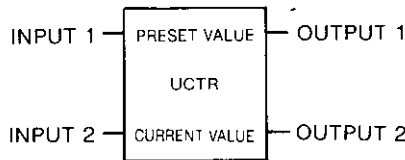
Example:



5.4.3 Function and Operation of Counter

5.4.3.1 Up Counter

(1) Up Counter Function



- When input 2 is ON, the up counter counts the number of times in which input 1 is turned from OFF to ON. The current value is increased by one every counting.
- When the current value becomes equal to the preset value, the up counter stops counting with output 1 turned ON and output 2 OFF.
- When input 2 is OFF, the up counter does not count even if input 1 is changed from OFF to ON. At this time, the current value is 0, output 1 is OFF and output 2 ON.

(2) Up Counter Operation

Fig. 5.21 and Table 5.12 show the timing chart and operation of the up counter.

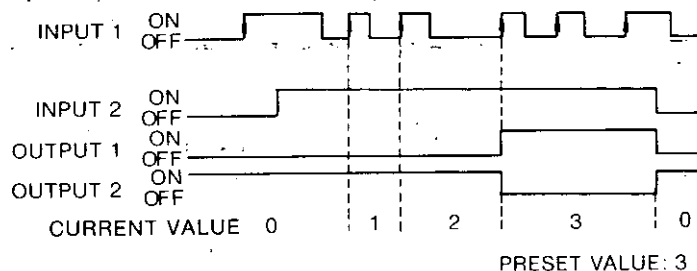


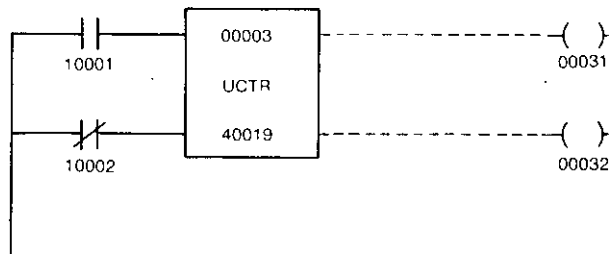
Fig. 5.21 Up Counter Operation

Table 5.12 Up Counter Operation

Input Status		Counter Status		Current Value	Output Status	
Input 1	Input 2				Output 1	Output 2
• ON Status • OFF Status • OFF→ON • ON→OFF	OFF	Reset status		0	OFF	ON
OFF→ON	ON	Operating status	Current value < Preset value	Increase (+1)	OFF*	ON*
			Current value = Preset value	Preset value	ON	OFF
• ON Status • ON→OFF • OFF Status	ON	Standstill status	Current value < Preset value	Constant	OFF	ON
			Current value = Preset value	Preset value	ON	OFF

* As a result of increasing current value, current value < preset value.

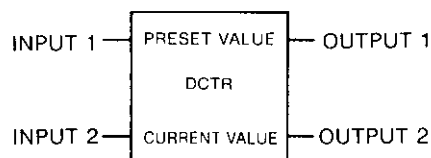
(3) Sample Up Counter



- The up counter counts only when input 2 is ON (input relay 10002 is OFF).
- When input 1 (input relay 10001) is turned from OFF to ON while input 2 is ON, the up counter counts and increases the current value (contents of 40019) by one.
- When the current value becomes equal to the preset value (3), the up counter stops counting and the output 1 (coil 00031) is turned ON and output 2 (coil 00032) OFF.
- When input 2 is OFF (input relay 10002 is ON), the up counter does not count even if input 1 is changed from OFF to ON. At this time, the current value is 0, output 1 OFF, and output 2 ON.

5.4.3.2 Down Counter

(1) Down Counter Function



- When input 2 is ON, the down counter counts the number of times in which input 1 is turned from OFF to ON. The current value is decreased by one every counting.
- When the current value becomes zero, the down counter stops counting with output 1 turned ON and output 2 OFF.
- When input 2 is OFF, the down counter does not count even if input 1 is changed from OFF to ON. At this time, the current value becomes equal to the preset value with output 1 turned OFF and output 2 ON.

(2) Down Counter Operation

Fig. 5.22 and Table 5.13 show the timing chart and operation of the down counter.

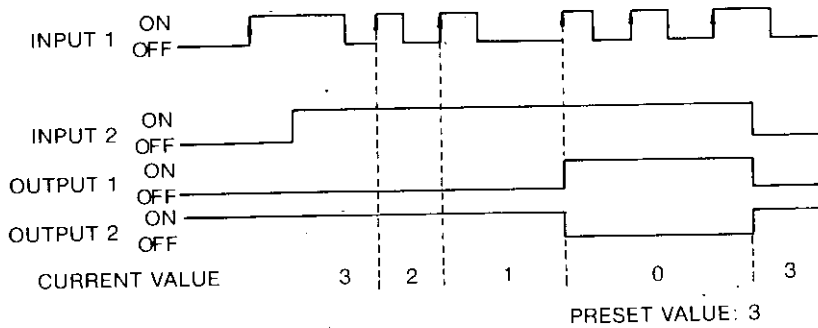


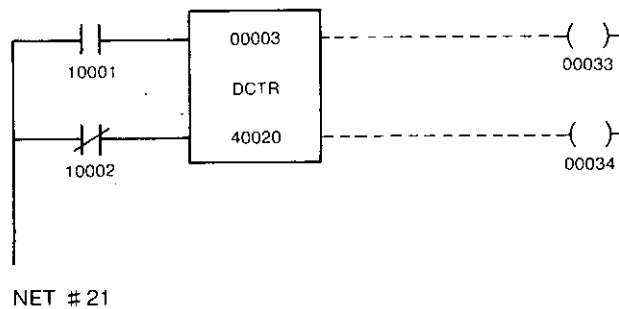
Fig. 5.22 Timing Chart of Down Counter

Table 5.13 Down Counter Operation

Input Status		Counter Status		Current Value	Output Status	
Input 1	Input 2				Output 1	Output 2
• OFF Status • ON Status • OFF→ON • ON→OFF	OFF	Reset status		Preset value	OFF	ON
OFF→ON	ON	Operating status	Current value > 0	Decrease (-1)	OFF*	ON*
			Current value = 0	0	ON	OFF
• OFF Status • ON Status • ON→OFF	ON	Standstill status	Current value > 0	Constant	OFF	ON
			Current value = 0	0	ON	OFF

* As a result of decreasing current value, current value > 0.

(3) Sample Down Counter



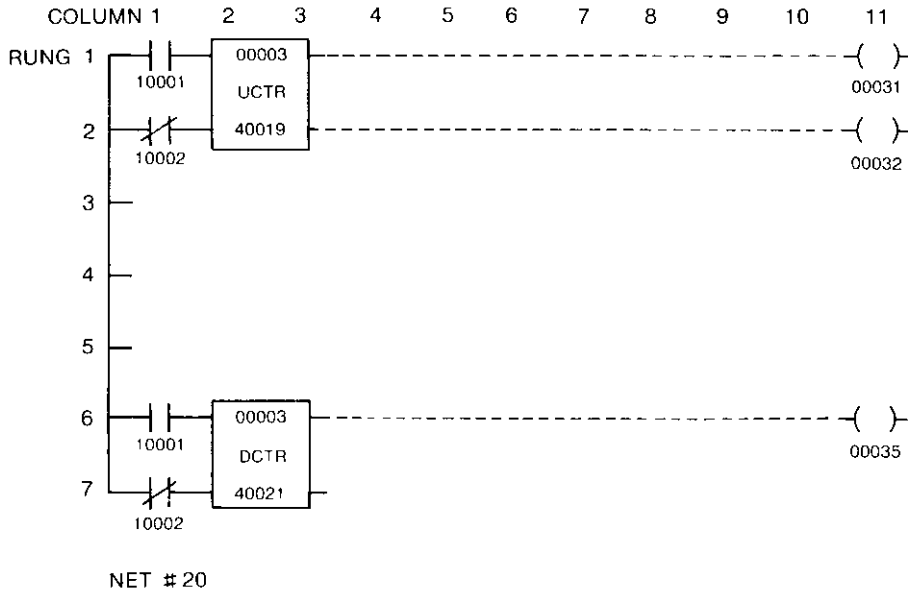
- The down counter counts only when input 2 is ON (input relay 10002 is OFF).
- When input 1 (input relay 10001) is turned from OFF to ON while input 2 is ON, the down counter decreases the current value (contents of 40020) by one.
- When the current value becomes zero, the down counter stops to counting and output 1 (coil 00033) is turned ON and output 2 (coil 00034) OFF.
- When input 2 is OFF (input relay 10002 is ON), the down counter does not count even if input 1 is changed from OFF to ON. At this time, current value becomes equal to the preset value and output 1 is turned OFF and output 2 ON.

5.4.4 Programming Counter Circuit and Precautions

(1) Programming Counter Circuit

An up and down counter occupies two elements placed vertically (top and bottom) in a network. It can be used at any intersection of the 7 lines-by-10 columns matrix, but the top element (preset value) cannot be located on line 7.

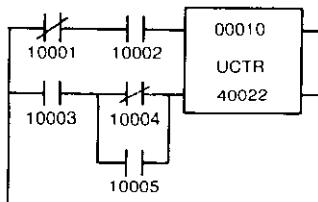
Example:



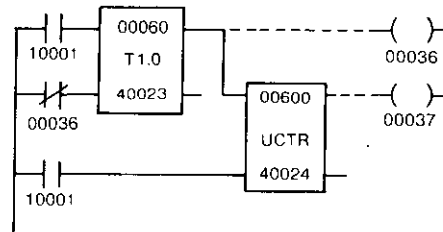
(2) Counter Inputs

Inputs to the counter may be outputs of relays, other counters, timers, arithmetic operations, and data processing circuits.

Example 1:



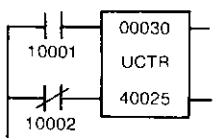
Example 2:



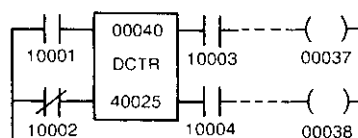
(3) Counter Outputs

Coils may not be connected to outputs 1 and 2 of a counter. It is permitted to insert a reley contact to the right of an output or to connect an output directly to an input of logic, except relays.

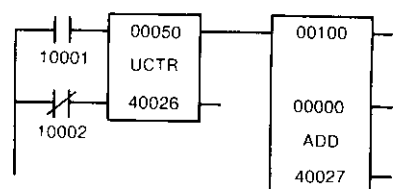
Example 1:



Example 2:



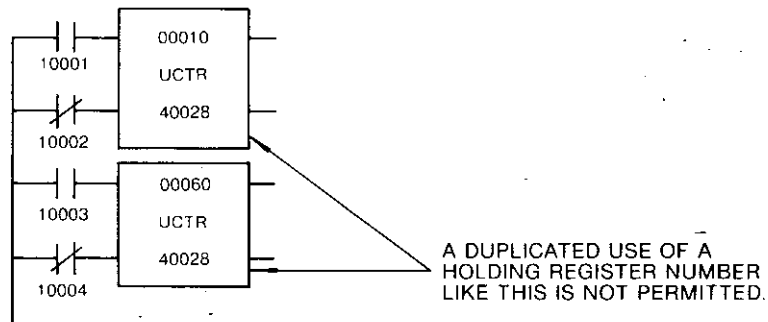
Example 3:



(4) Storing Counter Current Value.

The register number of the holding register storing the current value of a counter cannot be the same as the register number of the holding register storing the current value of another counter or timer. In special cases when used as an up/down counter (see (5) of Application Circuits of Counter), a register number may be used for different counters or timers.

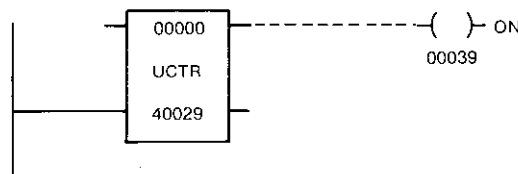
Example:



(5) Preset Value of Counter

When the preset value of an up- or down-counter is 0 and input 2 is ON, output 1 is ON (output 2 is OFF), regardless of ON/OFF operation of input 1.

Example:



(6) Relationship of Preset and Current Values

Ordinarily, the current value does not exceed the preset value, but it is possible to make the current value greater than the preset value by arithmetic operation or data transfer function. In such a case, the current value becomes equal to the preset value as soon as the counter circuit is solved.

(7) Input Pulse Width

When operating a counter by signals (count pulses) fed from an external device, one scan time or more is required for each pulse width (ON or OFF). In addition, delay of response of the input module which receives the input pulses must be taken into consideration.

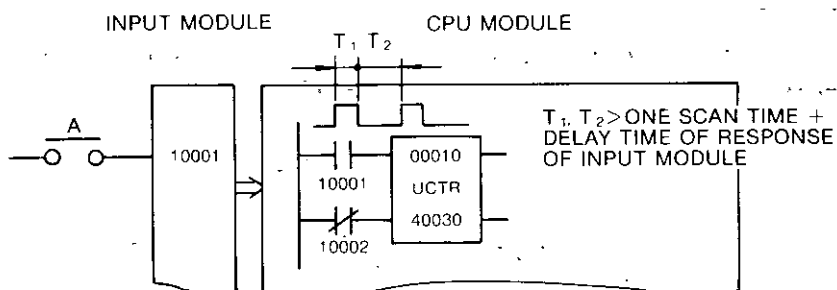


Fig. 5.23 Counter Operation by Signals Fed from an External Device

(8) Current Value of Counter at Power ON

Like the timer, the counter keeps the current value even if power fails. When the CPU is turned ON, the current value is determined by signals for input 2.

Table 5.14 Current Value of Counter at Power ON

Signal to Input 2			Current Value at Power ON		
Type of Reference Coil	Status	Contact	Up Counter	Down Counter	
Input Relay Latch Coil	ON	NO	Value at power failure	Value at power failure	
		NC	0	Preset value	
	OFF	NO	0	Preset value	
		NC	Value at power failure	Value at power failure	
Coil (Not including Latch Coil)	$n \leq m$	ON	Value at power failure	Value at power failure	
		NC	0	Preset value	
	OFF	NO	0	Preset value	
		NC	Value at power failure	Value at power failure	
	$n > m$	OFF	NO	0	Preset value
			NC	Value at power failure	Value at power failure

NOTE

1. The number of the network including the reference coil of signal to input 2 is n and the number of the network including the counter circuit is m .
2. If the signals to input 2 is composed of more than one contact, obtain the input 2 status from each contact status.
3. The current value of the counter at the CPU power ON, is the value read when the counter circuit has been solved during the first scanning cycle after power is supplied to the CPU.

5.4.5 Application Counter Circuits

(1) Large-capacity Counter

- A large-capacity counter (having a preset value of 10000 or more) can be created by combining multiple counters.
- Fig. 5.24 shows a large-capacity counter (preset value = $100 \times 150 = 15000$) using two up counters.

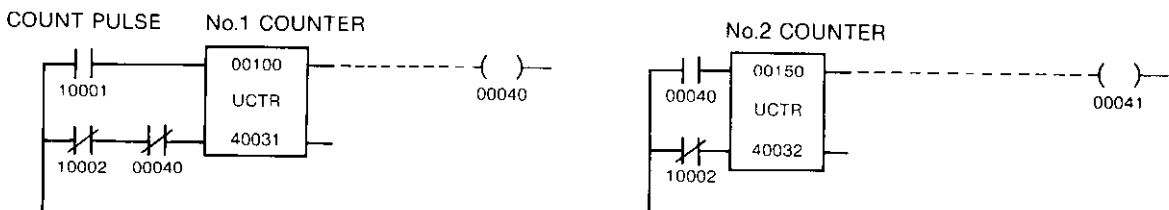
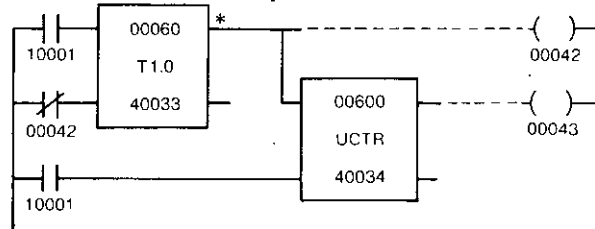


Fig. 5.24 Large-capacity Counter

- The No. 1 counter generates a pulse (ON time = 1 scan time) for the coil 00040 for every 100 input pulses.
- When the No. 2 counter counts 150 pulses output by the No. 1 counter, coil 00041 is turned ON to indicate that 15000 pulses (100×150) have been counted.
- If the contents of 40031 is x ($0 \leq x \leq 100$) and the contents of 40032 is y ($0 \leq y \leq 150$), the current value (pulse count) of the large-capacity counter is $x + 100y$.

(2) Long-time Timer

- A long-timer (having a preset value of 10000 or more) can be created by combining a timer and an up counter.
- Fig. 5.25 shows a long-time timer (preset value = $60 \times 600 = 36000$) using a 1-second timer and an up counter.



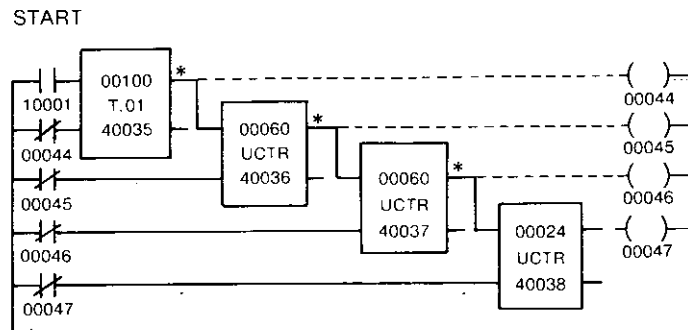
*Requires horizontal shunt.

Fig. 5.25 Long-time Timer

- The 1-second timer generates a pulse (ON time = 1 scan time) for coil 00042 for every 60 seconds while input relay 10001 is ON.
- When the up counter counts 600 pulses output by the 1-second timer, coil 00043 is turned ON to indicate that 36000 seconds (60×600) have elapsed.
- If the contents of 40033 is x ($0 \leq x \leq 60$) and the contents of 40034 is y ($0 \leq y \leq 600$), the current value of the long-timer is $x + 60y$ (the number of seconds the timer has operated).

(3) Clock

- A clock circuit can be created by combining a timer with an up counter.
- Fig. 5.26 shows an example of a clock circuit.



*Requires horizontal shunt.

Fig. 5.26 Sample Timer Counter Cascaded Logic

- This clock operates while input relay 10001 is ON. The contents of 40036, 40037, and 40038 give the values in seconds, minutes, and hours, respectively. The clock is not provided with a compensating circuit.

(4) Circuit for Measuring Scan Time

- It is possible to make up a circuit for measuring the scanning time of the GL60H or GL70H by combining an up counter with a 0.1-second timer.
- Fig. 5.27 shows a sample measuring circuit for scan time.

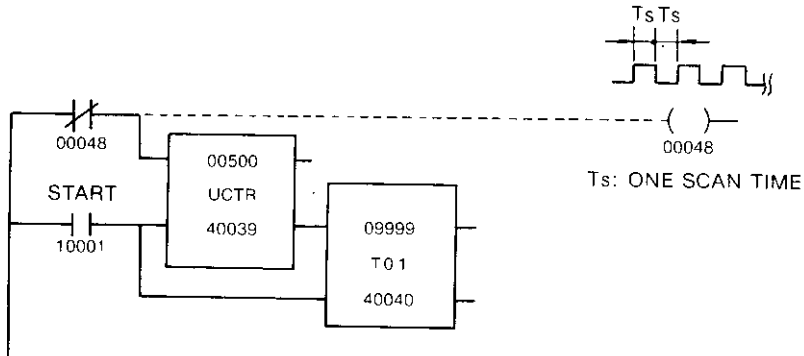


Fig. 5.27 A Circuit for Measuring Scan Time

- When input relay 10001 is turned ON, both the up counter and the 0.1-second timer start to measure the CPU scanning time.
- Coil 00048 is turned ON and OFF alternately every scanning cycle. When the up counter counts 500 pulses (alternations of ON and OFF), the output 2 is turned OFF and the 0.1-second timer stops.
- Now the timer's current value (contents of 40040) is the time taken to count 500 pulses or, the time of 1000 scanning cycles given in units of 0.1 second.
- If, for example, the value is 205, then the average value of one scan time is $205 \times 0.1 \div 1000 = 20.5$ milli-seconds.

(5) Up/Down Counter

- An up/down counter can be made by combining an up counter with a down-counter.
- Fig. 5.28 shows an example of an up/down counter.

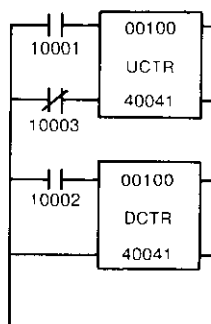


Fig. 5.28 Up/Down Counter

- The current value (contents of 40041) is increased by one each time input relay 10001 is turned from OFF to ON.
- The current value is decreased by one each time input relay 10002 is turned from OFF to ON.
- The current value is reset to zero when input relay 10003 is turned ON.

5.5 ARITHMETIC FUNCTIONS

5.5.1 Types of Arithmetic Functions

The arithmetic operations are addition, subtraction, multiply, and divide applied on operand V_1 by operand V_2 . There are eight variations of arithmetic functions as described in Table 5.15.

Table 5.15 Types of Arithmetic Functions

Type	Symbol	Operator	Range of Operand V_1	Range of Operand V_2	Reference Page
Addition	ADD	$V_1 + V_2$	0-9,999		84
Double-precision Addition	DADD		0-99,999,999		86
Subtraction	SUB	$V_1 - V_2$	0-9,999		88
Double-precision Subtraction	DSUB		0-99,999,999		90
Multiply	MUL	$V_2 \times V_2$	0-9,999		92
Double-precision Multiply	DMUL		0-99,999,999		93
Divide	DIV	$V_1 \div V_2$	0-9,999 (0-99,989,999) *	0-9,999	95
Double-precision Divide	DDIV		0-9,999,999,899,999,999	0-99,999,999	98

*The range of V_1 becomes as shown in parentheses when two successive registers are used.

5.5.2 Addition (ADD)

(1) Function

Operates addition in 4-digit decimal without sign.

(2) Form

- Fig. 5.29 shows the form of addition (ADD)
- ADD is the symbol denoting the addition.
- Addition operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.16, specify any of constant K, reference number of various registers.

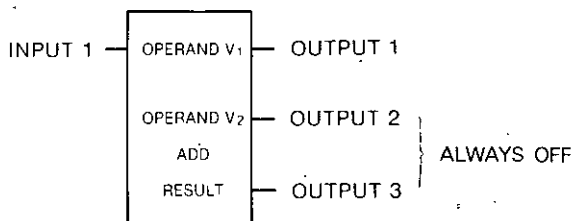


Fig. 5.29 ADD General Form

Table 5.16 Elements of ADD Function

Element Position	Specified Number	Description
Top	<ul style="list-style-type: none"> • Constant K (00000-09999) Any one of the following: • Input register (30001 - 30512) • Holding register (4000 - 49999) 	<ul style="list-style-type: none"> • When constant K is specified, the value is the operand ($V_1 = 0$ to 9,999). • When register reference Nos. are specified, the contents are the operand ($V_1 = 0$ to 9,999).
Middle	<ul style="list-style-type: none"> • Constant register (31001 - 35096) • Link register (R0001 - R1024) • Timer register (50001 - 50512) 	<ul style="list-style-type: none"> • When constant K is specified, the value is the operand ($V_2 = 0$ to 9,999). • When register reference Nos. are specified, the contents are the operand ($V_2 = 0$ to 9,999).
Bottom	<ul style="list-style-type: none"> • Holding register (40001 - 49999) • Link register (R0001 - R1024) 	The result of addition function (0 to 9,999) is stored in $4 \times \times \times \times$ or $R \times \times \times \times$.

(3) Operation

- By the addition (ADD), $V_1 + V_2$ is calculated when the input 1 is ON. The result is treated as follows.
 - (a) If $V_1 + V_2 \leq 9,999$,
 $V_1 + V_2$ is stored in R. The Output 1 remains OFF.
 - (b) If $V_1 + V_2 \geq 10,000$, $V_1 + V_2 - 10,000$ is stored in R. The output 1 is turned ON.
- The outputs 2 and 3 are always OFF.
- The result remains in R even after the input 1 is turned from ON to OFF.
- Table 5.17 shows an ADD operation.

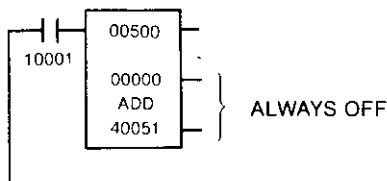
Table 5.17 ADD Operation

Input 1	Condition	Operation	Output 1
ON	$V_1 + V_2 \leq 9,999$	$V_1 + V_2 \rightarrow R$	OFF
	$V_1 + V_2 \geq 10,000$	$V_1 + V_2 - 10,000 \rightarrow R$	ON
OFF	None	Not operated.	OFF

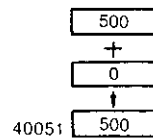
NOTE $V_1 + V_2 \rightarrow R$ indicates that $V_1 + V_2$ operation result is stored in R.

(4) Example

Example 1:



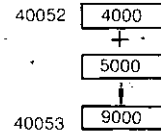
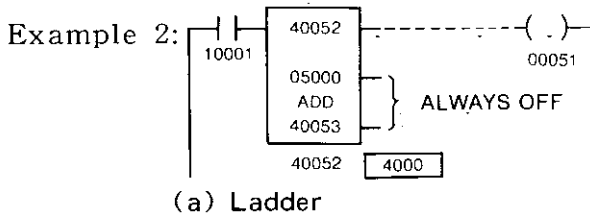
(a) Ladder



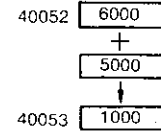
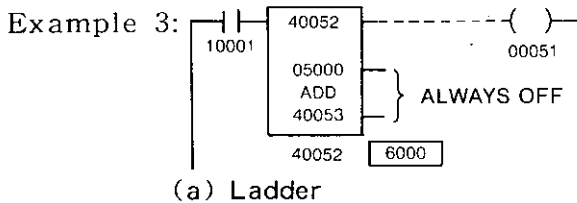
(b) ADD Operation

ADD in (a) executes the operation of (b) when input relay 10001 is ON. The output 1 remains OFF. The result remains in 40051 even after input relay 10001 is turned OFF.

5.5.2 Addition (Cont'd)



ADD in (a) executes the operation of (b) when input relay 10001 is ON. The output 1 remains OFF. The result remains in 40053 even after input relay 10001 is turned OFF.



ADD in (a) executes the operation of (b) when input relay 10001 is ON. The output 1 is turned ON. The result remains in 40053 even after input relay 10001 is turned OFF.

5.5.3 Double-precision Addition (DADD)

(1) Function

Operates double-precision addition in 8-digit decimal without sign.

(2) Form

- Fig. 5.30 shows the form of double-precision addition (DADD).

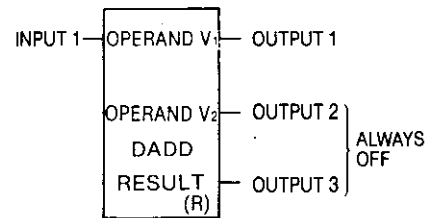


Fig. 5.30 DADD General Form

- DADD is the symbol denoting the double-precision addition.
- Double-precision addition operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.18, specify any reference number of various registers.

Table 5.18 Elements of DADD Function

Element Position	Specified Number	Description
Top	Either one of the following: • Input register (30001 - 30511) • Holding register (40001 - 49998)	Operand ($V_1 = 0$ to 99,999,999) is stored as follows. $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 \\ \boxed{V_1H} & \boxed{V_1L} \end{array}$ V_1H : Higher-place 4 digits of V_1 V_1L : Lower-place 4 digits of V_1
Middle	• Constant register (31001 - 35096) • Link register (R0001 - R1023)	Operand ($V_2 = 0$ to 99,999,999) is stored as follows. $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 \\ \boxed{V_2H} & \boxed{V_2L} \end{array}$ V_2H : Higher-place 4 digits of V_2 V_2L : Lower-place 4 digits of V_2
Bottom	• Holding register (40001 - 49998) • Link register (R0001 - R1023)	Result of operation ($V_1 + V_2 = 0$ to 99,999,999) is stored as follows $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 \\ \boxed{(V_1 + V_2)H} & \boxed{(V_1 + V_2)L} \end{array}$ $(V_1 + V_2)H$: Higher-place 4 digits of $(V_1 + V_2)$ $(V_1 + V_2)L$: Lower-place 4 digits of $(V_1 + V_2)$

(3) Operation

- By DADD, $V_1 + V_2$ is calculated when the input 1 is ON. The result is treated as follows.
 - (a) If $V_1 + V_2 \leq 99,999,999$,
The four higher-place digits of $V_1 + V_2$ are stored in R and the four lower-place digits in R + 1. The output 1 remains OFF.
 - (b) If $V_1 + V_2 \geq 100,000,000$,
The four higher-place digits of $V_1 + V_2 - 100,000,000$ are stored in R and the four lower-place digits in R + 1. The output 1 is turned ON.
- The outputs 2 and 3 are always OFF.
- The result remains in R and R + 1 even after the input 1 is turned from ON to OFF.
- Table 5.19 shows a DADD operation.

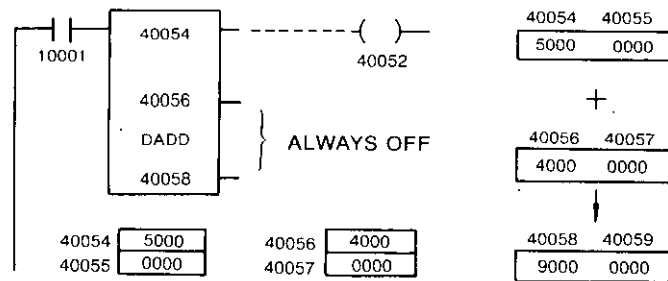
Table 5.19 Double-precision Addition Operation

Input 1	Condition	Operation	Output 1
ON	$V_1 + V_2 \leq 99,999,999$	$V_1 + V_2 \rightarrow R$ (Higher-place 4 digits), $R + 1$ (Lower-place 4 digits).	OFF
ON	$V_1 + V_2 \geq 100,000,000$	$V_1 + V_2 - 100,000,000 \rightarrow R$ (Higher-place 4 digits), $R + 1$ (Lower-place 4 digits).	ON
OFF	None	Not operated.	OFF

NOTE $V_1 + V_2 \rightarrow R$ and $R + 1$ indicate that $V_1 + V_2$ operation result is stored in R and R + 1, respectively.

(4) Example

Example 1:



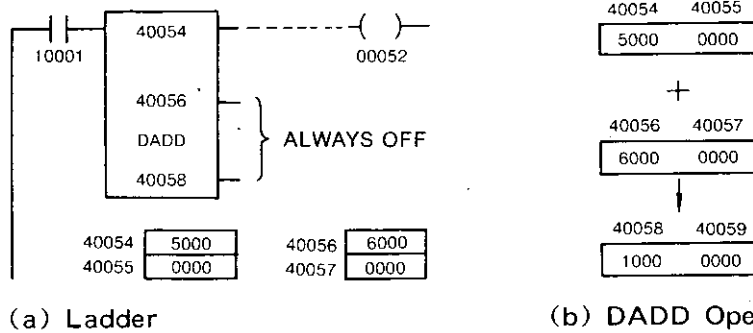
(a) Ladder

(b) DADD Operation

DADD in (a) executes the operation of (b) when input relay 10001 is ON. The output 1 remains OFF. The results remain in 40058 and 40059 even after input relay 10001 is turned OFF.

5.5.3 Double-precision Addition (DADD) (Cont'd)

Example 2:



DADD in (a) executes the operation of (b) when input relay 10001 is ON. The output 1 (coil 00052) is turned ON. The result remains in 40058 and 40059 even after input relay 10001 is turned OFF.

5.5.4 Subtraction (SUB)

(1) Function

Operates subtraction in 4-digit decimal without sign.

(2) Form

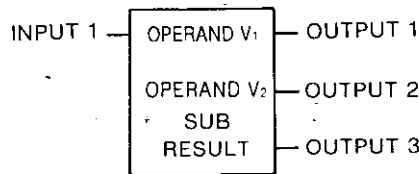


Fig. 5.31 SUB General Form

- Fig. 5.31 shows the form of subtraction (SUB).
- Subtraction operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.20, specify any of constant K or the reference number of various registers.

Table 5.20 Elements of SUB Function

Element Position	Specified Number	Description
Top	<ul style="list-style-type: none"> • Constant K (0000-9999) • Any one of the following: <ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) 	<ul style="list-style-type: none"> • When constant K is specified, the value is the operand ($V_1 = 0$ to 9,999). • When register reference Nos. are specified, the contents are the operand ($V_1 = 0$ to 9,999).
Middle	<ul style="list-style-type: none"> • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512) 	<ul style="list-style-type: none"> • When constant K is specified, the value is the operand ($V_2 = 0$ to 9,999) • When register reference Nos. are specified, the contents are the operand ($V_2 = 0$ to 9,999).
Bottom	<ul style="list-style-type: none"> • Holding register (40001-49999) • Link register (R0001-R1024) 	The result of subtraction function ($ V_1 - V_2 = 0$ to 9,999) is stored in $4\times\times\times$ or $R\times\times\times$.

(3) Operation

- By the subtraction (SUB), $V_1 - V_2$ will be calculated when the input 1 is ON. The result is treated as follows.
 - If $V_1 > V_2$
 $V_1 - V_2$ is stored in R and only the output 1 is turned ON.
 - If $V_1 = V_2$,
 $V_1 - V_2 = 0$ is stored in R and only the output 2 is turned ON.
 - If $V_1 < V_2$,
 $V_2 - V_1$ is stored in R and only the output 3 is turned ON.
- The result remains in R even after the input 1 is turned from OFF to ON.
- Table 5.21 shows a SUB operation.

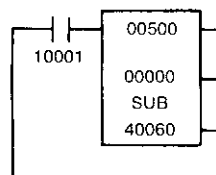
Table 5.21 SUB Operation

Input 1	Condition	Operation	Output 1	Output 2	Output 3
ON	$V_1 > V_2$	$V_1 - V_2 \rightarrow R$	ON	OFF	OFF
	$V_1 = V_2$	$0 \rightarrow R$	OFF	ON	OFF
	$V_1 < V_2$	$V_2 - V_1 \rightarrow R$	OFF	OFF	ON
OFF	None	Not operated.	OFF	OFF	OFF

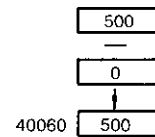
NOTE $V_1 - V_2 \rightarrow R$ indicates that $V_1 - V_2$ operation result is stored in R.

(4) Example

Example 1:



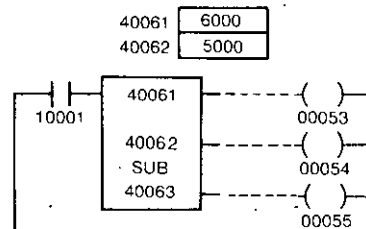
(a) Ladder



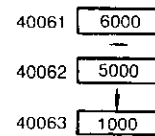
(b) SUB Operation

SUB in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 1 is turned ON. The result remains in 40060 even after input relay 10001 is turned OFF.

Example 2:



(a) Ladder

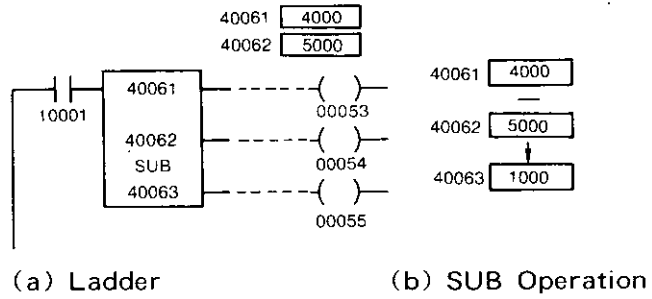


(b) SUB Operation

SUB in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 1 (coil 00053) is turned ON. The result remains in 40063 even after input relay 10001 is turned OFF.

5.5.4 Subtraction (Cont'd)

Example 3:



SUB in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 3 (coil 00055) is turned ON. The result remains in 40063 even after input relay 10001 is turned OFF.

5.5.5 Double-precision Subtraction (DSUB)

(1) Function

Operates double-precision subtraction in 8-digit decimal without sign.

(2) Form

- Fig. 5.32 shows the form of double-precision subtraction (DSUB).

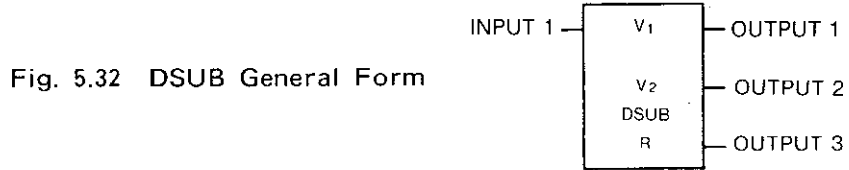


Fig. 5.32 DSUB General Form

- DSUB is the symbol denoting the double-precision subtraction.
- Double-precision subtraction operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.22, specify any reference number of various registers.

Table 5.22 Elements of DSUB Function

Element Position	Specified Number	Description
Top	Either one of the following: • Input register (30001-30511) • Holding register (40001-49998)	Operand ($V_1 = 0$ to 99,999,999) is stored as follows. $\begin{array}{cc} \times \times \times \times \times & \times \times \times \times \times + 1 \\ \boxed{V_1H} & \boxed{V_1L} \end{array}$ V_1H : Higher-place 4 digits of V_1 V_1L : Lower-place 4 digits of V_1
Middle	• Constant register (31001-35095) • Link register (R0001-R1023)	Operand ($V_2 = 0$ to 99,999,999) is stored as follows. $\begin{array}{cc} \times \times \times \times \times & \times \times \times \times \times + 1 \\ \boxed{V_2H} & \boxed{V_2L} \end{array}$ V_2H : Higher-place 4 digits of V_2 V_2L : Lower-place 4 digits of V_2
Bottom	• Holding register (40001-49998) • Link register (R0001-R1023)	Result of operation ($ V_1 - V_2 = 0$ to 99,999,999) is stored as follows. $\begin{array}{cc} \times \times \times \times \times & \times \times \times \times \times + 1 \\ \boxed{ V_1 - V_2 H} & \boxed{ V_1 - V_2 L} \end{array}$ $ V_1 - V_2 H$: Higher-place 4 digits of $ V_1 - V_2 $ $ V_1 - V_2 L$: Lower-place 4 digits of $ V_1 - V_2 $

(3) Operation

- By the double-precision subtraction (DSUB), $V_1 - V_2$ is calculated when the input 1 is ON. The result is treated as follows.
 - If $V_1 > V_2$,
The four higher-place digits of $V_1 - V_2$ are stored in R and the four lower-place digits in R+1. Only the output 1 is turned ON.
 - If $V_1 = V_2$,
 $V_1 - V_2 = 0$ is stored in R and R+1 and only the output 2 is turned ON.
 - If $V_1 < V_2$,
The four higher-place digits of $V_2 - V_1$ are stored in R and the four lower-place digits in R+1. Only the output 3 is turned ON.
- The result remains in R and R+1 even after the input 1 is turned from ON to OFF.
- Table 5.23 shows a double-precision subtraction operation.

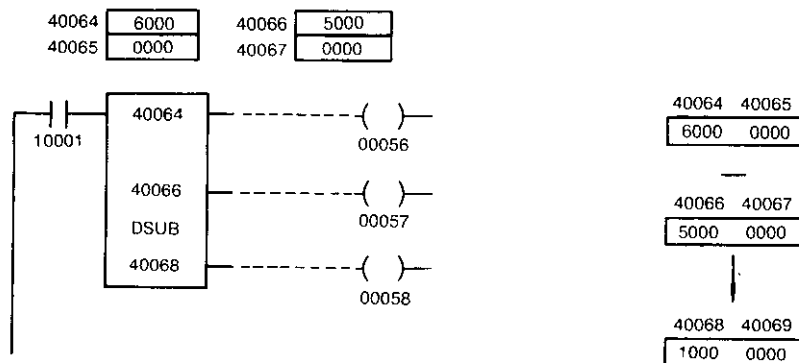
Table 5.23 DSUB Operation

Input 1	Condition	Operation	Output 1	Output 2	Output 3
ON	$V_1 > V_2$	$V_1 - V_2 \rightarrow R$ (Higher-place 4 digits), $R+1$ (Lower-place 4 digits).	ON	OFF	OFF
	$V_1 = V_2$	$0 \rightarrow R, R+1$	OFF	ON	OFF
	$V_1 < V_2$	$V_2 - V_1 \rightarrow R$ (Higher-place 4 digits), $R+1$ (Lower-place 4 digits).	OFF	OFF	ON
OFF	None	Not operated.	OFF	OFF	OFF

NOTE $V_1 - V_2 \rightarrow R$ and $R+1$ indicates that $V_1 - V_2$ operation result is stored in R and R+1, respectively.

(4) Example

Example 1:



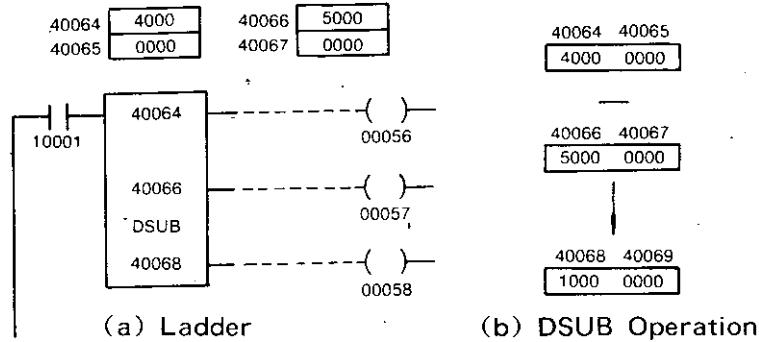
(a) Ladder

(b) DSUB Operation

DSUB in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 1 (coil 00056) is turned ON. The result remains in 40068 and 40069 even after input relay 10001 is turned OFF.

5.5.5 Double-precision Subtraction (DSUB) (Cont'd)

Example 2:



DSUB in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 3 (coil 00058) is turned ON. The result remains in 40068 and 40069 even after input relay 10001 is turned OFF.

5.5.6 Multiply (MUL)

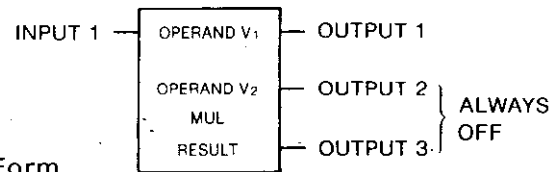
(1) Function

Operates multiply in 4-digit decimal without sign.

(2) Form

- Fig. 5.33 shows the form of multiply (MUL).

Fig. 5.33 MUL General Form



- MUL is the symbol denoting the multiply.
- Multiply operation requires three elements placed vertically (top, middle, and bottom) Referring to Table 5.24, specify any of constant K or the reference number of various registers.

Table 5.24 Elements of MUL Function

Element Position	Specified Number	Description
Top	Any one of the following: • Input register (30001-30512) • Holding register (40001-49999)	• When constant K is specified, the value is the operand ($V_1 = 0$ to 9,999). • When register reference Nos. are specified, the contents are the operand ($V_1 = 0$ to 9,999).
Middle	• Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512)	• When constant K is specified, the value is the operand ($V_2 = 0$ to 9,999). • When register reference Nos. are specified, the contents are the operand ($V_2 = 0$ to 9,999).
Bottom	• Holding register (40001-49999) • Link register (R0001-R1024)	Result of operation ($ V_1 + V_2 = 0$ to 99,999,999) is stored as follows. $\begin{array}{ccc} \times \times \times \times & & \times \times \times \times + 1 \\ \boxed{(V_1 + V_2)H} & & \boxed{(V_1 + V_2)L} \end{array}$ ($V_1 + V_2$)H: Higher-place 4 digits of ($V_1 + V_2$)H. ($V_1 + V_2$)L: Lower-place 4 digits of ($V_1 + V_2$)L.

(3) Operation

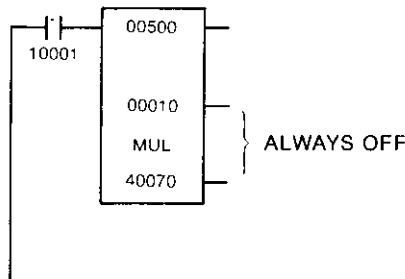
- By the multiply (MUL), $V_1 \times V_2$ is calculated when the input 1 is ON. The result is treated as follows. The four higher-place digits of $V_1 \times V_2$ are stored in R and the four lower-place digits in R+1. Output 1 is turned ON.
- Table 5.25 shows a MUL operation.

Table 5.25 MUL Operation

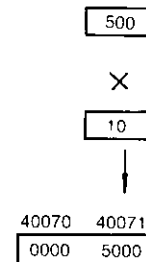
Input 1	Condition	Operation	Output 1
ON	None	$V_1 \times V_2 \rightarrow R$ (Higher-place 4 digits), R+1 (Lower-place 4 digits).	ON
OFF	None	Not operated.	OFF

(4) Example

Example 1:



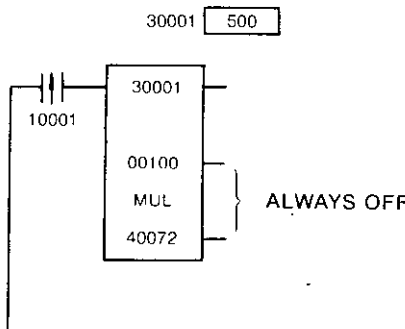
(a) Ladder



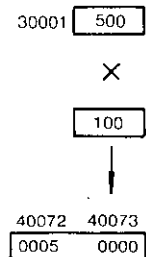
(b) MUL Operation

MUL in (a) executes the operation of (b) when input relay 10001 is ON. The output 1 is turned ON. The result remains in 40070 and 40071 even after input relay 10001 is turned OFF.

Example 2:



(a) Ladder



(b) MUL Operation

MUL in (a) executes the operation of (b) only during the scanning cycle when input relay 10001 is turned ON. The output 1 is turned ON. The result remains in 40072 and 40073.

5.5.7 Double-precision Multiply (DMUL)

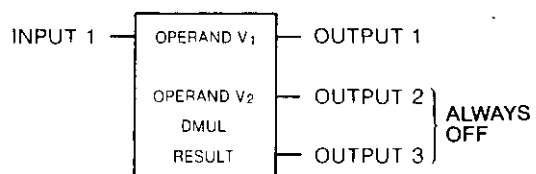
(1) Function

Operates double-precision multiply in 8-digit decimal without sign.

(2) Form

- Fig. 5.34 shows the form of double-precision multiply (DMUL).

Fig. 5.34 DMUL General Form



5.5.7 Double-precision Multiply (DMUL) (Cont'd)

- DMUL is the symbol denoting the double-precision multiply.
- Double-precision multiply operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.26, specify any reference.

Table 5.26 Elements of DMUL Function

Element Position	Specified Number	Description
Top	Either one of the following: • Input register (30001-30511) • Holding register (40001-49998)	Operand ($V_1 = 0$ to 99,999,999) is stored as follows. $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 \\ \boxed{V_1H} & \boxed{V_1L} \end{array}$ V_1H : Higher-place 4 digits of V_1 V_1L : Lower-place 4 digits of V_1
Middle	• Constant register (31001-35095) • Link register (R0001-R1023)	Operand ($V_2 = 0$ to 99,999,999) is stored as follows. $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 \\ \boxed{V_2H} & \boxed{V_2L} \end{array}$ V_2H : Higher-place 4 digits of V_2 V_2L : Lower-place 4 digits of V_2
Bottom	• Holding register (40001-49996) • Link register (R0001-R1021)	Result of operation ($V_1 \times V_2 = 0$ to 99,999,999) is stored as follows. $\begin{array}{cccc} \times \times \times \times & \times \times \times \times + 1 & \times \times \times \times + 2 & \times \times \times \times + 3 \\ \boxed{(V_1 \times V_2)_1} & \boxed{(V_1 \times V_2)_2} & \boxed{(V_1 \times V_2)_3} & \boxed{(V_1 \times V_2)_4} \end{array}$ $(V_1 \times V_2)_1$: Most significant 4 digits of $(V_1 \times V_2)$ $(V_1 \times V_2)_2$: Higher-place 4 digits of $(V_1 \times V_2)$ $(V_1 \times V_2)_3$: Lower-place 4 digits of $(V_1 \times V_2)$ $(V_1 \times V_2)_4$: Least significant 4 digits of $(V_1 \times V_2)$

(3) Operation

- By the double-precision multiply (DMUL), $V_1 \times V_2$ is calculated when the input 1 is ON. The result is treated as follows.

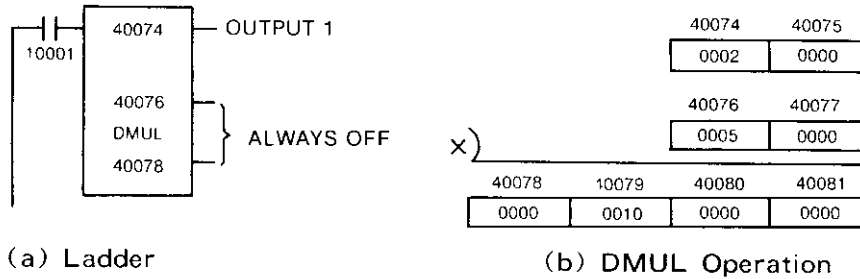
The most significant four digits of $V_1 \times V_2$ are stored in R, the second most significant four digits in R+1, the second least significant four digits in R+2, the least significant four digits in R+3, and the output 1 is turned ON.

- The result remains in R, R+1, R+2, and R+3 after the input 1 is turned from ON to OFF.
- Table 5.27 shows a DMUL multiply.

Table 5.27 DMUL Operation

Input 1	Condition	Operation	Output 1
ON	None	$V_1 \times V_2 \rightarrow R$ (Most significant 4 digits), R+1 (2nd most significant 4 digits), R+2 (2nd least significant 4 digits), R+3 (Least significant 4 digits).	ON
OFF	None	Not operated.	OFF

(4) Example



DMUL in (a) executes the operation of (b) when input relay 10001 is ON. The output 1 is turned ON. The result remains in 40078-40081 even after input relay 10001 is turned OFF.

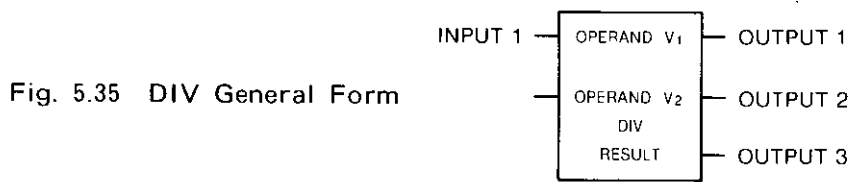
5.5.8 Divide (DIV)

(1) Function

Operates divide in 4-digit decimal without sign.

(2) Form

- Fig. 5.35 shows the form of divide (DIV).



- DIV is the symbol denoting the divide.
- Divide operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.28, specify any of constant K or reference number of various registers.

Table 5.28 Elements of DIV Function

Element Position	Specified Number	Description
Top	<ul style="list-style-type: none"> • Constant K (0-9999) Any one of the following: <ul style="list-style-type: none"> • Input register (30001-30511) • Holding register (40001-49998) • Constant register (31001-35095) • Link register (R0001-R1023) • Timer register (50001-50511) 	<ul style="list-style-type: none"> • When constant K is specified, the value is the operand ($V_1=0$ to 9,999). • When register reference Nos. are specified, the operand ($V_1=0$ to 99,989,999) is stored as follows. $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 \\ \boxed{V_1H} & \boxed{V_1L} \end{array}$ V_1H: Higher-place 4 digits of V_1 V_1L: Lower-place 4 digits of V_1
Middle	<ul style="list-style-type: none"> • Constant K (1-9999) Any one of the following: <ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512) 	<ul style="list-style-type: none"> • When constant K is specified, the value is the operand ($V_2=1$ to 9,999). • When register reference Nos. are specified, the contents are the operand ($V_2=1$ to 9,999).
Bottom	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023) 	Result of divide function ($V_1 \div V_2$) is stored as follows. Where input 2 is OFF, Where input 2 is ON, $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 & \times \times \times \times & \times \times \times \times + 1 \\ \boxed{} & \boxed{} & \boxed{} & \boxed{} \end{array}$ The quotient The remainder The integer quotient The decimal quotient

(3) Operation

By the divide (DIV), $V_1 \div V_2$ is calculated when the input 1 is ON. The result is treated as follows.

① If the input 2 is OFF,

The quotient of $V_1 \div V_2$ is stored in R and the remainder in R+1. Only the output 1 is turned ON.

② If the input 2 is ON,

The integer part of the quotient of $V_1 \div V_2$ is stored in R and the decimal part (rounded off to the fifth decimal place) in R+1. Only the output 1 is turned ON.

③ The result remains in R and R+1 even after the input 1 is turned from ON to OFF.

④ In the following cases, divide operation is not executed and zero is placed in each of R and R+1.

(I) When $V_2 = 0$, the output 3 is turned ON.

(II) If the quotient or the integer part of quotient overflows in R, the output 2 is turned ON.

Tables 5.29 and 5.30 show a divide operation (DIV).

Table 5.29 DIV Operation (Constant in Top Place)

Input 1	Input 2	Condition	Operation	Output 1	Output 2	Output 3
ON	OFF	$V_2 \neq 0$	$V_1 \div V_2 \rightarrow \begin{matrix} R \\ R+1 \end{matrix}$ (Quotient) (Remainder)	ON	OFF	OFF
		$V_2 = 0$	$0 \rightarrow \begin{matrix} R \\ R+1 \end{matrix}$	OFF	OFF	ON
ON	OF	$V_2 \neq 0$	$V_1 \div V_2 \rightarrow \begin{matrix} R \\ R+1 \end{matrix}$ (Integer quotient) (Decimal quotient)	ON	OFF	OFF
		$V_2 = 0$	$0 \rightarrow \begin{matrix} R \\ R+1 \end{matrix}$	OFF	OFF	ON
OFF	ON OFF	None	Not operated.	OFF	OFF	OFF

NOTE $V_1 \div V_2 \rightarrow R$ and $R+1$ indicate that a result of $V_1 \div V_2$ is stored R and R+1 respectively.

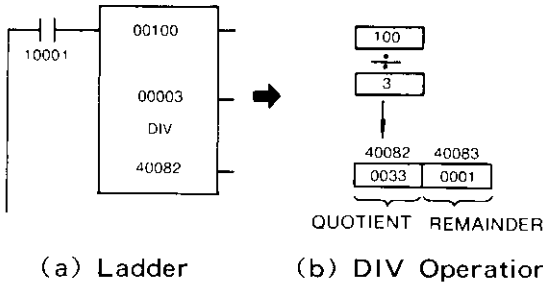
Table 5.30 DIV Operation (Register in Top Place)

Input 1	Input 2	Condition	Operation	Output 1	Output 2	Output 3
ON	OFF	$V_2 \neq 0, V_{1H} < V_2$	$(V_{1H} \times 10000 + V_{1L}) \div V_2 \rightarrow \begin{matrix} R \\ R+1 \end{matrix}$ (Quotient) (Remainder)	ON	OFF	OFF
		$V_2 \neq 0, V_{1H} \geq V_2$	$0 \rightarrow \begin{matrix} R \\ R+1 \end{matrix}$	OFF	ON	OFF
		$V_2 = 0$	$0 \rightarrow \begin{matrix} R \\ R+1 \end{matrix}$	OFF	OFF	ON
ON	ON	$V_2 \neq 0, V_{1H} < V_2$	$(V_{1H} \times 10000 + V_{1L}) \div V_2 \rightarrow \begin{matrix} R \\ R+1 \end{matrix}$ (Integer quotient) (Decimal quotient)	ON	OFF	OFF
		$V_2 \neq 0, V_{1H} \geq V_2$	$0 \rightarrow \begin{matrix} R \\ R+1 \end{matrix}$	OFF	ON	OFF
		$V_2 = 0$	$0 \rightarrow \begin{matrix} R \\ R+1 \end{matrix}$	OFF	OFF	ON
OFF	ON OFF	None	Not operated.	OFF	OFF	OFF

NOTE $V_1 \div V_2 \rightarrow R$ and $R+1$ indicate that a result of $V_1 \div V_2$ is stored R and R+1 respectively.

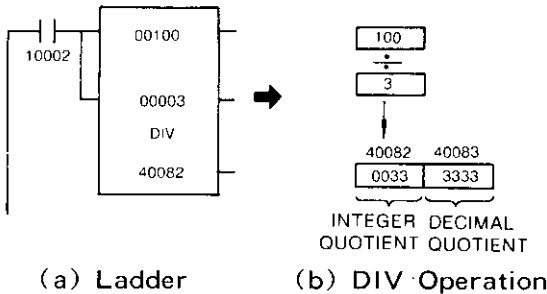
(4) Example - Divide

Example 1:



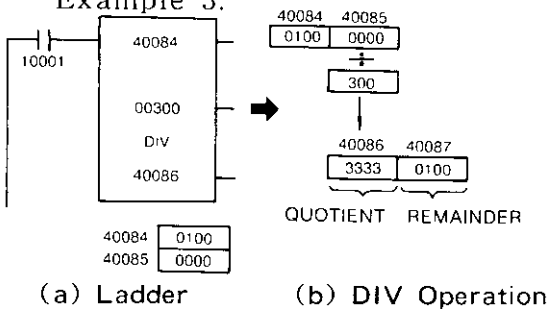
DIV in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 1 is turned ON. The result remains in 40082 and 40083 even after input relay 10001 is turned OFF.

Example 2:



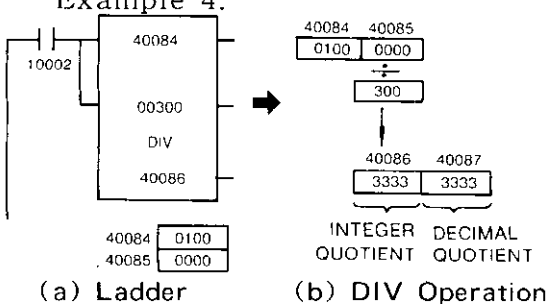
DIV in (a) executes the operation of (b) when input relay 10002 is ON. Only the output 1 is turned ON. The result remains in 40082 and 40083 even after input relay 10002 is turned OFF.

Example 3:



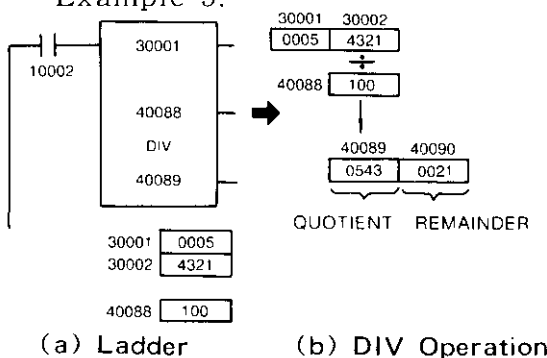
DIV in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 1 is turned ON. The result remains in 40086 and 40087 even after input relay 10001 is turned OFF.

Example 4:



DIV in (a) executes the operation of (b) when input relay 10002 is ON. Only the output 1 is turned ON. The result remains in 40086 and 40087 even after input relay 10002 is turned OFF.

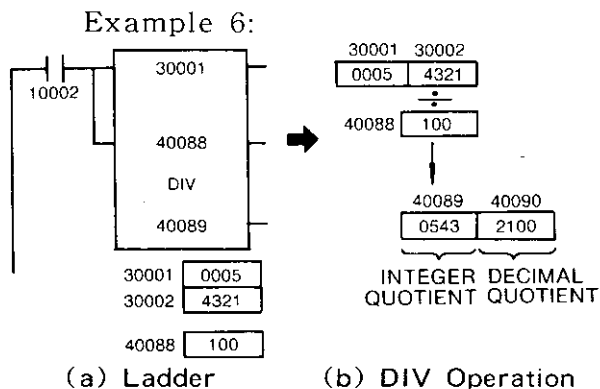
Example 5:



DIV in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 1 is turned ON. The result remains in 40089 and 40090 even after input relay 10001 is turned OFF.

5

5.5.8 Divide (Cont'd)



DIV in (a) executes the operation of (b) when input relay 10002 is ON. Only the output 1 is turned ON. The result remains in 40089 and 40090 even after input relay 10002 is turned OFF.

5.5.9 Double-precision Divide Function (DDIV)

(1) Function

Operates double-precision divide function in 8-digit decimal without sign.

(2) Form

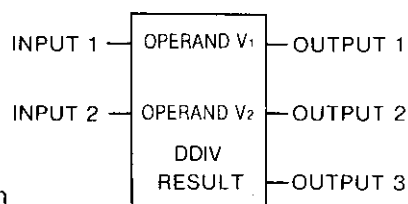


Fig. 5.36 DDIV General Form

- Fig. 5.36 shows the form of double-precision divide (DDIV).
- DDIV is the symbol denoting the double-precision divide.
- Double-precision divide operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.31, specify any reference number of various registers.

Table 5.31 Elements of DDIV Function

Element Position	Pecified Number	Description
Top	Either one of the following: • Input register (30001-30509). • Holding register (40001-49996) • Constant register (31001-35093) • Link register (R0001-R1021)	The operand ($V_1=0$ to 9,999,999,999,999,999) is stored as follows. $\begin{array}{cccc} \text{x x x x} & \text{x x x x} & \text{x x x x} & \text{x x x x} \\ \text{+1} & \text{+2} & \text{+3} & \end{array}$ $\begin{array}{ c c c c } \hline V_1H_1 & V_1H_2 & V_1L_1 & V_1L_2 \\ \hline \end{array}$ V_1H_1 : Most significant 4 digits of V_1 V_1H_2 : 2nd most significant 4 digits of V_1 V_1L_1 : 2nd least significant 4 digits of V_1 V_1L_2 : Least significant 4 digits of V_1
Middle	Either one of the following: • Input register (30001-30511) • Holding register (40001-49998) • Constant register (31001-35095) • Link register (R0001-R1023)	The operand ($V_2=1$ to 99,999,999) is stored as follows. $\begin{array}{cc} \text{x x x x} & \text{x x x x} \\ \text{+1} & \end{array}$ $\begin{array}{ c c } \hline V_2H & V_2L \\ \hline \end{array}$ V_2H : Higher-place 4 digits of V_2 V_2L : Lower-place 4 digits of V_2
Bottom	Either one of the following: • Holding register (40001-49996) • Link register (R0001-R1021)	The result of operation ($V_1 \div V_2$) is stored as follows. Where input 2 is OFF, $\begin{array}{cccc} \text{x x x x} & \text{x x x x} & \text{x x x x} & \text{x x x x} \\ \text{+1} & \text{+2} & \text{+3} & \end{array}$ $\begin{array}{ c c c c } \hline & & & \\ \hline \end{array}$ Quotient of ($V_1 \div V_2$) Remainder of ($V_1 \div V_2$) Where input 2 is ON, $\begin{array}{cccc} \text{x x x x} & \text{x x x x} & \text{x x x x} & \text{x x x x} \\ \text{+1} & \text{+2} & \text{+3} & \end{array}$ $\begin{array}{ c c c c } \hline & & & \\ \hline \end{array}$ Integer quotient of ($V_1 \div V_2$) Decimal quotient of ($V_1 \div V_2$)

(3) Operation

By the double-precision divide (DDIV), $V_1 \div V_2$ is calculated when the input 1 is ON. The result is treated as follows.

① If the input 2 is OFF,

- The four higher-place digits of the quotient of $V_1 \div V_2$ are stored in R and the four lower-place digits in R+1.
- The four higher-place digits of the remainder of $V_1 \div V_2$ are stored in R+2 and the four lower-place digits in R+3.
- Only the output 1 is turned ON.

② If the input 2 is ON,

- The four higher-place digits of the integer part of the quotient of $V_1 \div V_2$ are stored in R and the four lower-place digits in R+1.
- The four high-place digits of the decimal part of the quotient of $V_1 \div V_2$ are stored in R+2 and the four lower-place digits in R+3.
- Only the output 1 is turned ON.

③ The result remains in each of R, R+1, R+2, and R+3 even after the input 1 is turned from ON to OFF.

④ In the following cases, the divide operation is not executed and zero is placed in each of R, R+1, R+2, and R+3.

(I) When $V_2 = 0$, the output 3 is turned ON.

(II) If the quotient or the integer part of quotient overflows in R and R+1, the output 2 is turned ON.

Example: Where $V_1 = 5,000,000,000$, and $V_2 = 10$, the quotient 5,000,000,000 cannot be stored in R and R+1.

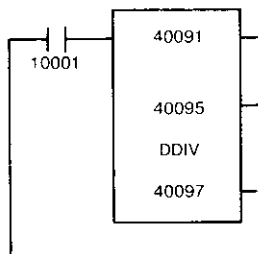
Table 5.32 shows a double-precision divide operation (DDIV).

Table 5.32 DDIV Operation

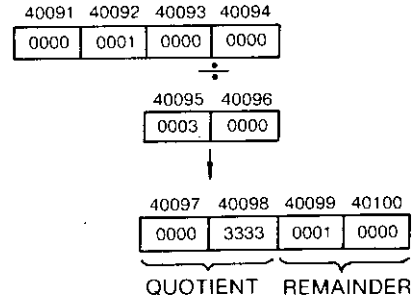
Input 1	Input 2	Condition	Operation	Output 1	Output 2	Output 3
ON	OFF	$V_2 = (V_{H1} \times 10^4 + V_{L1}) \neq 0$, $(V_{H1} \times 10^4 + V_{L1}) < V_2$ Higher-place 8 digits of V_1 :	$\frac{(V_{H1} \times 10^{12} + V_{H1} \times 10^8 + V_{L1} \times 10^4 + V_{L1})}{V_2} \rightarrow$	ON	OFF	OFF
		$V_2 \neq 0$, $(V_{H1} \times 10^4 + V_{L1}) \geq V_2$	$0 \rightarrow R, R+1, R+2, R+3$	OFF	ON	OFF
		$V_2 = 0$	$0 \rightarrow R, R+1, R+2, R+3$	OFF	OFF	ON
ON	ON	$V_2 = (V_{H1} \times 10^2 + V_{L1}) \neq 0$, $(V_{H1} \times 10^2 + V_{L1}) < V_2$ Higher-place 8 digits of V_1 :	$\frac{(V_{H1} \times 10^{12} + V_{H1} \times 10^8 + V_{L1} \times 10^2 + V_{L1})}{V_2} \rightarrow$	ON	OFF	OFF
		$V_2 \neq 0$, $(V_{H1} \times 10^2 + V_{L1}) \geq V_2$	$0 \rightarrow R, R+1, R+2, R+3$	OFF	ON	OFF
		$V_2 = 0$	$0 \rightarrow R, R+1, R+2, R+3$	OFF	OFF	ON
OFF	ON, OFF	None	Not operated.	OFF	OFF	OFF

(4) Example

Example 1:



40091	0000
40092	0001
40093	0000
40094	0000
40095	0003
40096	0000

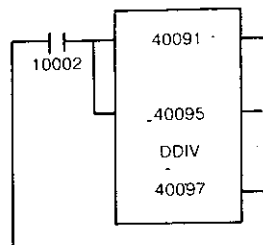


(a) Ladder

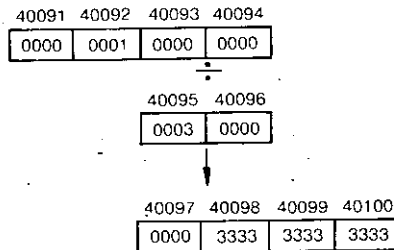
(b) DDIV Operation

DDIV in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 1 is turned ON. The result remains in 40097 to 40100 even after input relay 10001 is turned OFF.

Example 2:



40091	0000
40092	0001
40093	0000
40094	0000
40095	0003
40096	0000



(a) Ladder

(b) DDIV Contents

DDIV in (a) executes the operation of (b) when input relay 10002 is ON. Only the output 1 is turned ON. The result remains in 40097 to 40100 even after input relay 10001 is turned OFF.

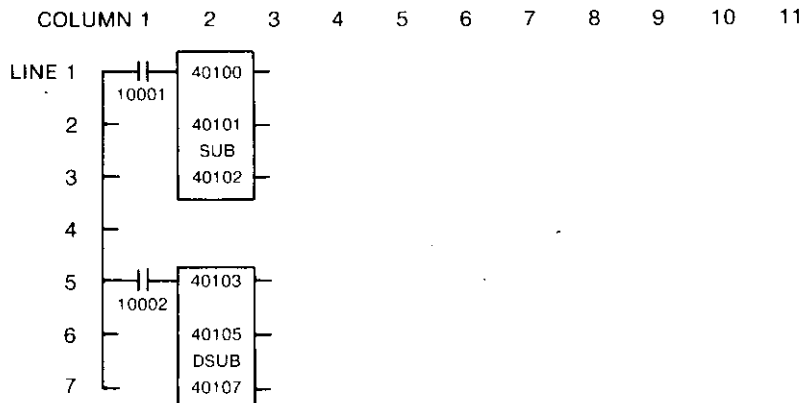
5.5.10 Programming Arithmetic Logic and Precautions

In all arithmetic operations, add, subtraction and multiply functions require only input 1, and divide requires only inputs 1 and 2. But the programming panel gives display as each output element line which may be connected to each input element line.

(1) Programming Arithmetic Logic

All arithmetic operations require three elements placed vertically (top, middle, and bottom) in a network. They can be used at any intersection of the 7 lines-by-10 columns matrix, however the top element (operand) cannot be used on either line 6 nor line 7.

Example:

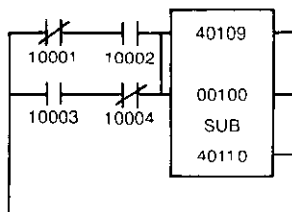


5

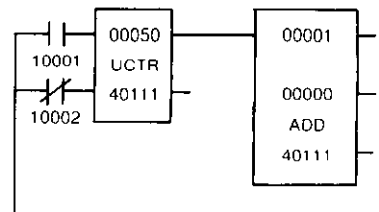
(2) Inputs of Arithmetic Logic

Inputs to the arithmetic logic may be outputs of relays, timers, counters, data processing circuits other arithmetic operations.

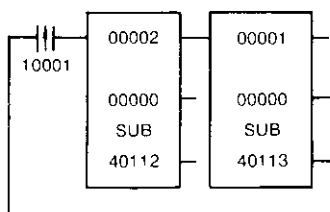
Example 1:



Example 2:



Example 3:

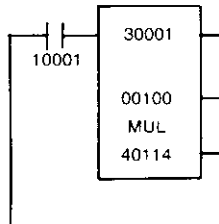


(3) Outputs of Arithmetic Logic

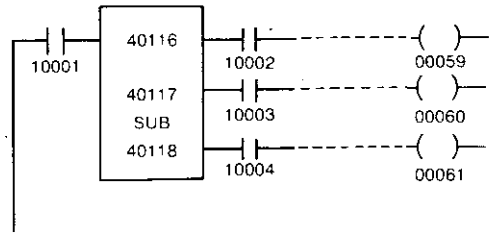
Coils need not be connected to three output nodes (1, 2, and 3) of an arithmetic function. A relay contact may be connected to the output nodes on the right or connect the output nodes directly to an input node of an arithmetic circuit, except relays.

If results of addition and double-precision addition operations exceed 9,999 or 99,999,999, respectively, only output 1 is ON. When other arithmetic circuits are cascade-connected to outputs of addition or double-precision addition, use proper care.

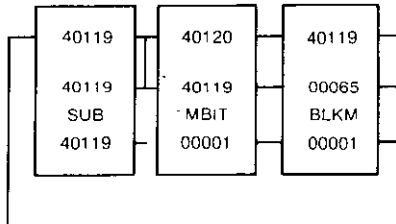
Example 1:



Example 2:



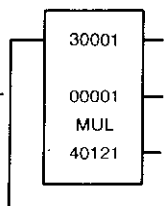
Example 3:



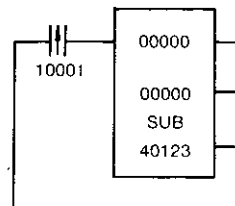
(4) Execution of Arithmetic Operations (Only One Scanning Cycle)

To execute a constant arithmetic operation, connect the inputs directly to the power rail on left. To execute it only in one scan, use a transitional contact as an input.

Example 1:



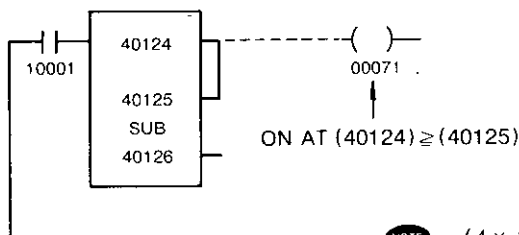
Example 2:



(5) ORed Outputs of Arithmetic Operations

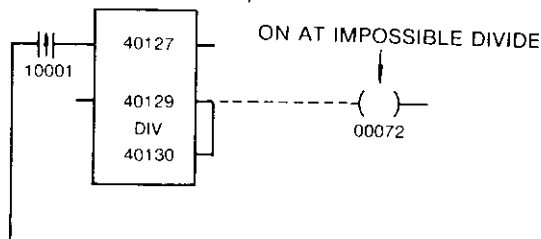
Outputs of the subtraction, the double-precision subtraction, the divide, and double-precision divide can be ORed by using a vertical shunt on the output side.

Example 1:

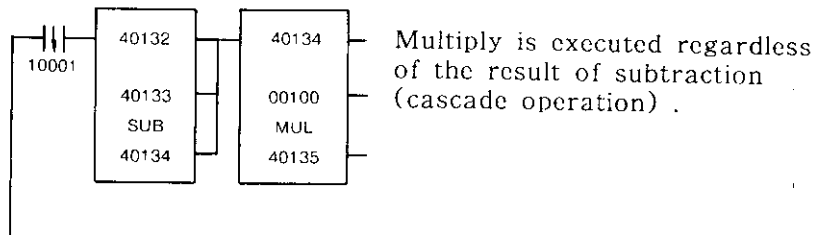


NOTE (4 × × × ×) 4 × × × × contents

Example 2:



Example 3:



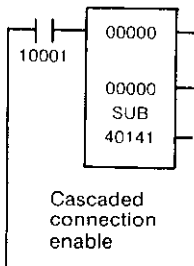
5.5.11 Example – Application Circuits of Arithmetic Functions

(1) Clearing Contents of Holding Registers to 0

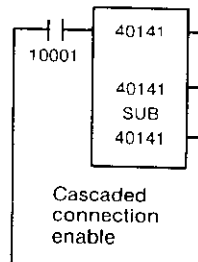
- To clear one holding register to 0

This is performed in three ways as shown in Examples 1 through 3 below. In all cases, the contents of 40141 become 0 when input relay 10001 is ON.

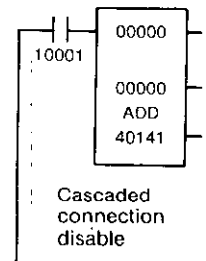
Example 1:



Example 2:

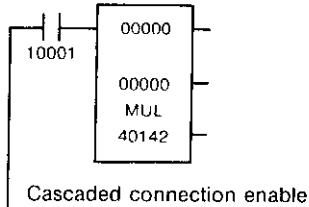


Example 3:



- To clear two successive holding registers to 0

Example:



When input relay 10001 is ON, the contents of 40142 and 40143 become 0.

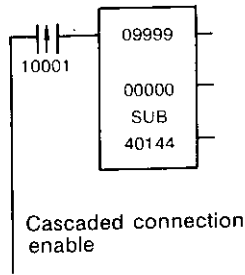
5.5.11 Example – Application Circuits of Arithmetic Functions (Cont'd)

(2) Setting Constant or Contents of a Register to Holding Register

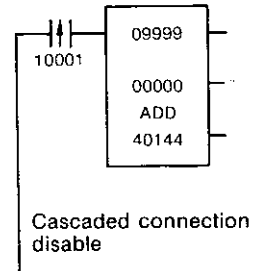
- To set the constant in a holding register

This can be performed in two ways as shown in Examples 1 and 2 below. In both cases, 9999 is set in 40144 during the scanning cycle when input relay 10001 is turned ON.

Example 1:



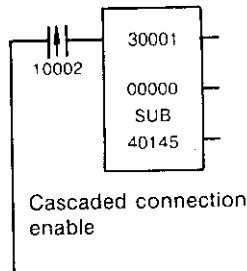
Example 2:



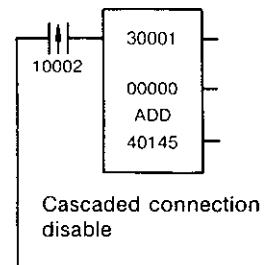
- To set the content of a register in a holding register

This can be performed in two ways as shown in Examples 1 and 2 below. In both cases, the contents of 30001 is set in 40145 during the scanning cycle when input relay 10002 is turned ON.

Example 1:



Example 2:

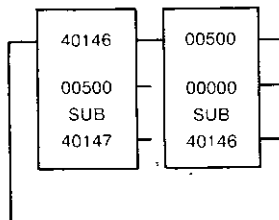


NOTE If the content of 30001 is greater than 10,000, it cannot be set correctly to 40145. The value of 30001 minus 10,000 is set to 40145. Use the method of Example 1 in this case.

(3) Limiting Contents of Holding Register

- To limit the maximum value

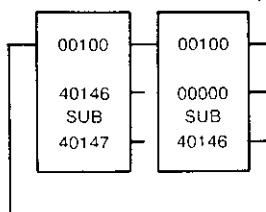
Example:



NOTE The maximum value of the contents of 40146 is limited to 500, i.e. $(40146) \leq 500$.

- To limit the minimum value

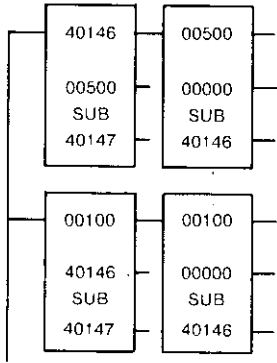
Example:



NOTE The minimum value of the contents of 40146 is limited to 100, i.e. $100 \leq (40146)$.

⑧ To limit the range

Example:



NOTE

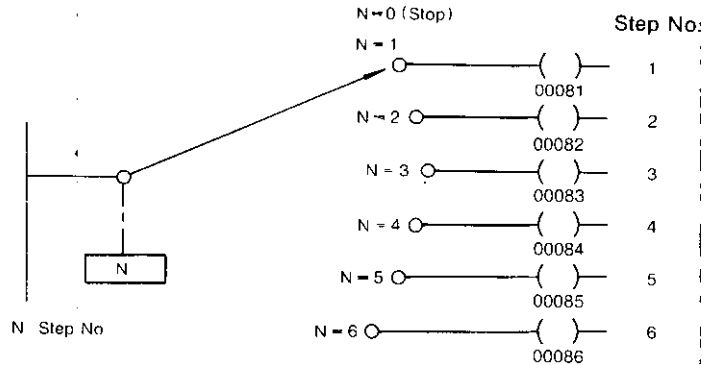
The range of the contents of 40146 is limited in a range of 100 to 500, i.e. $100 \leq (40146) \leq 500$.

(4) Stepping Switch (Having a few Steps)

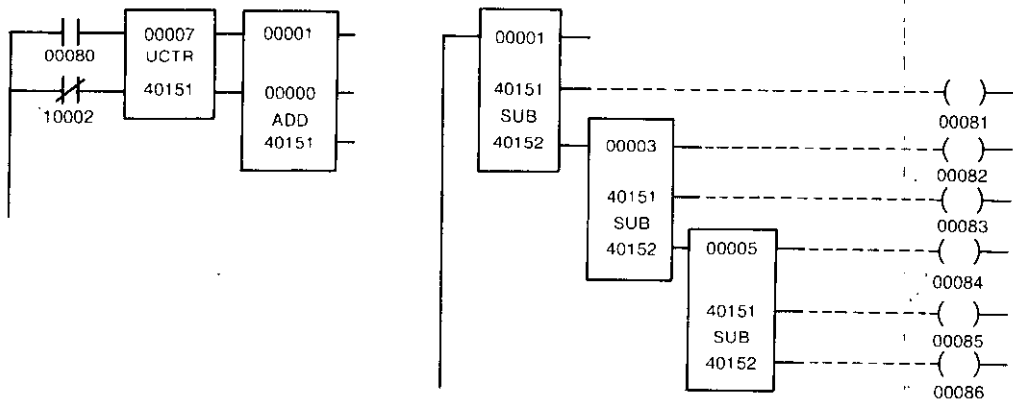
A stepping switch having a few steps can be programmed in the following way.

Example: Stepping switch with 6 steps

(a) Step Circuitry



(b) Ladder Diagram



5.5.11 Example—Application Circuits of Arithmetic Functions (Cont'd)

(5) Scaling

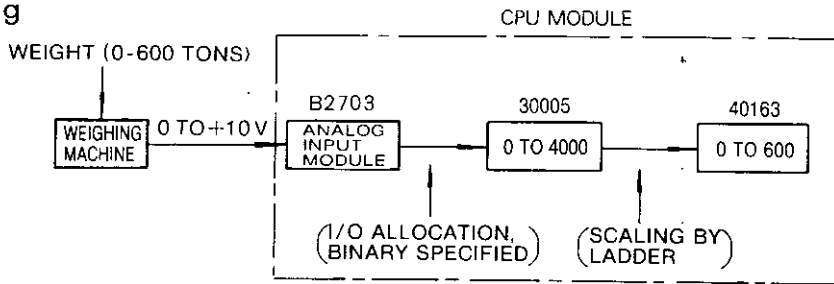
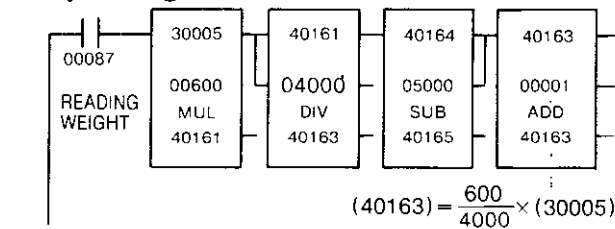


Fig. 5.37 Reading Analog Signal and Scaling

- Assume that a voltage signal of 0 to +10 V comes from the weighing machine to the analog input module B1075-1, as shown in Fig. 5.37, which converts the voltage signal to a numeric data of 0 to 4000 and enters it in input register 30005.
- The numeric data of 0 to 4000 entered in 30005 can be converted to a value of 0 to 600 (tons) by using the ladder circuit shown in Fig. 5.38.



NOTE The quotient of division is obtained by rounding the first decimal place.

Fig. 5.38 Scaling Circuit

(6) Saving I/O Modules

It is very easy to let the GL60H or GL70H read a value set by a digital switch or let a digital indicator display data stored in the GL60H or GL70H by allocating the register to the I/O module. The 2000-series I/O module can deal with 4-digit BCD data. Therefore, efficiency is the highest when it deals with four BCD digits.

The following is a method to let a single module process two data having different dimensions by using multiply and divide functions. Fig. 5.39 shows an example to let an input module read two numeric data of 2 digits each. Fig. 5.40 shows an example to let an output module output numeric data of 1-and 3-digit.

① Saving input module

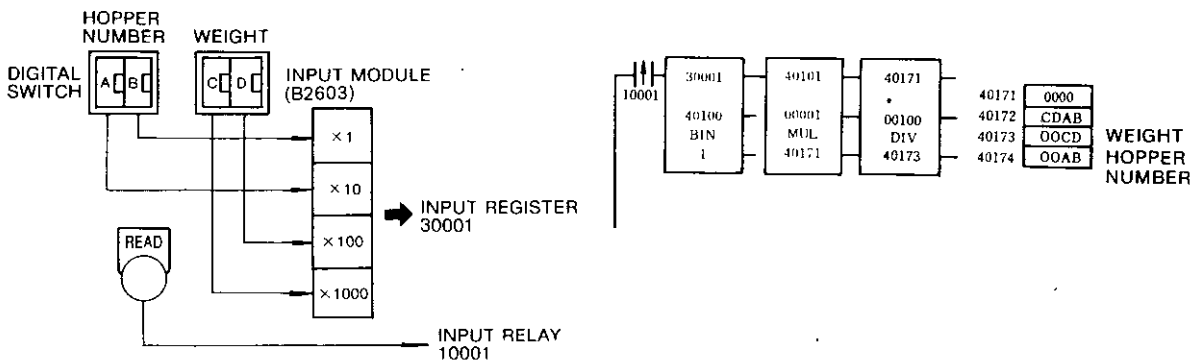
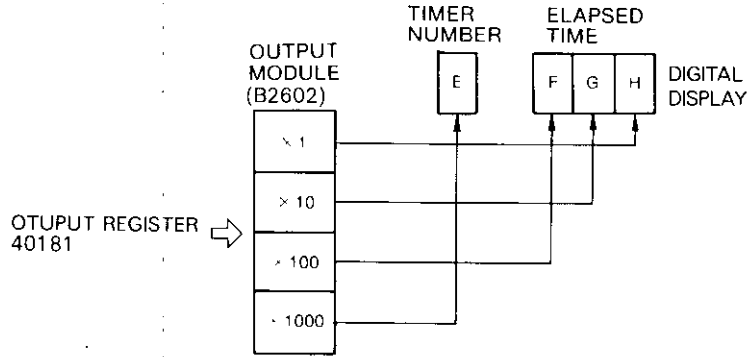


Fig. 5.39 Saving Input Module

② Saving output module

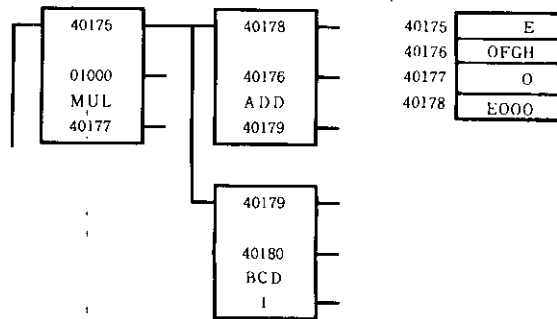
As the contents of 30001 is multiplied by the constant 1, 0 is entered in 40171 and CDAB ($= C \times 1000 + D \times 100 + A \times 10 + B$) in 40172. We use the divide to produce a remainder. By $CDAB \div 100 = CD$ with remainder AB, the weight and hopper number are separated in 40173 and 40174.

(a) Hardware Configuration



NOTE E, F, G and H are 0 to 9 and each digit is output by BCD code.

(b) Ladder Diagram



$$E \times 1000 + FGH = EFGH$$

(Where $0 \leq E \leq 9, 000 \leq FGH \leq 999$)

Fig. 5.40 Having Output Module

5.6 SIGNED ARITHMETIC FUNCTIONS

5.6.1 Types of Signed Arithmetic Functions

The signed arithmetic operations are signed addition, subtraction, multiply, and divide applied on operand V_1 by operand V_2 . There are six variations of arithmetic functions as described in Table 5.33.

Table 5.33 Types of Arithmetic Functions

Type Signed	Symbol	Operator	Range of Operand V_1	Range of Operand V_2	Reference Page
Signed Addition	SADD	$V_1 + V_2$	-9,999 to 9,999		109
Signed Double-precision Addition	SDAD		-99,999,999 to 99,999,999		112
Signed Subtraction	SSUB	$V_1 - V_2$ V_1 compared to V_2	-9,999 to 9,999		115
Signed Double-precision Subtraction	SDSB		-99,999,999 to 99,999,999		117
Signed Multiply	SMUL	$V_1 \times V_2$	-9,999 to 9,999		120
Signed Divide	SDIV	$V_1 \div V_2$	-9,999 to 9,999*	-9,999 to 9,999	122

* The range of V_1 becomes as with when two successive registers are used.

5.6.2 Sign Representation

With GL60H, GL70H, the internal representation of negative numbers is as follows:

- ① When a negative value is input in a decimal number from the programming panel of P150, it is first changed into a positive value and then converted into a binary number.
- ② "1" is placed in the most significant bit (MSB) of the binary number thus converted.
- ③ The numerical value is stored at the specified reference number.

Example 1:

When -100 is input:

When 100 is converted into a binary number,
the result is 0000 0000 0110 0100.

When 1 is placed in the MSB, the result is 1000 0000 0110 0100.

It is registered in a specified register.

Example 2:

When -10,005,000 is to be stored in continuous registers:

40001	-1000
40002	5000

The input low-order data are input without a minus sign.

Similarly, the operation data are also stored as a positive number.

NOTE

1. It is not desirable to use unsigned arithmetic operations together with signed arithmetic operations in one operation system, because it may not allow correct operation due to the difference in structure between the data.
2. When data conversion or other processing is made to an operand or an operator, 1 may be input to the MSB, making it a negative number.

5.6.3 Signed Addition (SADD)

(1) Function

Operates signed addition in 4-digit decimal.

(2) Form

- Fig. 5.41 shows the form of SADD.
- SADD is the symbol denoting the signed addition.
- SADD operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.34, specify any of reference numbers of various registers.

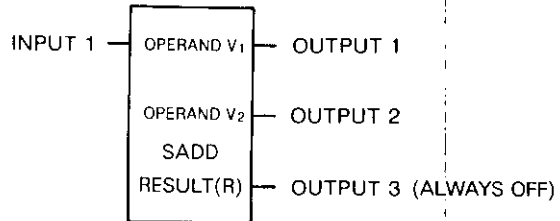


Fig. 5.41 SADD General Form

Table 5.34 Elements of SADD Function

Element Position	Specified Number	Description
Top	Any one of the following: • Input register (30001-30512) • Holding register (40001-49999)	• When register are specified, the contents are the operand ($V_1 = -9,999$ to $9,999$).
Middle	• Constant register (31001-35096) • Link register (R0001-R1024)	• When registers are specified, the contents are the operand ($V_2 = -9,999$ to $9,999$).
Bottom	• Holding register (40001-49999) • Link register (R0001-R1024)	The result of addition function ($V_1 + V_2 = -9,999$ to $9,999$) is stored in $4\times\times\times\times$ or $R\times\times\times\times$.

(3) Operation

- By the addition (SADD), $V_1 + V_2$ is calculated when the input 1 is ON. The result is treated as follows.
 - ① If $0 \leq V_1 + V_2 \leq 9,999$,
 $V_1 + V_2$ is stored in R. All outputs remain OFF.
 - ② If $V_1 + V_2 \geq 10,000$, $V_1 + V_2 - 10,000$ is stored in R. The output 1 remains OFF and the output 2 is turned ON.
 - ③ If $-9,999 \leq V_1 + V_2 \leq -1$,
 $V_1 + V_2$ is stored in R. The output 1 is turned ON and output 2 remains OFF.
 - ④ If $V_1 + V_2 \leq -10,000$,
 $V_1 + V_2 + 10,000$ is stored in R.
The outputs 1 and 2 are turned ON.

5.6.3 Signed Addition (SADD) (Cont'd)

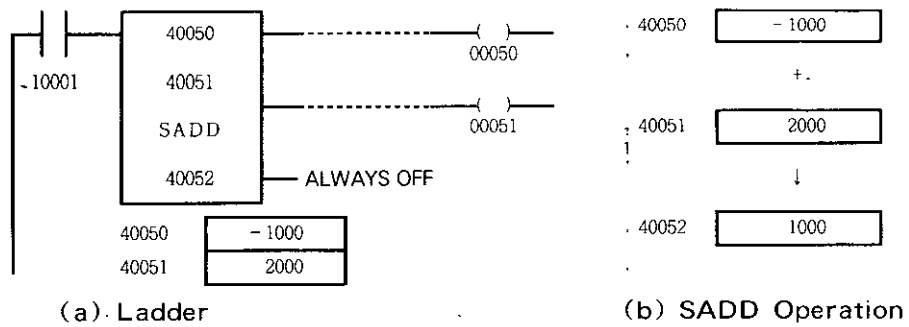
- The output 3 is always OFF.
- The result remains in R even after the input 1 is turned from ON to OFF.
- Table 5.35 shows an addition operation (SADD).

Table 5.35 SADD Operation

Input 1	Condition	Operation	Output 1	
ON	$0 \leq V1 + V2 \leq 9,999$	$V1 + V2 \rightarrow R$	OFF	OFF
	$10,000 \leq V1 + V2$	$V1 + V2 - 10,000 \rightarrow R$	OFF	ON
	$-9,999 \leq V1 + V2 \leq -1$	$V1 + V2 \rightarrow R$	ON	OFF
	$V1 + V2 \leq -10,000$	$V1 + V2 + 10,000 \rightarrow R$	ON	ON
OFF	None	Not operated.	OFF	OFF

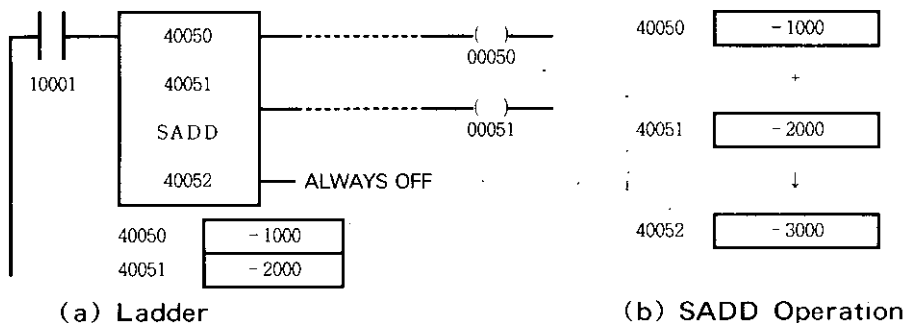
(4) Example

Example 1:



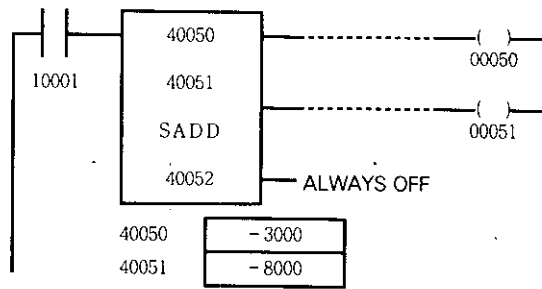
SADD in (a) executes the operation of (b) when input relay 10001 is ON. The outputs 1 and 2 remain OFF. The result remains in 40052 even after input relay 10001 is turned OFF.

Example 2:

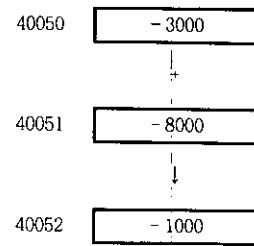


SADD shown in the Fig. (a) executes the operation shown in the Fig. (b) when input relay 10001 is in the ON state. It turns output 1 or coil 00050 to ON. Output 2 remains in the OFF state. Even when input relay 10001 turns from ON to OFF, the operation result remains in 40052. The data stored in 40052 are shown in the internal representation as 1000 1011 1011 1000.

Example 3:



(a) Ladder



(b) SADD Operation

SADD in (a) executes the operation of (b) when input relay 10001 is ON. The outputs 1 and 2 are turned ON. Thus the coils 00050 and 00051 are turned ON. The result remains in 40052 even after input relay 10001 is turned OFF.

5.6.4 Signed Double-Precision Addition (SDAD)

(1) Function

Operates signed double-precision addition in 8-digit decimal.

(2) Form

- Fig. 5.42 shows the form of signed double-precision addition (SDAD).
- SDAD is the symbol denoting the double-precision addition.
- SDAD operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.36, specify any of reference numbers of various registers.

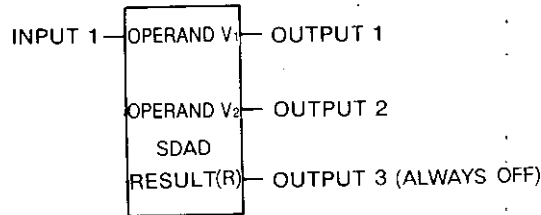


Fig. 5.42 SDAD General Form

Table 5.36 Elements of SDAD Function

Element Position	Specified Number	Description
Top	Either one of the following: • Input register (30001-30511) • Holding register (40001-49999) • Constant register (31001-35095) • Link register (R0001-R1023)	Operand ($V_1 = -99,999,999$ to $99,999,999$) is stored as follows. $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 \\ \boxed{V_1H} & \boxed{V_1L} \end{array}$ V_1H : Higher-place 4 digits of V_1 V_1L : Lower-place 4 digits of V_1
Middle		Operand ($V_2 = -99,999,999$ to $99,999,999$) is stored as follows. $\begin{array}{cc} \times \times \times \times \times & \times \times \times \times \times + 1 \\ \boxed{V_2H} & \boxed{V_2L} \end{array}$ V_2H : Higher-place 4 digits of V_2 V_2L : Lower-place 4 digits of V_2
Bottom	• Holding register (40001-49998) • Link register (R0001-R1023)	Result of operation ($V_1 + V_2 = -99,999,999$ to $99,999,999$) is stored as follows. Example, $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 \\ \boxed{(V_1 + V_2)H} & \boxed{(V_1 + V_2)L} \end{array}$ $(V_1 + V_2)H$: Higher-place 4 digits of $(V_1 + V_2)$ $(V_1 + V_2)L$: Lower-place 4 digits of $(V_1 + V_2)$

(3) Operation

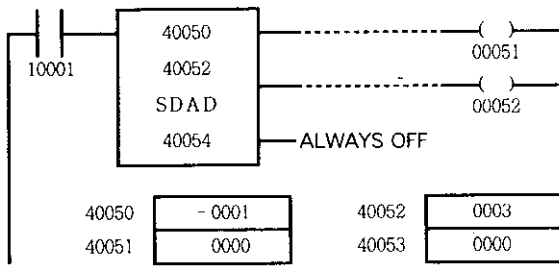
- By the SDAD, $V_1 + V_2$ is calculated when the input 1 is ON. The result is treated as follows.
 - ① If $0 \leq V_1 + V_2 \leq 99,999,999$,
 $V_1 + V_2$ is stored in R and R+1. All outputs remain OFF.
 - ② If $V_1 + V_2 \leq 100,000,000$,
 $V_1 + V_2 - 100,000,000$ is stored in R and R+1.
 The output 1 remains OFF and the output 2 is turned ON.
 - ③ If $-99,999,999 \leq V_1 + V_2 \leq -1$,
 $V_1 + V_2$ is stored in R and R+1.
 The output 1 is turned ON and the output 2 remains OFF.
 - ④ If $V_1 + V_2 \leq -100,000,000$,
 $V_1 + V_2 + 100,000,000$ is stored in R and R+1.
 The outputs 1 and 2 are turned ON.
- The output 3 is always OFF.
- The result remains in R and R+1 even after the input 1 is turned from ON to OFF.
- Table 5.37 shows the SDAD.

Table 5.37 SDAD Operation

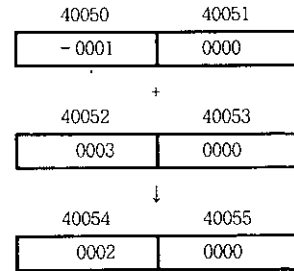
Input 1	Condition	Operation	Output 1	Output 2
ON	$0 \leq V_1 + V_2 \leq 99,999,999$	$V_1 + V_2 \rightarrow R$	OFF	OFF
	$100,000,000 \leq V_1 + V_2$	$V_1 + V_2 - 100,000,000 \rightarrow R$	OFF	ON
	$-99,999,999 \leq V_1 + V_2 \leq -1$	$V_1 + V_2 \rightarrow R$	ON	OFF
	$V_1 + V_2 \leq -100,000,000$	$V_1 + V_2 + 100,000,000 \rightarrow R$	ON	ON
OFF	None	Not operated.	OFF	OFF

(4) Example

Example 1:



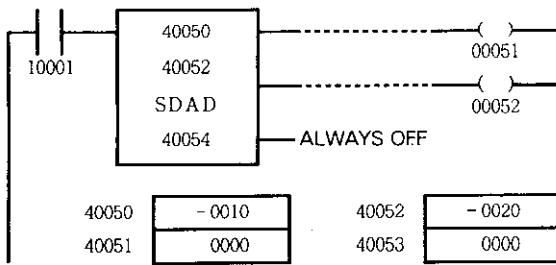
(a) Ladder



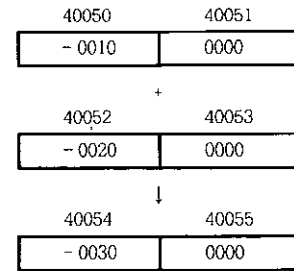
(b) SDAD Operation

SDAD in (a) executes the operation of (b) when input relay 10001 is ON. The outputs 1 and 2 remain OFF. The results remain in 40054 and 40055 even after input relay 10001 is turned OFF.

Example 2:



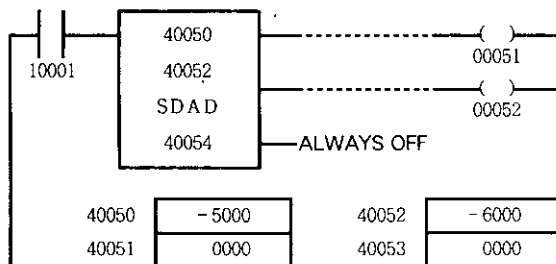
(a) Ladder



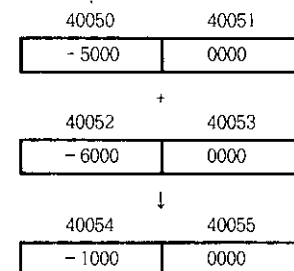
(b) SDAD Operation

SDAD in (a) executes the operation of (b) when input relay 10001 is ON. The output 2 remains OFF. The output 1 (coil 00051) is turned ON. The result remains in 40054 and 40055 even after input relay 10001 is turned OFF.

Example 3:



(a) Ladder



(b) SDAD Operation

SDAD in (a) executes the operation of (b) when input relay 10001 is ON. The outputs 1 and 2 are ON. Accordingly, the coils 00051 and 00052 are ON the output 3 remains OFF. The result remains in 40054 and 40055 even after input relay 10001 is turned OFF.

5.6.5 Signed Subtraction (SSUB)

(1) Function

Operates signed subtraction in 4-digit decimal.

(2) Form

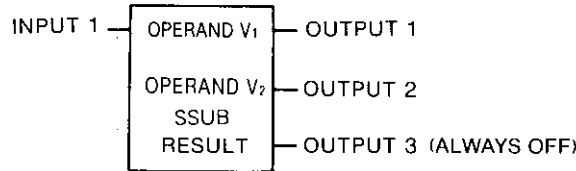


Fig. 5.43 SSUB General Form

- Fig. 5.43 shows the form of signed subtraction (SSUB).
- SSUB is the symbol denoting the signed subtraction.
- SSUB operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.38, specify any of the reference numbers for each of the top, middle and bottom elements.

Table 5.38 Elements of SSUB Function

Element Position	Specified Number	Description
Top	Any one of the following: • Input register (30001-30512) • Holding register (40001-49999)	• When the registers are specified, the contents are the operand ($V_1 = -9,999$ to $9,999$).
Middle	• Constant register (31001-35096) • Link register (R0001-R1024)	• When the registers are specified, the contents are the operand ($V_2 = -9,999$ to $9,999$).
Bottom	• Holding register (40001-49999) • Link register (R0001-R1024)	The result of SSUB function ($ V_1 - V_2 = -9,999$ to $9,999$) is stored in $4 \times \times \times \times$ or $R \times \times \times \times$.

(3) Operation

- By the SSUB, $V_1 - V_2$ will be calculated when the input 1 is ON. The result is treated as follows.
 - ① If $0 \leq V_1 - V_2 \leq 9,999$,
 $V_1 - V_2$ is stored in R and all the outputs remain OFF.
 - ② If $V_1 - V_2 \geq 10,000$,
 $V_1 - V_2 - 10,000$ is stored in R, the output 1 remains OFF and the output 2 is turned on.
 - ③ If $-9,999 \leq V_1 - V_2 \leq -1$,
 $V_1 - V_2$ is stored in R, the output 1 is turned ON and the output 2 remains OFF.
 - ④ If $V_1 - V_2 \leq -10,000$,
 $V_1 - V_2 + 10,000$ is stored in R, and both of the outputs 1 and 2 are turned ON.

5.6.5 Signed Subtraction (SSUB) (Cont'd)

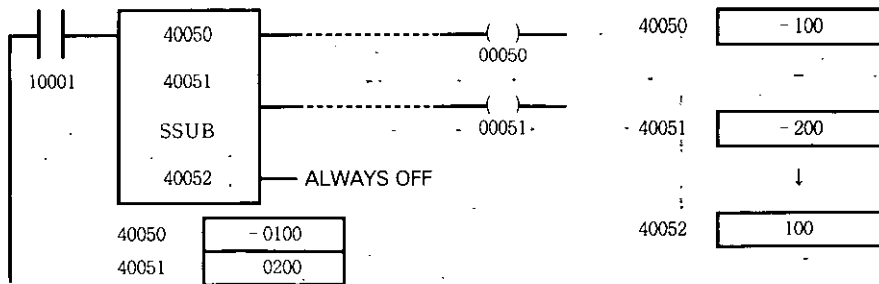
- The output 3 is always OFF.
- The result remains in even after the input 1 is turned from ON to OFF.
- Table 5.39 shows a SSUB operation.

Table 5.39 SSUB Operation

Input 1	Condition	Operation	Output 1	Output 2
ON	$0 \leq V1 - V2 \leq 9,999$	$V1 - V2 \rightarrow R$	OFF	OFF
	$10,000 \leq V1 - V2$	$V1 - V2 - 10,000 \rightarrow R$	OFF	ON
	$-9,999 \leq V1 - V2 \leq -1$	$V1 - V2 \rightarrow R$	ON	OFF
	$V1 - V2 \leq -10,000$	$V1 - V2 + 10,000 \rightarrow R$	ON	ON
OFF	None	Not operated	OFF	OFF

(4) Example

Example 1:

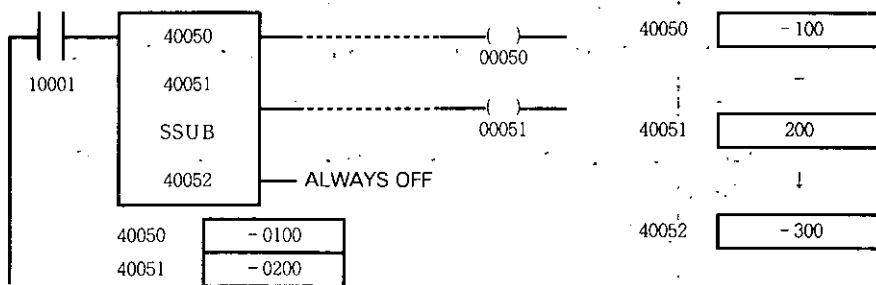


(a) Ladder

(b) SSUB Operation

SSUB in (a) executes the operation of (b) when input relay 10001 is ON. The outputs 1 and 2 remain OFF. The result remains in 40052 even after input relay 10001 is turned OFF:

Example 2:

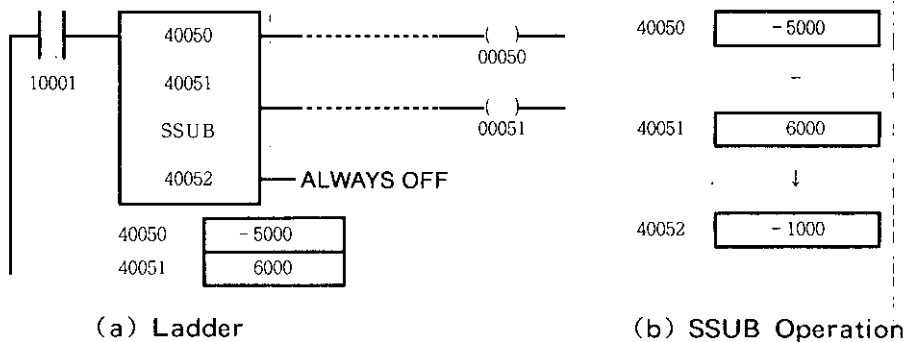


(a) Ladder

(b) SSUB Operation

SSUB in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 1 (coil 00050) is turned ON. The output 2 remains OFF. The result remains in 40052 even after input relay 10001 is turned OFF.

Example 3:



SSUB in (a) executes the operation of (b) when input relay 10001 is ON. Only the outputs 1 and 2 (coils 00050 and 00051) is turned ON. The result remains in 40062 even after input relay 10001 is turned OFF.

5.6.6 Signed Double-precision Subtraction (SDSB)

(1) Function

Operates signed double-precision subtraction in 8-digit decimal.

(2) Form

- Fig. 5.44 shows the form of signed double-precision subtraction (SDSB).

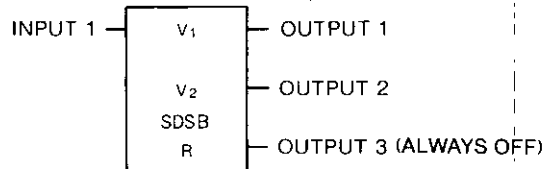


Fig. 5.44 SDSB General Form

- SDSB is the symbol denoting the signed double-precision subtraction.
- SDSB operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.40, specify either reference number for each of the top, middle and bottom elements.

Table 5.40 Elements SDSB Function

Element Position	Specified Number	Description
Top	Either one of the following: • Input register (30001-30511) • Holding register (40001-49998) • Constant register (31001-35095) • Link register (R0001-R1023)	Operand ($V_1 = -99,999,999$ to $99,999,999$) is stored as follows. $\begin{array}{cc} \times \times \times \times \times & \times \times \times \times \times + 1 \\ \boxed{V_1H} & \boxed{V_1L} \end{array}$ V_1H : Higher-place 4 digits of V_1 . V_1L : Lower-place 4 digits of V_1 .
Middle	Same as Top	Operand ($V_2 = -99,999,999$ to $99,999,999$) is stored as follows. $\begin{array}{cc} \times \times \times \times \times & \times \times \times \times \times + 1 \\ \boxed{V_2H} & \boxed{V_2L} \end{array}$ V_2H : Higher-place 4 digits of V_2 . V_2L : Lower-place 4 digits of V_2 .
Bottom	• Holding register (40001-49998) • Link register (R0001-R1023)	Result of operation ($-99,999,999$ to $99,999,999$) is stored as follows. $\begin{array}{cc} \times \times \times \times \times & \times \times \times \times \times + 1 \\ \boxed{(V_1 - V_2)H} & \boxed{(V_1 - V_2)L} \end{array}$ $(V_1 - V_2)H$: Higher-place 4 digits of $(V_1 - V_2)H$. $(V_1 - V_2)L$: Lower-place 4 digits of $(V_1 - V_2)L$.

(3) Operation

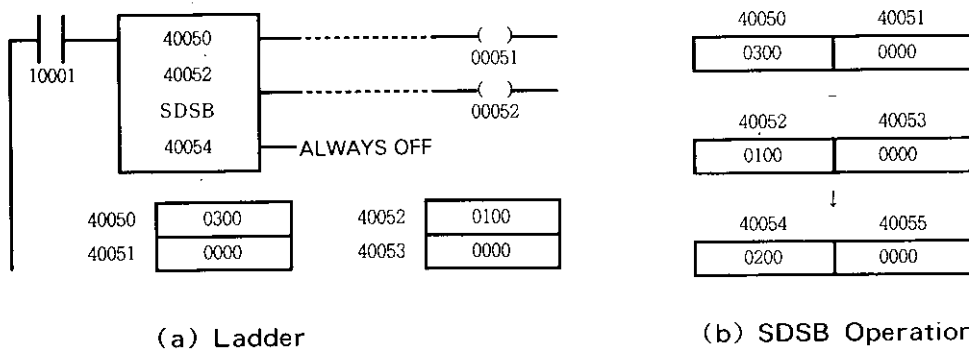
- By the SDSB, $V_1 - V_2$ will be calculated when the input 1 is ON. The result is treated as input 1 is ON. The result is
 - ① If $0 \leq V_1 - V_2 \leq 99,999,999$,
 $V_1 - V_2$ is stored in R, and all outputs remain OFF.
 - ② If $V_1 - V_2 \geq 100,000,000$,
 $V_1 - V_2 - 100,000,000$ is stored in R. The output 1 remains OFF and the output 2 is turned ON.
 - ③ If $-99,999,999 \leq V_1 - V_2 \leq -1$,
 $V_1 - V_2$ is stored in R. The output 1 is turned on and the output 2 remains OFF.
 - ④ If $V_1 - V_2 \leq -100,000,000$,
 $V_1 - V_2 + 100,000,000$ is stored in R and the outputs 1 and 2 are turned ON.
- The output 3 is always OFF.
- The result remains in R even after the input 1 is turned from ON to OFF.
- Table 5.41 shows SDSB operation.

Table 5.41 SDSB Operation

Input 1	Condition	Operation	Output 1	Output 2
ON	$0 \leq V_1 - V_2 \leq 99,999,999$	$V_1 - V_2 \rightarrow R$	OFF	OFF
	$100,000,000 \leq V_1 - V_2$	$V_1 - V_2 - 100,000,000 \rightarrow R$	OFF	ON
	$-99,999,999 \leq V_1 - V_2 \leq -1$	$V_1 - V_2 \rightarrow R$	ON	OFF
	$V_1 - V_2 \leq -100,000,000$	$V_1 - V_2 + 100,000,000 \rightarrow R$	ON	ON
OFF	—	Not operated.	OFF	OFF

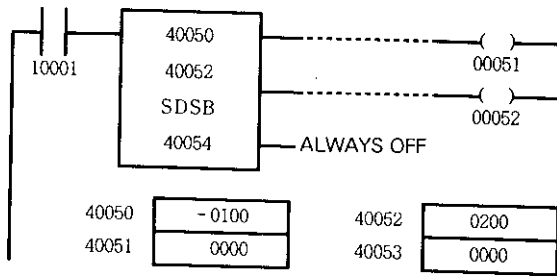
(4) Example

Example 1:



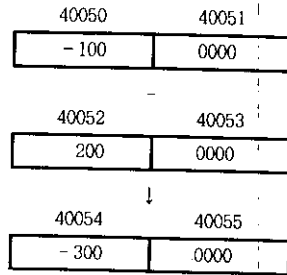
SDSB in (a) executes the operation of (b) when input relay 10001 is ON. The outputs 1 and 2 remain OFF. The result remains in 40054 and 40055 even after input relay 10001 is turned OFF.

Example 2:



(a) Ladder

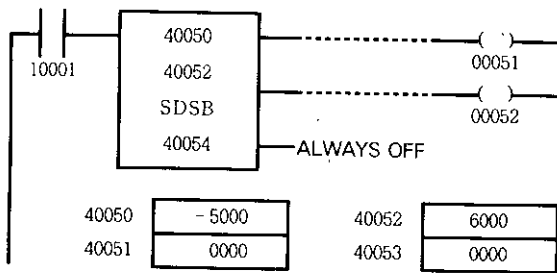
40050	- 0100	40052	0200
40051	0000	40053	0000



(b) SDSB Operation

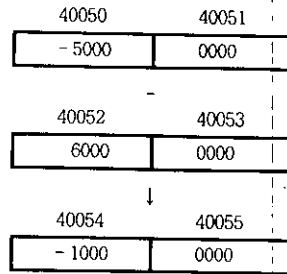
SDSB in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 1 (coil 00051) is turned ON. The output 2 remains OFF. The result remains in 40054 and 40055 even after input relay 10001 is turned OFF.

Example 3:



(a) Ladder

40050	- 5000	40052	6000
40051	0000	40053	0000



(b) SDSB Operation

SDSB in (a) executes the operation of (b) when input relay 10001 is ON. The outputs 1 and 2 (coils 00051 and 00052) are turned ON. The output 3 remains OFF. The result remains in 40054 and 40055 even after input relay 10001 is turned OFF.

5.6.7 Signed Multiply (SMUL)

(1) Function

Operates signed multiply is 4-digit decimal.

(2) Form

- Fig. 5.45 shows the form of signed multiply (SMUL).

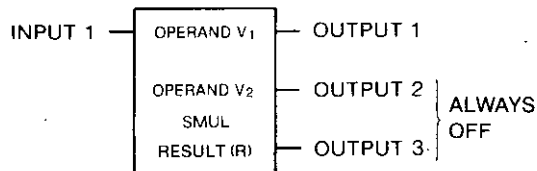


Fig. 5.45 SMUL General Form

- Multiply operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.42, specify any of reference number for each of the top, middle and bottom elements.
- SMUL is the symbol denoting the signed multiply.

Table 5.42 Elements of SMUL Function

Element Position	Specified Number	Description
Top	Any one of the following: • Input register (30001-30512) • Holding register (40001-49999)	• When registers are specified, the contents are the operand ($V_1 = -9,999$ to $9,999$).
Middle	• Constant register (31001-35096) • Link register (R0001-R1024)	• When registers are specified, the contents are the operand ($V_2 = -9,999$ to $9,999$).
Bottom	• Holding register (40001-49998) • Link register (R0001-R1023)	The result of SMUL function ($V_1 \times V_2 = -99,980,001$ to $99,980,001$) is stored as follows: $\begin{array}{ccc} \times \times \times \times & , & \times \times \times \times + 1 \\ \boxed{(V_1 \times V_2)_H} & & \boxed{(V_1 \times V_2)_L} \end{array}$ ($V_1 \times V_2)_H$: Higher-place 4 digits of ($V_1 \times V_2$) ($V_1 \times V_2)_L$: Lower-place 4 digits of ($V_1 \times V_2$)

(3) Operation

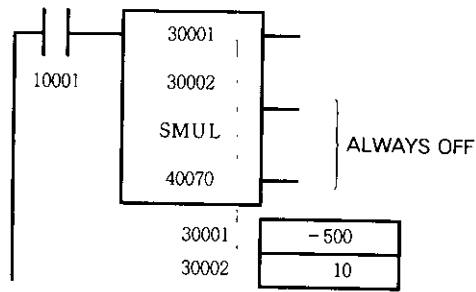
- By the SMUL, $V_1 \times V_2$ is calculated when the input 1 is ON. The result is treated as follows. The four higher-place digits of $V_1 \times V_2$ are stored in R and the four lower-place digits in R + 1. When the result is negative, the output 1 is ON. When the result is positive, the output remains OFF.
- The outputs 2 and 3 are always OFF.
- Table 5.43 shows SMUL operation.

Table 5.43 SMUL Operation

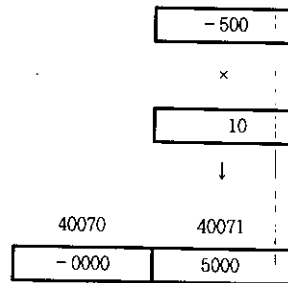
Input 1	Condition	Operation	Result	Output 1
ON	None	$V1 \times V2 \rightarrow R$ (Higher-place 4 digits)	Positive	OFF
		$R + 1$ (Lower-place 4 digits)	Negative	ON
OFF	None	Not provided.	-	OFF

(4) Example

Example 1:



(a) Ladder



(b) SMUL Operation

SMUL in (a) executes the operation of (b) when input relay 10001 is ON. The output 1 is turned ON. The result remains in 40070 and 40071 even after input relay 10001 is turned OFF.

5

5.6.8 Signed Divide (SDIV).

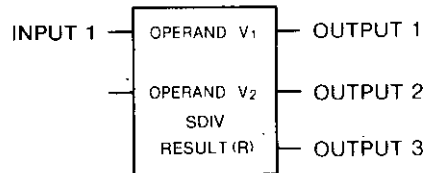
(1) Function

Operates signed divide in 8-digit decimal.

(2) Form

- Fig. 5.46 shows the form of signed divide (SDIV).

Fig. 5.46 SDIV General Form



- SDIV is the symbol denoting the signed divide.
- SDIV operation requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.44, specify any of reference numbers for each of the top, middle and bottom elements.

Table 5.44 Elements of SDIV Function

Element Position	Specified Number	Description
Top	Any one of the following: • Input register (30001-30511) • Holding register (40001-49998) • Constant register (31001-35095) • Link register (R0001-R1023)	• The operand ($V_1 = -99,989,999$ to $99,989,999$) is stored as follows. $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 \\ \boxed{V_1H} & \boxed{V_1L} \end{array}$ V_1H : Higher-place 4 digits of V_1 V_1L : Lower-place 4 digits of V_1
Middle	Any one of the following: • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024)	• When registers are specified, the contents are the operand ($V_2 = 1$ to $9,999$).
Bottom	• Holding register (40001-49998) • Link register (R0001-R1023)	Result of divide function ($V_1 \div V_2$) is stored as follows. Where input 2 is OFF, Where input 2 is ON, $\begin{array}{cccc} \times \times \times \times & \times \times \times \times + 1 & \times \times \times \times & \times \times \times \times + 1 \\ \boxed{} & \boxed{} & \boxed{} & \boxed{} \end{array}$ The quotient The remainder The integer quotient The decimal quotient

(3) Operation

By the SDIV, $V_1 \div V_2$ is calculated when the input 1 is ON. The result is treated as follows.

(a) If the input 2 is OFF,

The quotient of $V_1 \div V_2$ is stored in R and the remainder in R + 1. When it is negative, the output 1 is turned ON. When it is positive, all outputs are OFF.

(b) If the input 2 is ON,

The integer part of the quotient of $V_1 \div V_2$ is stored in R and the decimal part (rounded off to the fifth decimal place) in R + 1. When it is negative, the output 1 is turned ON. When it is positive, all outputs are OFF.

(c) The result remains in R and R + 1 even after the input 1 is turned from ON to OFF.

(d) In the following cases, divide operation is not executed and zero is placed in each of R and R + 1.

- When $V_2 = 0$, the output 3 is turned ON.
- If the quotient or the integer part of quotient overflows in R, the output 2 is turned ON.

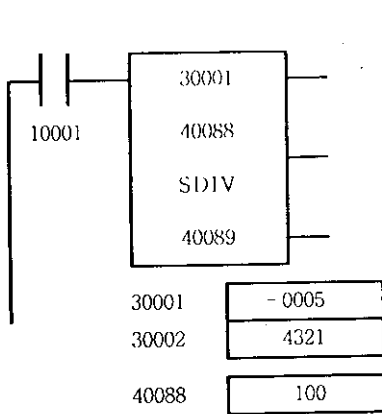
Tables 5.45 shows SDIV operation.

Table 5.45 SDIV Operation

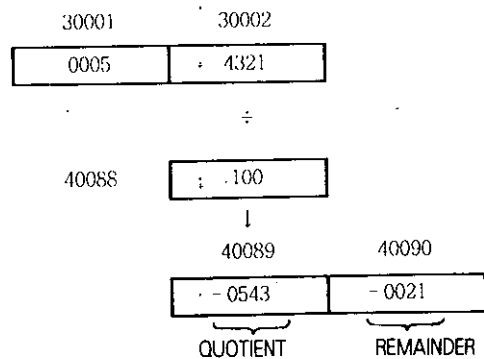
Input 1	Input 2	Condition	Operation	Result	Output 1	Output 2	Output 3
ON	OFF	$V_2 \neq 0,$ $V_{1H} < V_2$	$(V_{1H} \times 10,000 + V_{1L}) \div V_2$ Quotient \rightarrow R Remainder R + 1	+	OFF	OFF	OFF
				-	ON	OFF	OFF
		$V_2 \neq 0,$ $V_{1H} \geq V_2$	$0 \rightarrow$ R R + 1		OFF	ON	OFF
		$V_2 = 0$	$0 \rightarrow$ R R + 1		OFF	OFF	ON
ON	ON	$V_2 \neq 0,$ $V_{1H} < V_2$	$(V_{1H} \times 10,000 + V_{1L}) \div V_2$ Integer quotient \rightarrow R Decimal quotient R + 1	+	OFF	OFF	OFF
				-	ON	OFF	OFF
		$V_2 \neq 0,$ $V_{1H} \geq V_2$	$0 \rightarrow$ R R + 1		OFF	ON	OFF
		$V_2 = 0$	$0 \rightarrow$ R R + 1		OFF	OFF	ON
OFF	ON OFF	None	Not operated.		OFF	OFF	OFF

(4) Example

Example 1:



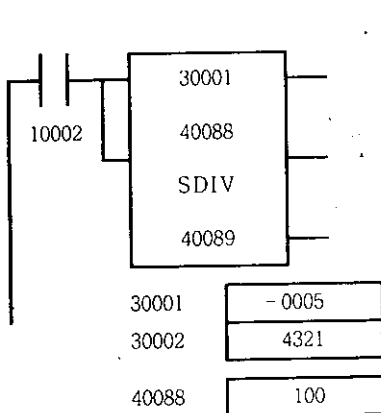
(a) Ladder



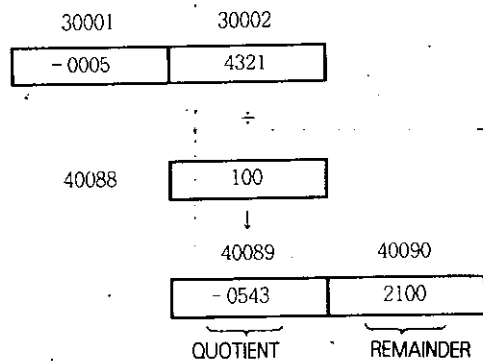
(b) SDIV Operation

SDIV in (a) executes the operation of (b) when input relay 10001 is ON. Only the output 1 is turned ON. The result remains in 40089 and 40090 even after input relay 10001 is turned OFF.

Example 2:



(a) Ladder



(b) SDIV Operation

SDIV in (a) executes the operation of (b) when input relay 10002 is ON. Only the output 1 is turned ON. The result remains in 40089 and 40090 even after input relay 10002 is turned OFF.

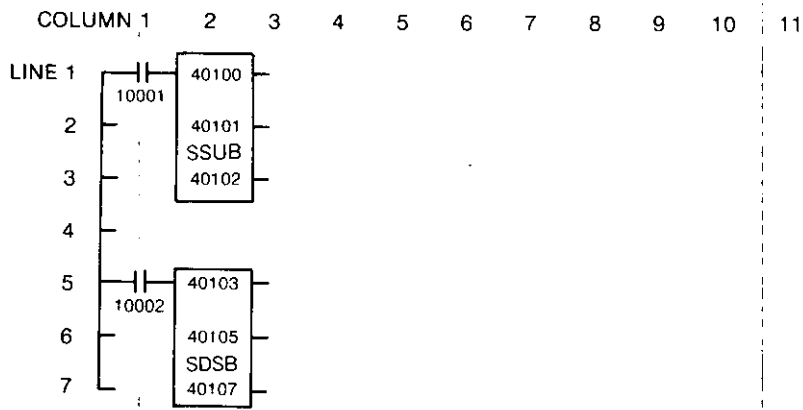
5.6.9 Programming Signed Arithmetic Logic and Precautions

In all signed arithmetic operations, add, subtraction and multiply function require only input 1, and divide requires only inputs 1 and 2. But the programming panel gives display as each output element line which may be connected to each input element line.

(1) Programming Arithmetic Logic

All signed arithmetic operations require three elements placed vertically (top, middle, and bottom) in a network. They can be used at any intersection of the 7 lines-by-10 columns matrix, however the top element (operand) cannot be used on either line 6 nor line 7.

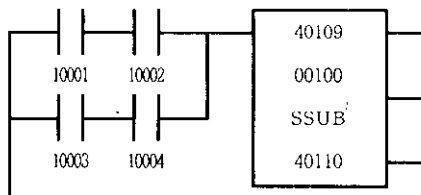
Example:



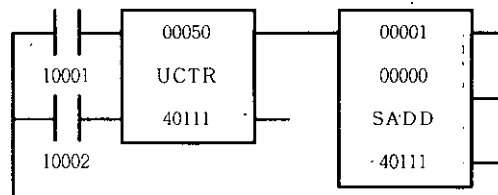
(2) Inputs of Signed Arithmetic Logic

Inputs to the signed arithmetic logic may be outputs of relays, timers, counters, data processing circuits and other arithmetic operations.

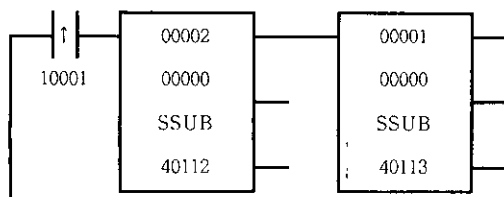
Example 1:



Example 2:



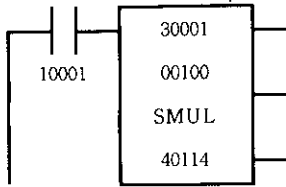
Example 3:



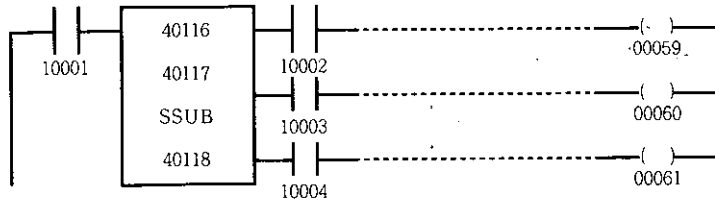
(3) Outputs of Signed Arithmetic Logic

Coils need not be connected to three output nodes (1, 2, and 3) of signed arithmetic function. A relay contact may be connected to the output nodes on the right or connect the output nodes directly to an input node of an arithmetic circuit, except relays.

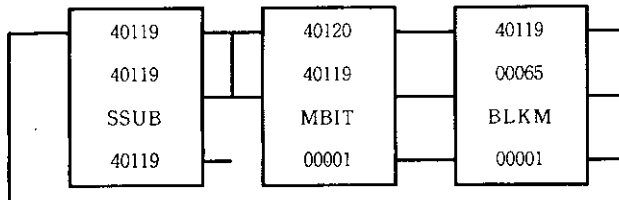
Example 1:



Example 2:

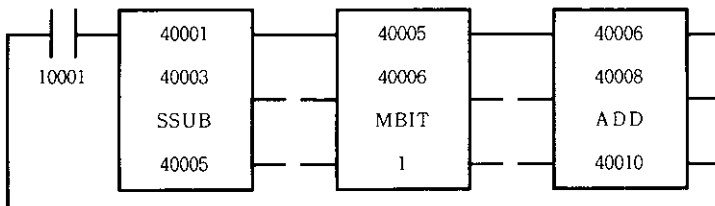


Example 3:



(4) Data Handling

In case the operation result of a signed arithmetic operation is used in an arithmetic operation, first obtain the absolute value of the operation result using "MBIT" as shown below, before using it in the next operation. When the operation is executed with the negative number remaining as is, no correct result can be obtained. However, there will be no problem, even if an unsigned operation result is used in a signed arithmetic operation.

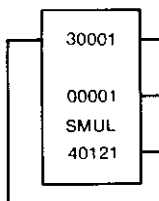


"1" in the MSB is cleared by MBIT.

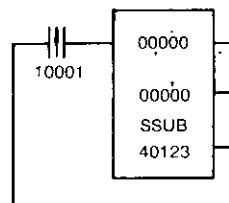
(5) Execution of Signed Arithmetic Operations (Only One Scanning Cycle)

To execute a constant arithmetic operation, connect the inputs directly to the power rail on the left. To execute it only in one scan, use a transitional contact as an input.

Example 1:



Example 2:



5.7 SQUARE ROOT

5.7.1 Types of Square Root

Square root is the function that calculates the square root of the operand V which is a 4- or 8-digit decimal number. Two types of square root are available as shown in Table 5.46

Table 5.46 Types of Square Root Function

Type	Symbol	Function	Range of Operand V	Reference Page
Square Root	SQRT	Calculation of V	0 to 9,999	121
Double-precision Square Root	DSQR		0 to 99,999,999	123

5.7.2 Square Root (SQRT)

(1) Function

Operates square root in 4-digit decimal up to 4 places of decimal.

(2) Form

- Fig. 5.47 shows the form of square root (SQRT).

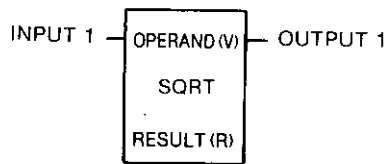


Fig. 5.47 SQRT General Form

- SQRT is the symbol denoting the square root.
- Square root operation requires two elements placed vertically (top and bottom). Referring to Table 5.47, specify any of reference numbers of various registers for each of the elements.

Table 5.47 Elements of Square Root

Element Position	Specified Number	Description
Top	Either one of the following: <ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) 	The contents of specified registers are the operand (V = 0 to 9,999).
Bottom	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023) 	The result of square root operation (\sqrt{V}) is stored as follows. $\begin{array}{c} \times \times \times \times \\ \boxed{} \end{array} \quad \begin{array}{c} \times \times \times \times + 1 \\ \boxed{} \end{array}$ Integer part of \sqrt{V} . Decimal part of \sqrt{V}

(3) Operation

By the square root (SQRT), \sqrt{V} is calculated when the input 1 is ON. The result is treated as follows. The integer part of \sqrt{V} is stored in R and the decimal part (rounded off at the fifth decimal place) in R + 1. The output 1 is turned ON. The result remains in R and R + 1 even after the input 1 is turned from ON to OFF.

Table 5.48 shows a square root operation (SQRT).

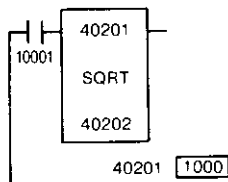
Table 5.48 SQRT Operation

Input 1	Operation	Output 1
ON	$\sqrt{V} \rightarrow R$ (Integer part), R + 1 (Decimal part)*	ON
OFF	Not operated.	OFF

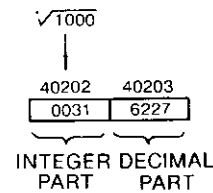
*Rounded off at the 5th decimal place.

(4) Example

Example:



(a) Ladder



(b) SQRT Operation

SQRT in (a) executes the operation of (b) when input relay 10001 is ON. The output 1 is turned ON. The result remains in 40202 and 40203 even after input relay 10001 is turned OFF.

5.7.3 Double-precision Square Root (DSQR)

(1) Function

Operates double-precision square root in 8-digit decimal up to 4 places of decimal.

(2) Form

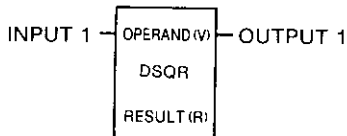


Fig. 5.48 DSQR General Form

- Fig. 5.48 shows the form of double-precision square root (DSQR).
- DSQR is the symbol denoting the double-precision square root.
- Double-precision square root operation requires two elements placed vertically (top and bottom). Referring to Table 5.49, specify either register reference number for each of the elements.

Table 5.49 Elements of DSQR Function

Element Position	Specified Number	Description								
Top	Either one of the following: <ul style="list-style-type: none"> • Input register (30001-30511) • Holding register (40001-49998) • Constant register (31001-35095) • Link register (R0001-R1023) 	The operand ($V=0$ to 99,999,999) is stored as follows. <table style="margin-left: auto; margin-right: auto; border-collapse: collapse;"> <tr> <td style="text-align: center;">× × × × ×</td> <td style="text-align: center;">× × × × × + 1</td> </tr> <tr> <td style="text-align: center;"> <table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="text-align: center;">V_H</td></tr> </table> </td> <td style="text-align: center;"> <table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="text-align: center;">V_L</td></tr> </table> </td> </tr> <tr> <td colspan="2" style="text-align: center;"> V_H: Higher-place 4 decimal digits of V V_L: Lower-place 4 decimal digits of V </td> </tr> </table>	× × × × ×	× × × × × + 1	<table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="text-align: center;">V_H</td></tr> </table>	V_H	<table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="text-align: center;">V_L</td></tr> </table>	V_L	V_H : Higher-place 4 decimal digits of V V_L : Lower-place 4 decimal digits of V	
× × × × ×	× × × × × + 1									
<table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="text-align: center;">V_H</td></tr> </table>	V_H	<table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="text-align: center;">V_L</td></tr> </table>	V_L							
V_H										
V_L										
V_H : Higher-place 4 decimal digits of V V_L : Lower-place 4 decimal digits of V										
Bottom	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023) 	The result of double-precision square root operation (\sqrt{V}) is stored as follows. <table style="margin-left: auto; margin-right: auto; border-collapse: collapse;"> <tr> <td style="text-align: center;">× × × × ×</td> <td style="text-align: center;">× × × × × + 1</td> </tr> <tr> <td style="text-align: center;"> <table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="width: 100%;"></td></tr> </table> </td> <td style="text-align: center;"> <table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="width: 100%;"></td></tr> </table> </td> </tr> <tr> <td style="text-align: center;">Integer part of \sqrt{V}</td> <td style="text-align: center;">Decimal part of \sqrt{V}</td> </tr> </table>	× × × × ×	× × × × × + 1	<table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="width: 100%;"></td></tr> </table>		<table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="width: 100%;"></td></tr> </table>		Integer part of \sqrt{V}	Decimal part of \sqrt{V}
× × × × ×	× × × × × + 1									
<table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="width: 100%;"></td></tr> </table>		<table border="1" style="display: inline-table; width: 60px; height: 15px;"> <tr><td style="width: 100%;"></td></tr> </table>								
Integer part of \sqrt{V}	Decimal part of \sqrt{V}									

(3) Operation

- By double-precision square root (DSQR), \sqrt{V} is calculated when the input is ON. The result is treated as follows. The integer part of \sqrt{V} is stored in R and the decimal part (rounded off at the fifth decimal place) in R + 1. The output 1 is turned ON.
- The result remains in R and R + 1 even after the input 1 is turned from ON to OFF.
- Table 5.50 shows a double-precision square root (DSQR).

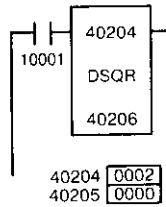
Table 5.50 Double-precision Square Root Operation

Input 1	Operation	Output 1
ON	$\sqrt{V} \rightarrow R$ (Integer part), R+1 (Decimal part)*	ON
OFF	Not operated.	OFF

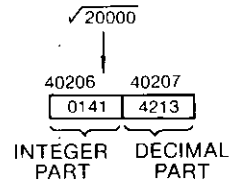
* Rounded off at the 5th decimal place.

(4) Example

Example:



(a) Ladder



(b) DSQR Operation

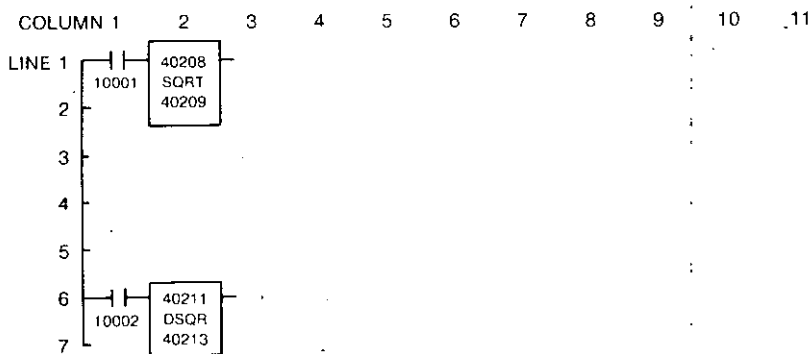
DSQR in (a) executes the operation of (b) when input relay 10001 is ON. The output 1 is turned ON. The result remains in 40206 and 40207 even after input relay 10001 is turned OFF.

5.7.4 Programming Square Root Circuit and Precautions

(1) Programming Square Root Circuit

Square root operation requires two elements placed vertically (top and bottom). It can be used at any intersection of the 7 lines-by-10 columns matrix, but the top element (operand) cannot be used on line 7.

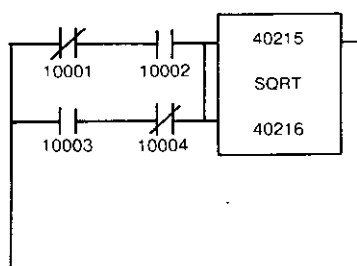
Example:



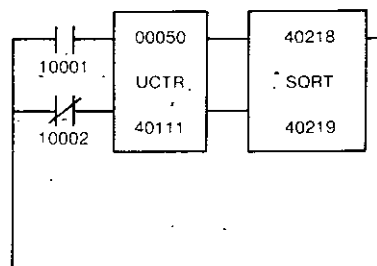
(2) Input of Square Root

Input to square root may be the output of other square roots, relays, timers, counters, or data processing circuits.

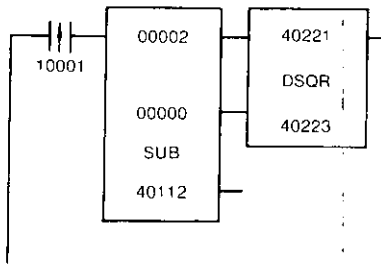
Example 1:



Example 2:



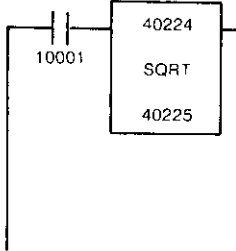
Example 3:



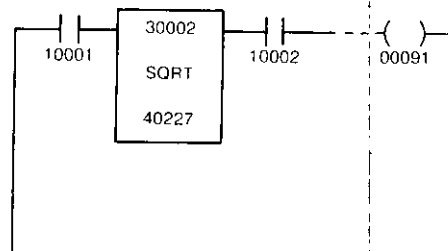
(3) Outputs of Square Root

Coils need not be connected to two output nodes (1 and 2) of a square root. It is permitted to connect a relay contact to the output node at right or connect the output nodes directly to an input node of an arithmetic circuit, except relays.

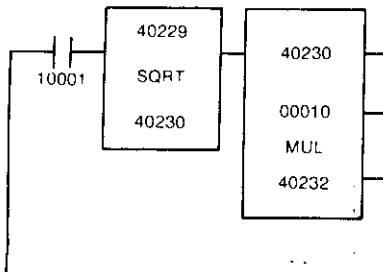
Example 1:



Example 2:



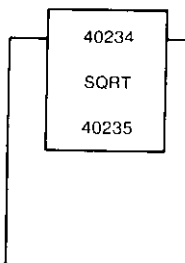
Example 3:



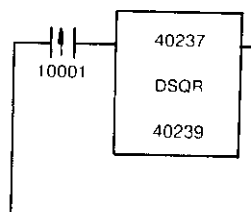
(4) Execution of Square Root (Only 1 Scan Cycle)

To execute a square root constantly, connect the inputs directly to the power rail at left. To execute it only during one scanning cycle, use a transitional contact as an input.

Example 1:



Example 2:



5.8 TRIGONOMETRIC FUNCTION

5.8.1 Types of Trigonometric Function Operations

Trigonometric function operation is intended to obtain the sine and cosine values of operand V down to the 4th decimal place in the range from 0° to 360°. There are two types of trigonometric function operations as shown in Table 5.51.

Table 5.51 Types of Trigonometric Function Operations

Type	Symbol	Content of Operation	Range of Operand V	Reference page
Sine	SIN	Calculation of SIN(V)	0.0000 ~ 360.0000	128
Cosine	COS	Calculation of COS(V)		

NOTE The operands of trigonometric function are represented in degrees. When any numeric value represented in radian is input, no correct result will be obtained.

5.8.2 Sine (SIN)

(1) Function

Obtains the value of the sine in the range form 0° to 360°.

(2) Form

- Fig. 5.49 shows the form of sine (SIN).
- SIN is the symbol denoting the sine.
- SIN operation requires two elements placed vertically (top and bottom). Referring to Table 5.52, specify any register reference number for each of the elements.

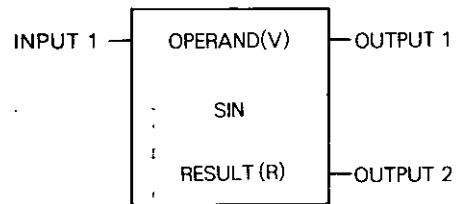


Fig. 5.49 SIN General Form

Table 5.52 Elements of SIN

Element Position	Specified Number	Description
Top	Either one of the following: • Input register (30001-30511) • Holding register (40001-49998) • Constant register (31001-35095) • Link register (R0001-R1023)	The contents of specified registers are the operand (V = 0.0000 to 360.0000). $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 \\ \boxed{0 \times \times \times} & \boxed{\times \times \times \times} \\ \text{Integer} & \text{Decimal} \\ \text{part} & \text{part} \end{array}$
Bottom	• Holding register (40001-49998) • Link register (R0001-R1023)	The result of sine operation is stored as follows: $\begin{array}{cc} \times \times \times \times & \times \times \times \times + 1 \\ \boxed{000 \times} & \boxed{\times \times \times \times} \\ \text{Integer} & \text{Decimal} \\ \text{part} & \text{part} \end{array}$

(3) Operation

- When input 1 is ON, SIN will calculate $\text{SIN}(V)$ and process the result as follows: It stores the integer of $\text{SIN}(V)$ in R and the fraction (The fifth decimal place and greater are discarded.) in R+1. It turns output 1 to ON when the result is negative. Output 1 remains in the OFF state when the result is positive. It turns output 2 to ON when the value of V exceeds 360° , and no operation is executed. Even when input 1 turns from ON to OFF, the operation result remains in R and R+1, respectively.
- The actions of SIN are tabulated in Table 5.53.

Table 5.53 Actions of SIN

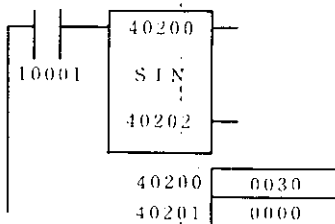
Input 1	Operation	Result	Output 1	Output 2
ON	$\text{SIN}(V) \rightarrow R$ (Integer part)	Positive	OFF	OFF
	R+1 (Decimal part)*	Negative	ON	OFF
	If $V > 360^\circ$ †	-	OFF	ON
OFF	No operation.	-	OFF	OFF

*The fifth decimal place and greater are discarded.

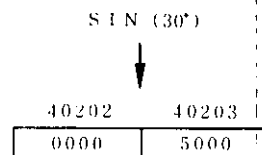
† No operation is performed even when input 1 is ON.

(4) Examples

Example 1:



(a) Ladder

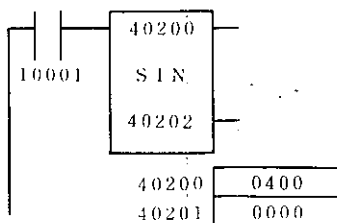


(b) Operation Content

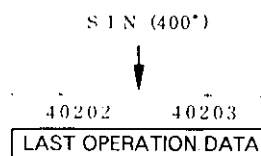
When relay 10001 is ON, SIN in (a) executes the operation of (b). Output 1 remains in the OFF state.

Even when input relay 10001 turns from ON to OFF, the operation result remains in 40202 and 40203, respectively.

Example 2:



(a) Ladder



(b) Operation Content

Since the value of V exceeds 360° , even when input relay 10001 turns to ON, the SIN in (a) turns output 2 to ON, without executing the operation.

5.8.3 Cosine (COS)

(1) Function

Obtains the value of cosine in the range from 0° to 360°.

(2) Form

- Fig. 5.50 shows the form of cosine (COS).

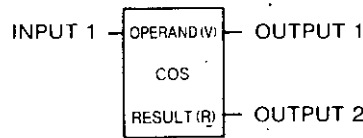


Fig. 5.50 COS General Form

- COS is the symbol denoting the cosine.
- COS operation requires two elements placed vertically (top and bottom). Referring to Table 5.54, specify register reference number for each of the elements.

Table 5.54 Elements of COS

Element Position	Specified Number	Description
Top	Either one of the following: • Input register (30001-30511) • Holding register (40001-49998) • Constant register (31001-35095) • Link register (R0001-R1023)	The contents of specified reference are the operand (V=0.0000 to 360.0000). $\begin{array}{c} \times \times \times \times \times \\ \boxed{0 \times \times \times} \\ \text{Integer} \\ \text{part} \end{array}$ $\begin{array}{c} \times \times \times \times \times + 1 \\ \boxed{\times \times \times \times} \\ \text{Decimal} \\ \text{part} \end{array}$
Bottom	• Holding register (40001-49998) • Link register (R0001-R1023)	The result of COS operation is stored as follows: $\begin{array}{c} \times \times \times \times \times \\ \boxed{000 \times} \\ \text{Integer} \\ \text{part} \end{array}$ $\begin{array}{c} \times \times \times \times \times + 1 \\ \boxed{\times \times \times \times} \\ \text{Decimal} \\ \text{part} \end{array}$

(3) Operation

- When input 1 is ON, COS will calculate COS(V) and process the result as follows: It stores the integer of COS (V) in R and the fraction (The fifth decimal place and greater are discarded.) in R+1. It turns output 1 to ON when the result is negative. Output 1 remains in the OFF state when the result is positive. It turns output 2 to ON when the value of V exceeds 360°, and no operation is executed. Even when input 1 turns from ON to OFF, the operation result remains in R and R+1, respectively.
- The actions of COS are tabulated in Table 5.55.

Table 5.55 Actions of COS

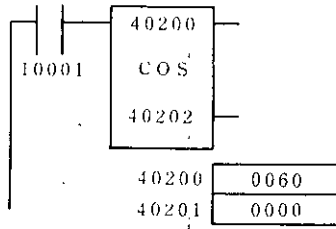
Input 1	Operation	Result	Output 1	Output 2
ON	COS (V) → R (Integer part) R+1 (Decimal part)*	Positive	OFF	OFF
		Negative	ON	OFF
	If V > 360° †	-	OFF	ON
OFF	No operation	-	OFF	OFF

* The fifth decimal place and greater are discarded.

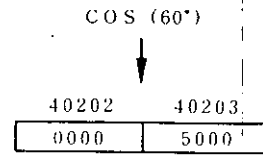
† No operation is performed even when input 1 is ON.

(4) Examples

Example 1:



(a) Ladder

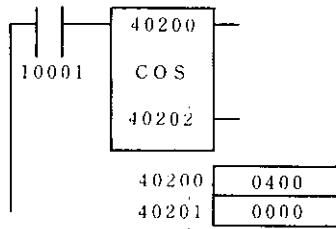


(b) Operation Content

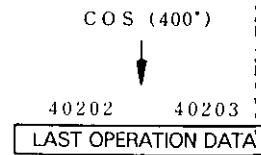
When relay 10001 is ON, SIN in (a) executes the operation of (b). Output 1 remains in the OFF state.

Even when input relay 10001 turns from ON to OFF, the operation result remains in 40202 and 40203, respectively.

Example 2:



(a) Ladder



(b) Operation Content

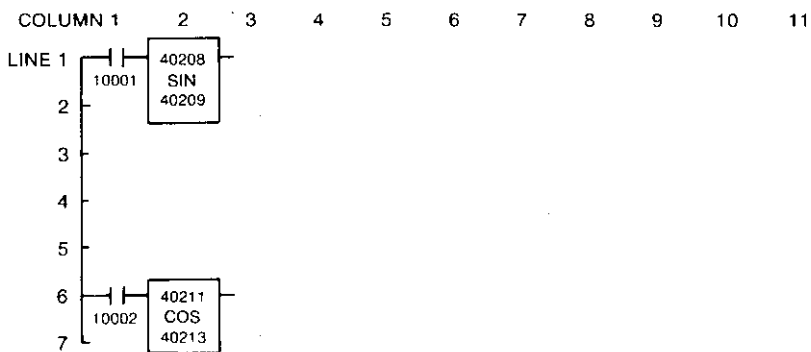
Since the value of V exceeds 360°, even when input relay 10001 turns to ON, the SIN in (a) turns output 2 to ON, without executing the operation.

5.8.4 Programming Trigonometric Function Circuit and Precautions

(1) Programming Trigonometric Function Circuit

Trigonometric function requires two elements placed vertically (top and bottom). It can be used at any intersection of the 7 lines-by-10 columns matrix, but the top element (operand) cannot be used on line 7.

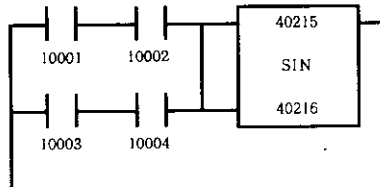
Example:



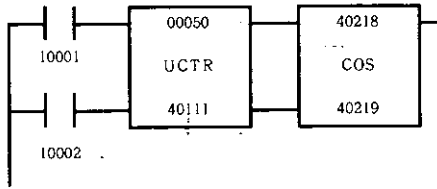
(2) Input of Trigonometric Function

Input to trigonometric function may be the output of other square roots, relays, timers, counters, or data processing circuits.

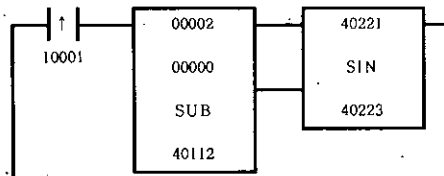
Example 1:



Example 2:



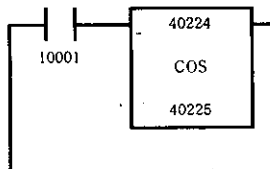
Example 3:



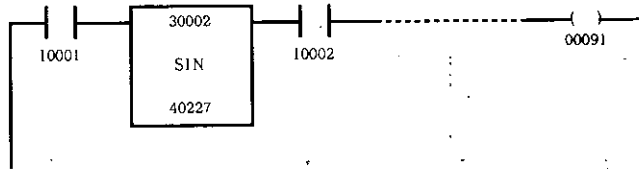
(3) Outputs of Trigonometric Function

Coils need not be connected to two output nodes (1 and 2) of trigonometric function. It is permitted to connect a relay contact to the output node at right or connect the output nodes directly to an input node of an arithmetic circuit, except relays.

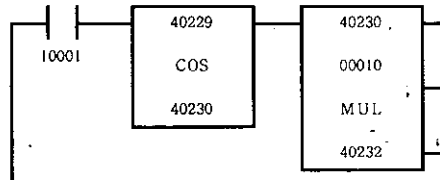
Example 1:



Example 2:



Example 3:



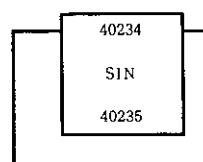
(4) Unit of Operand

The unit of operands handled in the operation of trigonometric functions is degrees. When any angle represented in radian is used as data, no correct result can be obtained, because the operation is performed according to the degree system.

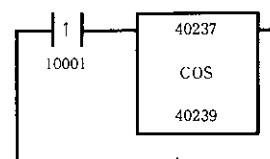
(5) Execution of Trigonometric Function (Only 1 Scan Cycle)

To execute a trigonometric functions constantly, connect the inputs directly to the power rail at left. To execute it only during one scanning cycle, use a transitional contact as an input.

Example 1:



Example 2:



5.9 DATA MOVE

Numerical values in data tables can be transferred or moved to other data tables within the controller with this function. This function has the ability to simulate the operation of electro-mechanical relays, timers and counters, as well as to perform arithmetic functions (addition, subtraction, multiply, and divide), and square root functions.

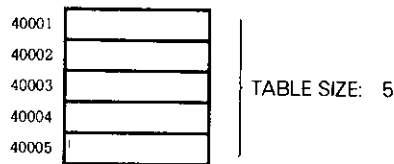
5.9.1 Basic Terminology

(1) Data Table and Table Size

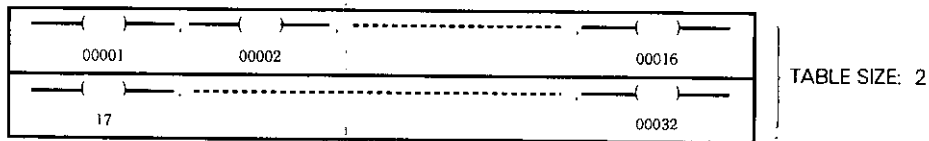
The term "Data Table" refers to register tables, coil tables and relay tables having serial numbers.

The term "Table size" refers to the size of data table, and it is counted in register units in the case of register tables and in hexadecimal units in the cases of coil tables and relay tables.

Example 1: Data table consisting of five serial holding registers



Example 2: Coil table consisting of 32 continuous coils

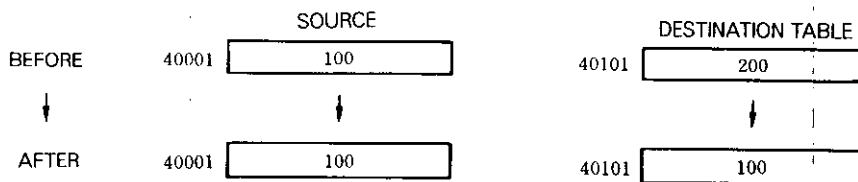


(2) Source and Destination

The source of data transfer is called "Source" and the destination of data transfer is called "Destination". Data must be transferred between the source and the destination.

- ① The content of the source is stored in the destination.
- ② The content of the source is the same as that before execution of transfer.

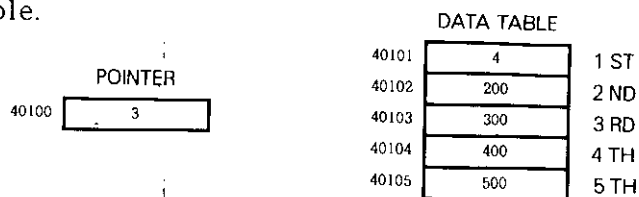
Example:



(3) Pointer

The term "Pointer" refers to the register used to indicate a specified position in data table.

Example:



5.9.1 Basic Terminology (Cont'd)

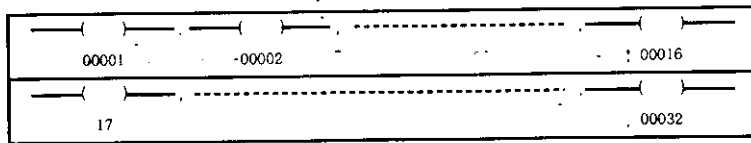
In the example shown in the preceding page, where the content of the pointer is 3, the pointer indicates 40103 located at the third place from the top.

Note that the pointer may be on the source side or on the destination side.

(4) Reference Numbers of Source and Destination

- ① In case the reference number of coil or relay is specified as the reference number of source or destination, it is necessary that the relation of $n = 16m + 1$ ($m = 0, 1, 2 \dots$) holds, where the lower-place 4 digits (lower-place 3 digits in case of step relay) is assumed to be n .

Example:



The example shown above is a case where $m = 0$, and the reference No. (n) that is specified is 00001.

- ② In the case of coil or relay, even if the actual need is less than 16 out of a set of 16 points and the rest are not used, 16 points will be transferred in one lot.
- ③ In case the destination are coils, the number of the coil table cannot overlap the coil No. (even one) used in another circuit.

5.9.2 Types of Data Move

Nine types of data move are available, as shown in Table 5.56.

Table 5.56 Types of Data Move:

Type	Symbol	Reference Page
<u>Block Move</u>	BLKM	139
<u>Register-to-Table Move</u>	R → T	142
<u>Table-to-Register Move</u>	T → R	145
<u>Table-to-Table Move</u>	T → T	148
<u>First In</u>	FIN	151
<u>First Out</u>	FOUT	153
<u>Table Search</u>	SRCH	157
<u>Table Set</u>	TSET	160
<u>Get Controller System Status</u>	STAT	162

5.9.3 Block Move (BLKM)

(1) Function

This is a move from a source table to a destination table. This function causes an entire table of registers to be copied into another table in one scan.

(2) Form

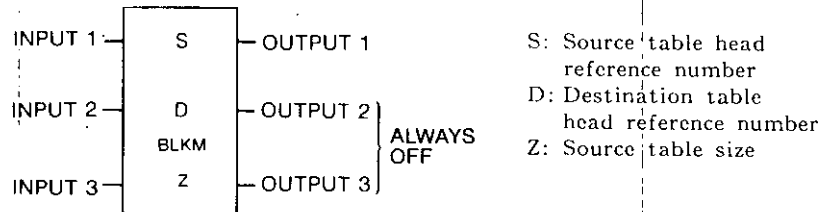


Fig. 5.51 BLKM General Form

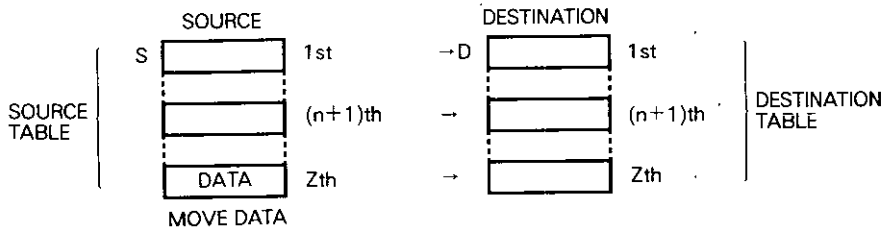
- Fig. 5.51 shows the form of block move.
- BLKM is the symbol denoting the block move.
- BLKM operation requires three elements placed vertically (top, middle and bottom). Referring to Table 5.57, specify the needed number for each element.

Table 5.57 Elements of BLKM

Element	Description	Specified Number
Top	Source table head reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Input relay (10001-14081) • Step relay (S001-S497) • Link coil (D0001-D1009) • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512)
Middle	Destination table head reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Link coil (D0001-D1009) • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<ul style="list-style-type: none"> • Coil (1-100) • Input relay (1-100) • Step relay (1-32) • Link Coil (1-64) • Each register (1-100)

- NOTE**
1. When a relay reference No. is to be specified in the top stage, the lower-place 4 digits (3 digits in case of step relay) of the number that can be specified will be limited to $[16m + 1]$. ($m = 0, 1, 2, \dots$)
 2. In case coil or relay is specified in the top stage when transfer of data is executed, data are stored in the registers in the sequence of the latest number starting with the highest order of register.

(3) Operation



NOTE

All the registers (discrete) will be moved in one scanning cycle.

(a) Inputs

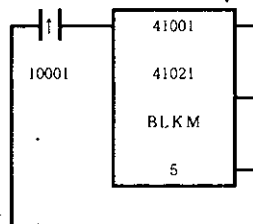
Only the input 1 is used with the BLKM function. When this input node receives power flow, the move is performed. Every scan, when enabled, the Block move will operate upon all registers.

(b) Outputs

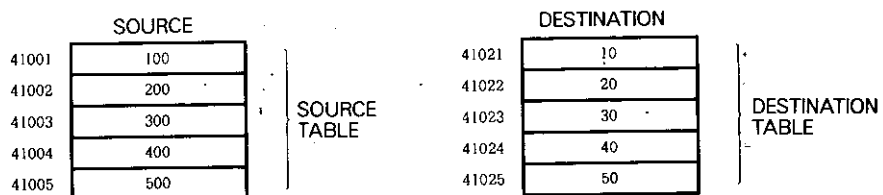
The block move function utilizes only the output 1. The lower two outputs 2 and 3 have no significance and will be OFF under all conditions. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows Function Blocks to be cascaded or chained horizontally within a network.

(4) Example

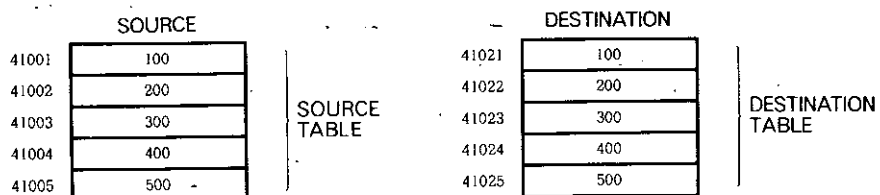
Example 1:



(a) Ladder



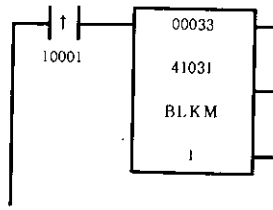
① Registers before Move



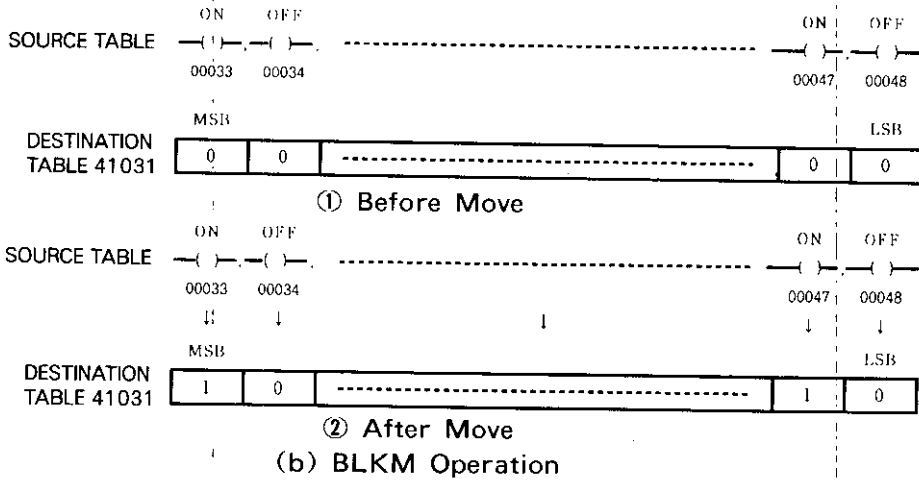
② Registers after Move

(b) BLKM Operation

Example 2:

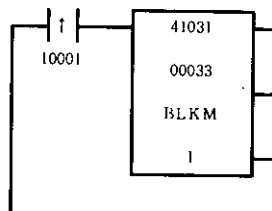


(a) Ladder

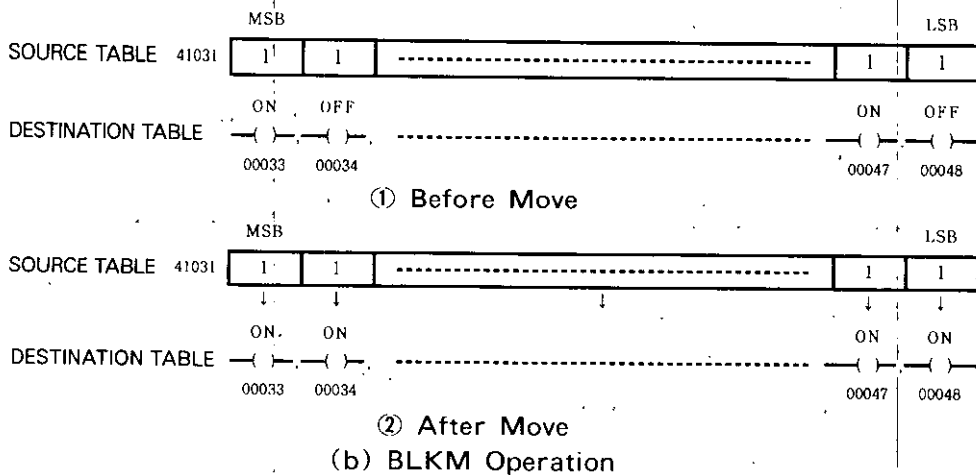


If the 16 points of source coil 00033 and onward are transferred to destination 41031, each of the ON and OFF states of the 16 points of coil 00033, 00034, ..., 00047, 00048 will enter the corresponding bits of 41031, with the ON state in the form of "1" and with the OFF state in the form of "0".

Example 3:



(a) Ladder



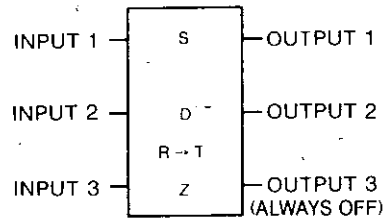
If the source holding register 41031 is transferred to the 16 points of destination coil 00033 and onward, the state of "1" and "0" of each bit of 41031 will be set in the state of "ON" and "OFF", respectively, in the 16 coils of 00033, 00034, 00047, 00048.

5.9.4 Register-to-Table Move ($R \rightarrow T$)

(1) Function

This is a move from a source to a destination table.

(2) Form



S: Source table head reference number
 P: Pointer
 Z: Destination table size

Fig. 5.42 $R \rightarrow T$ General Form

- Fig. 5.52 shows the form of register-to-table move.
- $R \rightarrow T$ is the symbol denoting the register-to-table move.
- $R \rightarrow T$ operation requires three elements placed vertically (top, middle and bottom). Referring to Table 5.58, specify the needed number for each element.

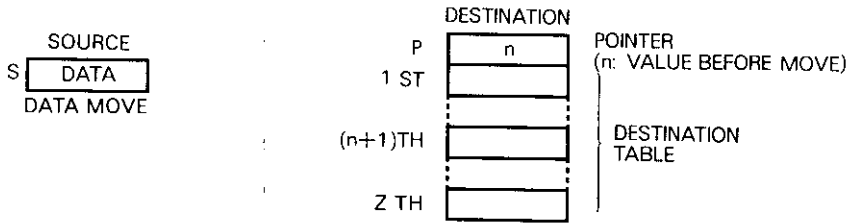
Table 5.58 Elements of $R \rightarrow T$

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Input relay (10001-14081) • Step relay (S001-S497) • Link Coil (D0001-D1009) • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512)
Middle	Pointer	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-999)

NOTE

1. When a relay reference No. is to be specified in the top stage, the lower place 4 digits (3 digits in case of step relay) of the number that can be specified will be limited to $[16m + 1]$. ($m = 0, 1, 2, \dots$)
2. In case coil or relay is specified in the top stage when transfer of data is executed, data are stored in the registers in the sequence of the latest number starting with the highest order of register.
3. Destination table starts with the reference No. next to the pointer.
4. Pointer is not included in the table size.

(3) Operation



NOTE Move will be performed at the rate of one register (or a set of 16 discrete points) per scanning cycle.

(a) Inputs

All three inputs are used with the $R \rightarrow T$ function. The input 1 controls the move. Every scan power flow is received at this node, the move is performed and the pointer incremented. Both the move and pointer incrementing occur during the solution of this Function Block. Incrementing the pointer will cause moves on future scans to occur into successive register locations. The pointer cannot exceed the table length. Thus when the pointer is equal to the table length, it will stop incrementing and the move will stop operating. A transitional contact can be used to control the input 1 if a single move operation is desired.

The input 2, when receiving power flow, inhibits the incrementing of the pointer. Thus moves with power flow at both inputs 1 and 2 can be made continuously to the same register in the table, until either another function increments the pointer or the input 2 loses power flow. The input 3 resets the pointer to zero. Whenever this input receives power flow, the pointer is reset to zero regardless of its current value.

The bottom input, when energized, will cause the $R \rightarrow T$ move to go to the first register in the table if the top element is also energized.

(b) Outputs

The register-to-table function utilizes only the first two outputs. The output 3 has no significance and will be OFF (no power flow) under all conditions. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows Function Blocks to be cascaded or chained horizontally within a network. The output 2 supplies power flow whenever the pointer is equal to the table length, when the move function has reached the end of the table.

$R \rightarrow T$ move functions at I/O ON are shown in Tables 5.59 and 5.60.

Table 5.59 $R \rightarrow T$ Move Functions at Input ON

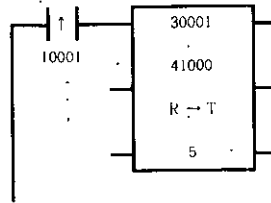
Input ×	Functions
Input 1	Executes move and adds 1 to the pointer.*
Input 2	Prohibits pointer remake.
Input 3	Sets the pointer to 0. (Regardless of Input 1 ON/OFF status.)

* If pointer \geq table size, move is not executed.

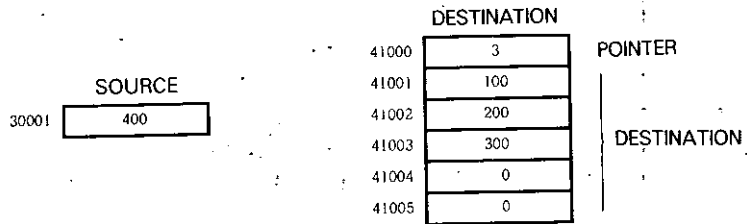
Table 5.60 $R \rightarrow T$ Move Functions at Output ON

Output ×	Functions
Output 1	Same as Input 1 ON/OFF status
Output 2	Pointer value = Table size turns ON (Regardless of Input 1 ON/OFF Status.)
Output 3	Always OFF

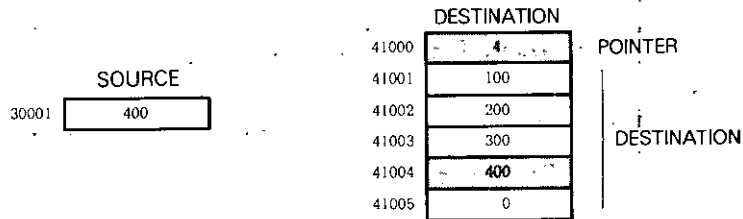
(4) Example



(a) Ladder



① Before Move



② After Move

(b) R → T Operation

R → T shown in (a) will execute transfer of data shown in (b), when input relay 10001 turns from OFF to ON. At this time, output 1 only will turn ON.

5.9.5 Table-to-Register Move ($T \rightarrow R$)

(1) Function

This is the opposite function to the $R \rightarrow T$, namely, a move from a source table to a destination.

(2) Form

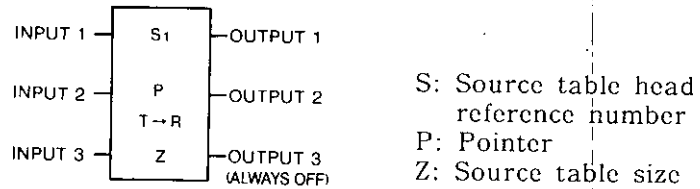


Fig. 5.53 $T \rightarrow R$ General Form

- Fig. 5.53 shows the form of block move.
- $T \rightarrow R$ is the symbol denoting the table-to-register move.
- $T \rightarrow R$ operation requires three elements placed vertically (top, middle and bottom). Referring to Table 5.61, specify the needed number for each element.

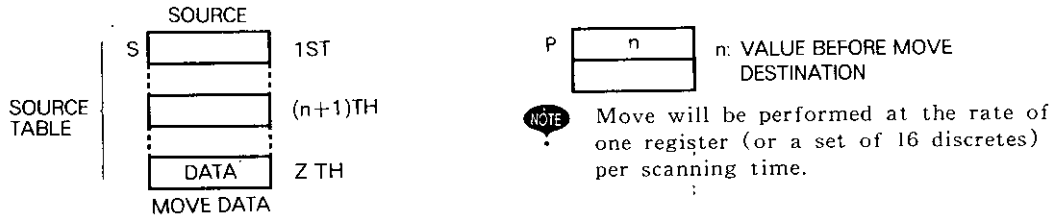
Table 5.61 Elements of $T \rightarrow R$

Element	Content	Specified number
Top	Source reference number	<ul style="list-style-type: none"> • Coil (0001-08177) • Input relay (1001-14081) • Step relay (S001-S497) • Link coil (D0001-D1009) • Input register (3001-30512) • Holding register (4001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512)
Middle	Pointer	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1024)
Bottom	Table size	<ul style="list-style-type: none"> • Coil (1-512) • Input relay (1-256) • Step relay (1-32) • Link coil (1-64) • Each register (1-999)

NOTE

1. When a relay reference No. is to be specified in the top stage, the lower place 4 digits (3 digits in case of step relay) of the number that can be specified will be limited to $[16m + 1]$. ($m = 0, 1, 2, \dots$)
2. In case coil or relay is specified in the top stage when transfer of data is executed, data are stored in the registers in the sequence of the latest number starting with the highest order of register.
3. Destination table starts with the reference No. next to the pointer.

(3) Operation



(a) Input

All three inputs are used with the $T \rightarrow R$ function. The input 1 controls the move. Every scan power flow is received at this node, the move is performed and the pointer incremented. Both the move and pointer incrementing occur during the solution of this Function Block. Incrementing the pointer will cause moves on future scans to occur from successive register locations. The pointer can not exceed the table length. Thus when the pointer is equal to the table length, it will stop incrementing and the move will stop operating. A transitional contact can be used to control the input 1 if a single move operation is desired.

The input 2, when receiving power flow, prohibits the incrementing of the pointer. Thus moves with power flow at both inputs 1 and 2 can be made continuously from the same register in the table until either another function increments the pointer or the input 1 loses power flow.

The input 3 resets the pointer to zero. Whenever this input receives power flow, the pointer is reset to zero regardless of its current value. The input 1, when energized at the same time as the input 3 will cause the first element in the table to be moved into the destination register.

(b) Outputs

The table-to-register function utilizes only the first two outputs. The output 3 has no significance and will be OFF (no power flow) under all conditions. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows Function Blocks to be cascaded or chained horizontally within a network. The output 2 supplies power flow whenever the pointer is equal to the table length, when the move function has reached the end of the table.

$T \rightarrow R$ move functions at I/O ON are shown in Tables 5.62 and 5.63.

Table 5.62 $T \rightarrow R$ Move Functions at Input ON

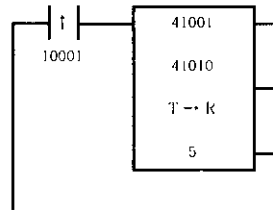
Input \times	Functions
Input 1	Executes move and add 1 to the pointer.*
Input 2	Prohibits pointer remake.
Input 3	Set the pointer to 0. (Regardless to Input 1 ON/OFF status.)

* If pointer \geq table size, move is not executed.

Table 5.63 $T \rightarrow R$ Move Functions at Output ON.

Output \times	Functions
Output 1	Same as Input 1 ON/OFF status.
Output 2	Pointer value = Table size turns ON (Regardless to Input 1 ON/OFF status.)
Output 3	Always OFF

(4) Example



(a) Ladder

41001	SOURCE TABLE
41002	100
41003	200
41004	300
41005	400
	500

41010	3	POINTER
41011	300	DESTINATION

① Before Move

	SOURCE TABLE
41001	100
41002	200
41003	300
41004	400
41005	500

41010	4	POINTER
41011	400	DESTINATION

② After Move

(b) T → R Operation

T → R shown in (a) will execute transfer of data shown in (b) when input relay 10001 turns from OFF to ON. At this time, output 1 only will turn to ON.

Unless holding register 41011 is not updated by another ladder, the relation between the pointer and destination indicates where the data of the holding register as a result of T → R is located in the table.

5.9.6 Table-to-Table Move (T → T)

(1) Function

This is a move function from a source table to a destination table.

(2) Form

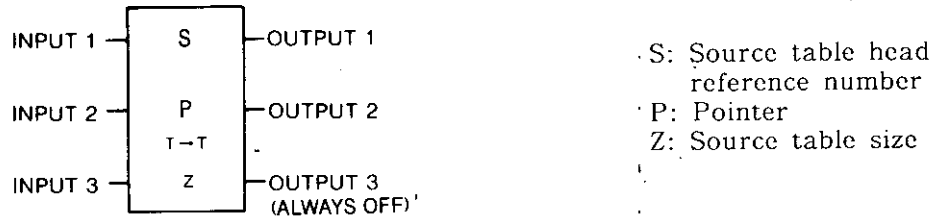


Fig. 5.54 T → T General Form

- Fig. 5.54 shows the form of table-to-table.
- T → T is the symbol denoting the table-to-table move.
- T → T operation requires three elements placed vertically (top, middle and bottom). Referring to Table 5.64, specify the needed number for each element.

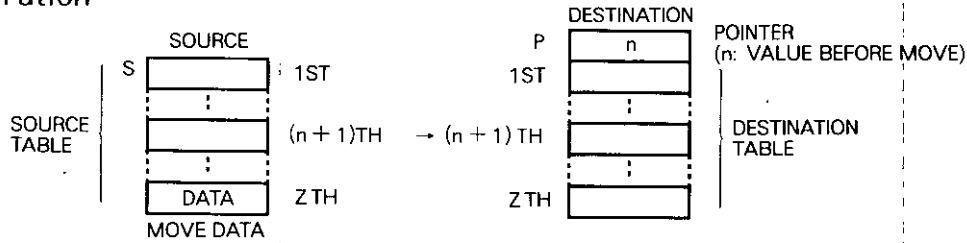
Table 5.64 Elements of T → T

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Input relay (10001-14081) • Step relay (S001-S497) • Link coil (D0001-D1009) • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512)
Middle	Pointer	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023)
Bottom	Table size	<ul style="list-style-type: none"> • Coil (1-512) • Input relay (1-256) • Step relay (1-32) • Link coil (1-64) • Each register (1-999)

NOTE

1. When a relay reference No. is to be specified in the top stage, the lower-place 4 digits (3 digits in case of step relay) of the number that can be specified will be limited to $(16m + 1)$. ($m = 0, 1, 2, \dots$)
2. In case coil or relay is specified in the top stage when transfer of data is executed, data are stored in the registers in the sequence of the latest number starting with the highest order of register.
3. Destination table starts with the reference No. next to the pointer.
4. Pointer is not included in the table size.

(3) Operation



NOTE Move will be performed at the rate of one register (or a set of 16 discrettes) per scanning cycle.

(a) Inputs

All three inputs are used with the $T \rightarrow T$ function. The input 1 controls the move. Every scan power flow is received at this node, the move is performed and the pointer incremented. Both the move and pointer incrementing occur during the solution of this function Block. Incrementing the pointer will cause moves on future scans to occur at successive register locations. The pointer cannot exceed the table length. Thus when the pointer is equal to the table length, it will stop incrementing and the move will stop operating.

A transitional contact will be used to control the input 1 if the single move operation is desired.

The input 2, when receiving power flow at both inputs 1 and 2 can be made continuously between the same registers in the tables, until another function increments the pointer or the input 2 loses power flow. The output 3 can reset the pointer to zero. Whenever this input receives power flow, the pointer is reset to zero regardless of its current value.

The input 1, when energized at the same time as the input 3, will cause the first element in the source table to be copied to the first element in the destination table.

(b) Outputs

The table-to-table function utilizes only the first two outputs. The output 3 has no significance and will be OFF (no power flow) under all conditions. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows Function Blocks to be cascaded or chained horizontally within a network. The output 2 supplies power flow whenever the pointer is equal to the table length and indicates when the move function has reached the end of the table.

$T \rightarrow T$ move functions at I/O ON are shown in Tables 5.65 and 5.66.

Table 5.65 $T \rightarrow T$ Move Functions at Input ON

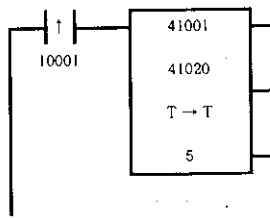
Input \times	Functions
Input 1	Executes move and add 1 to the pointer.*
Input 2	Prohibits pointer remake.
Input 3	Set the pointer to 0. (Regardless to Input 1 ON/OFF status.)

*If pointer \geq table size, move is not executed.

Table 5.66 $T \rightarrow T$ Move Functions at Output ON

Output \times	Functions
Output 1	Same as Input 1 ON/OFF status
Output 2	Pointer value = Table size turns ON (Regardless to Input 1 ON/OFF Status.)
Output 3	Always OFF

(4) Example



(a) Ladder

SOURCE TABLE		DESTINATION TABLE	
	SOURCE		DESTINATION
40001	100	41020	1
41002	200	41021	100
41003	300	41022	0
41004	400	41023	0
41005	500	41024	0
		41025	0

① Before Move

SOURCE TABLE		DESTINATION TABLE	
	SOURCE		DESTINATION
40001	100	41020	2
41002	200	41021	100
41003	300	41022	200
41004	400	41023	0
41005	500	41024	0
		41025	0

② After Move

(b) T → T Operation

T → T shown in (a) will execute transfer of data shown in (b) input relay 10001 turns from OFF to ON. At this time, output 1 only will turn ON.

5.9.7 First In (FIN)

(1) Function

FIN is used in pair with a First Out (FOUT) (see Par. 5.9.8). This is like move of the R → T. The difference is that the data stored in the destination by FIN can be retrieved by FOUT in the order the data are stored.

(2) Form

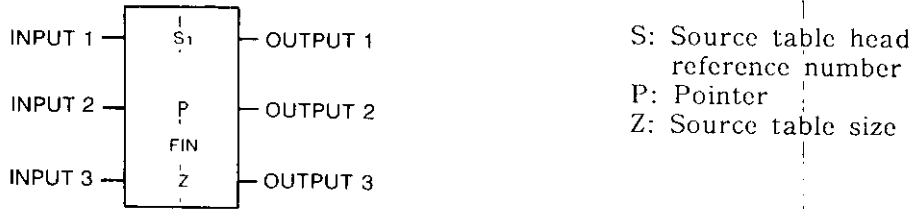


Fig. 5.55 FIN General Form

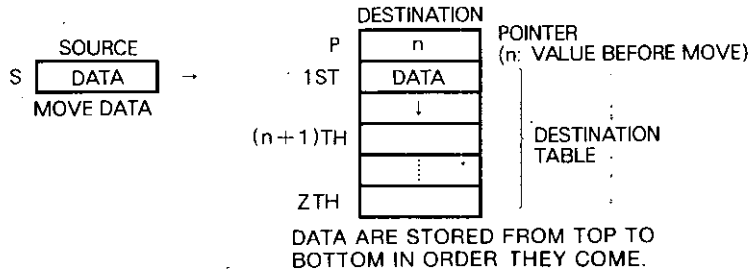
- Fig. 5.55 shows the form of first in.
- FIN is the symbol denoting the first in.
- FIN operation requires three elements placed vertically (top, middle and bottom). Referring to Table 5.67, specify the needed number for each element.

Table 5.67 Elements of FIN

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Input relay (10001-14081) • Step relay (S001-S497) • Link coil (D0001-D1009) • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512)
Middle	Pointer	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-100)

- NOTE**
1. When a relay reference No. is to be specified in the top stage, the lower-place 4 digits (3 digits in case of step relay) of the number that can be specified will be limited to $[16m + 1]$. ($m = 0, 1, 2, \dots$)
 2. In case coil or relay is specified in the top stage when transfer of data is executed, data are stored in the registers in the sequence of the latest number starting with the highest order of register.
 3. Destination table starts with the reference No. next to the pointer.
 4. Pointer is not included in the table size.

(3) Operation



NOTE Move will be performed at the rate of one register (or a set of 16 discretes) per scanning cycle.

(a) Inputs

When the input 1 is ON, the contents of the destination table are shifted down by one, according to the content of pointer, the n th data (the oldest one) to the $(n+1)$ th register, the $(n-1)$ th data to the n th register, and so on, until the first data are shifted down to the second register. Then the source data is moved to the first register emptied. Thus the destination table is managed in such a manner that the data are stored in the table from top to bottom in the order they come. The pointer is incremented by one after the shift and move of data. This process is performed all in one scanning cycle, regardless of the table size. If $n \geq$ table size, no data will be moved even when the input 1 is ON. Inputs 2 and 3 are not used.

(b) Outputs

FIN function uses all three outputs. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows Function Blocks to be cascaded or chained horizontally within a network. The output 2 supplies power flow whenever the table is full (pointer equal to table length); the output 3 supplies power flow whenever the table is empty (pointer equals zero). The outputs 2 and 3 do NOT require any inputs to receive power flow, they only require appropriate pointer values.

Table 5.68 shows FIN operation.

Table 5.68 FIN Operation

Input 1	Operation	Condition	Output1	Output2	Output3
ON	Execution	Pointer value = Table size	ON	ON	OFF
		Pointer value = 0		OFF	ON
		Other than the above		OFF	OFF
OFF		Pointer value = Table size	OFF	ON	OFF
		Pointer value = 0		OFF	ON
		Other than the above		OFF	OFF

* If pointer > table size, this operation will not be executed.

(4) Example

Example is shown in the section of FOUT.

5.9.8 First Out (FOUT)

(1) Function

This is an N-to-1 move that the destination table of FIN becomes the source table of FOUT.

(2) Form

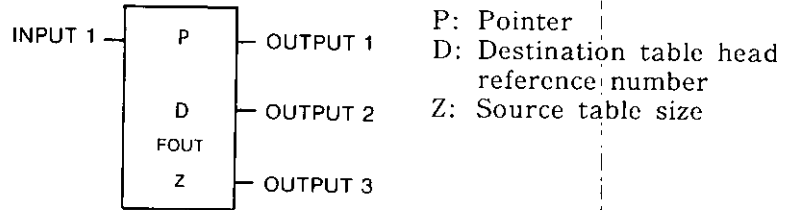


Fig. 5.56 FOUT General Form

- Fig. 5.56 shows the form of first out.
- FOUT is the symbol denoting the first out.
- FOUT operation requires three elements placed vertically (top, middle and bottom). Referring to Table 5.69, specify the needed number for each element.

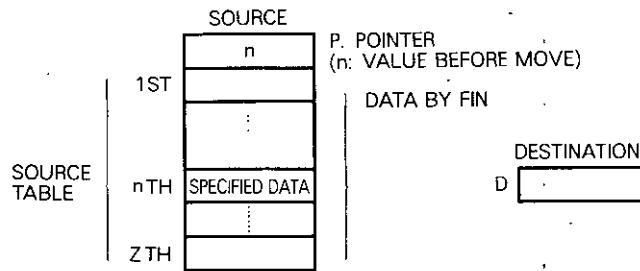
Table 5.69 FOUT Elements

Element	Description	Specified Number
Top	Pointer	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023)
Middle	Destination reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Link Coil (D0001-D1009) • Holding register (40001-49998) • Link register (R0001-R1023)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-100)

NOTE

1. When a relay reference No. is to be specified in the top stage, the lower place 4 digits (3 digits in case of step relay) of the number that can be specified will be limited to $[16m + 1]$. ($m = 0, 1, 2, \dots$)
2. In case coil or relay is specified in the top stage when transfer of data is executed, data are stored in the registers in the sequence of the latest number starting with the highest order of register.
3. Destination table starts with the reference No. next to the pointer.
4. Pointer is not included in the table size.

(3) Operation



Move will be performed at the rate of one register per scanning cycle. The data stored in a FIFO table will be retrieved by FOUT with the oldest one first, i.e. by the First in-First out principle.

(a) Inputs

When the input 1 is ON, the nth (but not the (n+1)th) contents of the source table are moved to the destination where n is the value of the pointer indicating the number of data stored. The pointer is decremented by one after moved of the data. If n=0, data will not be moved even when the input 1 is ON. The inputs 2 and 3 are not used.

NOTE The nth source register, whose contents are retrieved by FOUT, holds 0 unless new data is placed by FIN.

(b) Outputs

FOUT function uses all three outputs: each of the three outputs behaves in the same way on FIN function block. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows Function Blocks to be cascaded or chained horizontally within a network. The output 2 supplies power flow whenever the table is full (pointer equal to table length); the output 3 supplies power flow whenever the table is empty (pointer equals zero). The outputs 2 and 3 do not require any inputs to receive power flow, they only require appropriate pointer values.

Table 5.70 shows FOUT operation

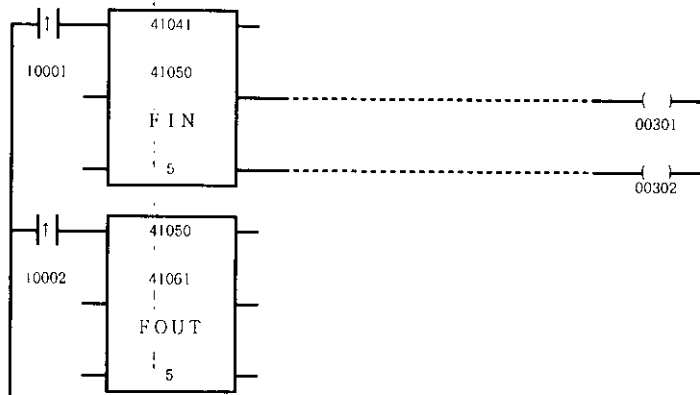
Table 5.70 FOUT Operation

Input 1	Operation	Condition	Output1	Output2	Output3
ON	Execution *	Pointer value = Table size	ON	ON	OFF
		Pointer value = 0		OFF	ON
		Other than the above		OFF	OFF
OFF	-	Pointer value = Table size	OFF	ON	OFF
		Pointer value = 0		OFF	ON
		Other than the above		OFF	OFF

* If pointer > table size, this operation will not be executed.

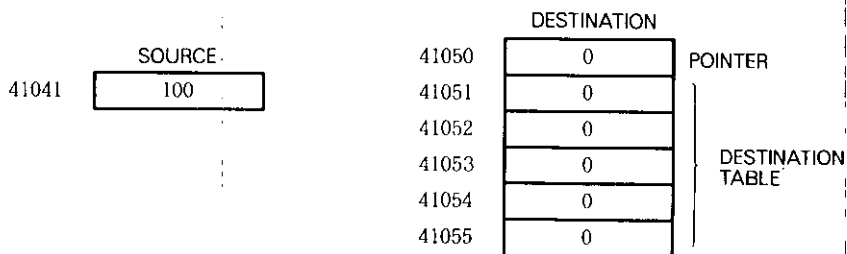
(4) Example (FIN, FOUT)

(a) Ladder

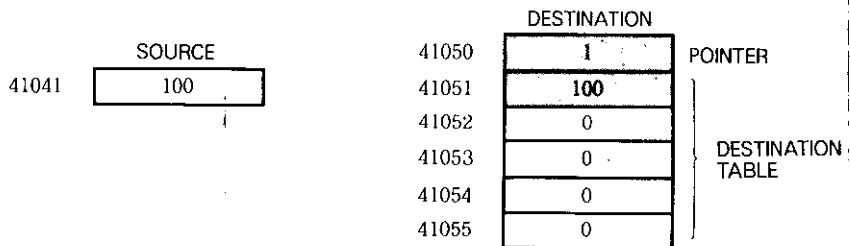


(b) Content of transfer (FIN)

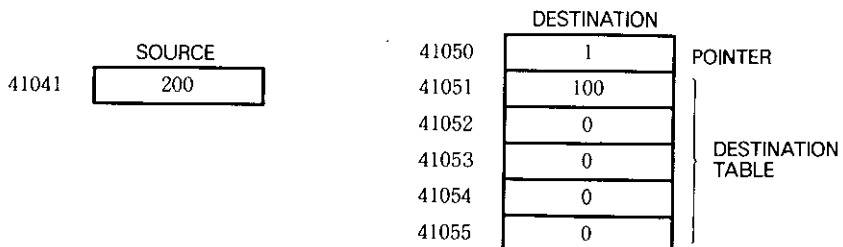
- ① When the content of the FIFO table before execution of (the first) transfer is as follows.



- ② When the first transfer is executed after turning 10001 from OFF to ON.

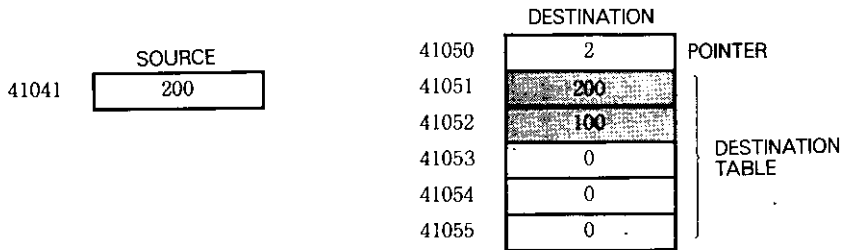


- ③ The content of the register before the second transfer is shown below.



- ④ When the second transfer is executed after turning 10001 from OFF to ON, the data transferred first are shifted down and the present source data are stored in the head of the destination table.

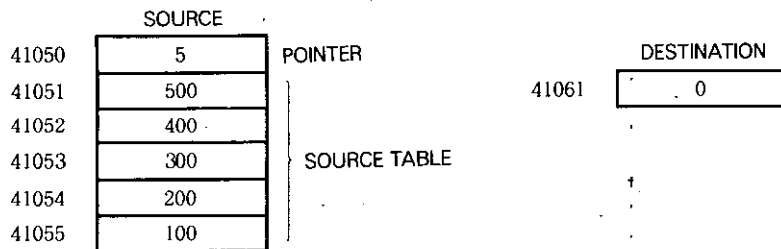
5.9.8 First Out (FOUT) (Cont'd)



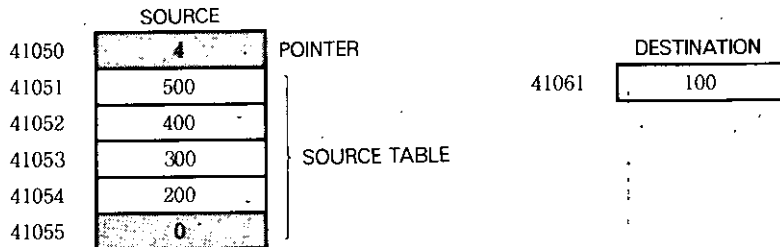
Coil 00301 turns ON when the value of pointer is 5, and coil 00302 turns ON when the value of pointer is 0.

(c) Content of transfer (FOUT)

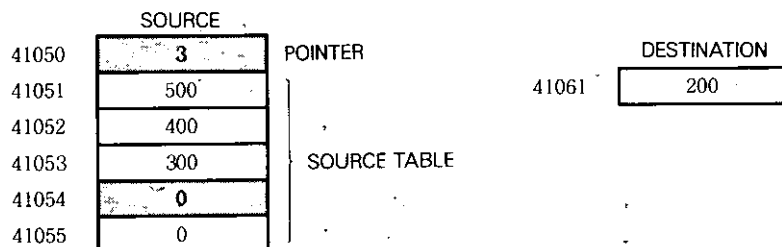
- ① When the content of the FIFO table before execution of (the first) transfer is as follows.



- ② When the first transfer is executed after turning 10002 from OFF to ON.



- ③ When the second transfer is executed after turning 10002 ON.



5.9.9 Table Search (SRCH)

(1) Function

This function searches a table of registers for a specified value. The source is not altered, only examined. If necessary, the SRCH function searches the entire table in one scan. It uses a pointer to indicate the location(s) within the table of registers which contain the value for which it is searching. This pointer is the only register whose value is altered by the SRCH function.

(2) Form

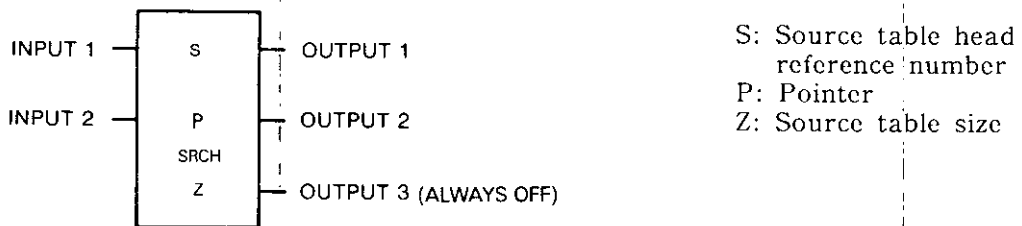


Fig. 5.57 SRCH General Form

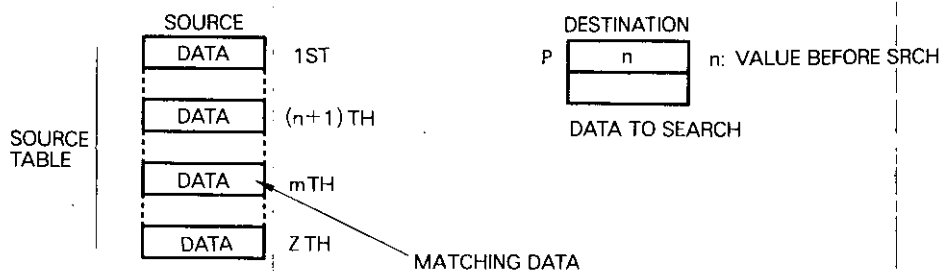
- Fig. 5.57 shows the form of search.
- SRCH is the symbol denoting the search.
- SRCH operation requires three elements placed vertically (top, middle and bottom). Referring to Table 5.71, specify the needed number for each element.

Table 5.71 Elements of SRCH

Element	Description	Specified Number
Top	Source table head reference number	<ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024)
Middle	Pointer	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-100)

NOTE Destination table starts with the reference No. next to the pointer.

(3) Operation



5.9.9 Table Search (SRCH) (Cont'd)

Before search, store the data to be searched in the location next to the destination pointer. Assume the value of the pointer is n when SRCH is required, then search is started from the $(n+1)$ th source data. When a matching data is found at the m th location, the search will be completed with m placed in the pointer. If a matching data is not found, the search will be completed with 0 placed in the pointer.

SRCH searches for one piece of matching data during every scanning cycle. If all data of the source table matches, it takes N scanning cycles (N is the table size) to complete the search. If there is no matching data, the search is completed in one scanning cycle.

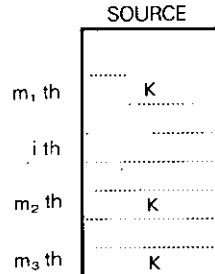
Where to start the next search depends on the status of inputs 1 and 2.

(a) Inputs

When only the input 1 is ON, n is set to 0 automatically before searching. SRCH always starts at the top of the source table. Even if there are many pieces of matching data in the source table, only the first one will be detected by search.

When the inputs 1 and 2 are ON, SRCH starts at the $(n+1)$ th data of the source table when the pointer is n before searching. If the search is required from the i th data, set the pointer to $i-1$ and turn on the inputs 1 and 2.

If the m_1 th, m_2 th, and m_3 th data of the source table are equal to the specified data K , set the pointer to $i-1$ and turn on the inputs 1 and 2. The search starts at the i th data and ends with $n=m_2$. The next search will start at the (m_2+1) th data and ends with $n=m_3$.



In any cases (only the input 1 is ON, or the inputs 1 and 2 are ON);

- If matching data is not found, the search of the scanning cycle is completed with $n=N$ (N is the table size) then n is cleared to 0. (The search is not performed again from the top of the table during the same scanning cycle.)
- If the input is turned on when $n \geq N$, n is automatically cleared to 0 before searching. As a result, the search is started at the top of source data. The input 3 is not used.

NOTE The pointer is 0 whenever the input 1 is OFF.

(b) Outputs

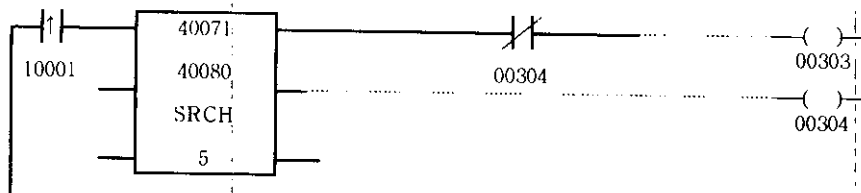
The SRCH function utilizes only the outputs 1 and 2. The output 3 has no significance and will be OFF (no power flow) under all conditions. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows Function Blocks to be cascaded or chained horizontally within a network. The output 2 supplies power flow whenever a match is found; if no match is found, this output will not supply power flow and the pointer will contain the number zero.

Table 5.72 shows SRCH operation.

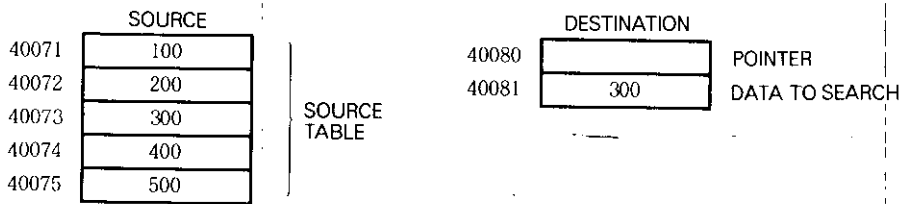
Table 5.72 SRCH operation

Input 1	Input 2	Operations	Miscompare	Output 1	Output 2
ON	OFF	Search from the first source register.	Yes	ON	ON
			No		OFF
	ON	Search from the next source register to one indicated by the pointer.	Yes		ON
			No		OFF
OFF	—	Not executed.	—	OFF	OFF

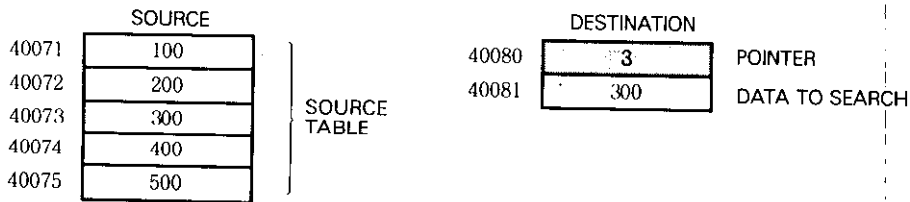
(4) Example



(a) Ladder



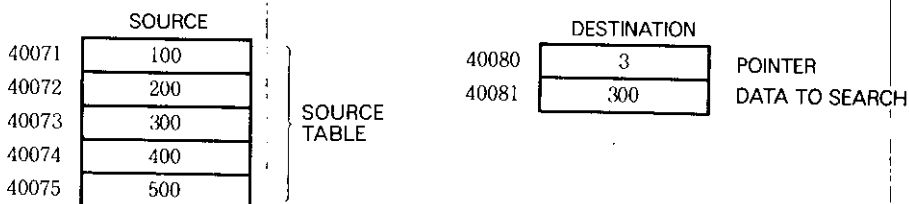
① Data before Execution



② Data after Execution

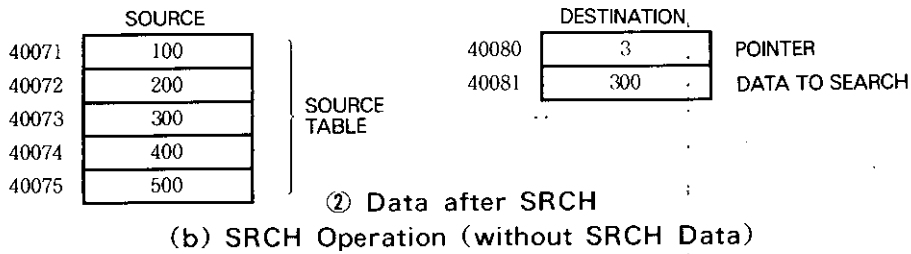
(b) SRCH Operation (with SRCH Data)

When input relay 10001 is turned ON, SRCH shown in (a) detects the data that coincide with the content of destination in the third register from the head of the source table as shown in (b) and sets 3 in the pointer. At this time, coil 00304 only turns ON.



① Data before Execution

5.9.9 Table Search (SRCH) (Cont'd)



When input relay 10001 is turned from OFF to ON, SRCH shown in (b) starts Search from the fourth register from the head of source table according to the instructions of the pointer. Nothing that coincides with the content of destination register in the fourth register onward of source table, so that the value of the pointer remains 3 without being updated. At this time, coil 00303 only turns ON. Coil 00303 can be used as Search End signal if there is no coinciding data.

5.9.10 Table Set (TSET)

(1) Function

Moves the source content in 1 scan cycle to all destination tables.

(2) Form

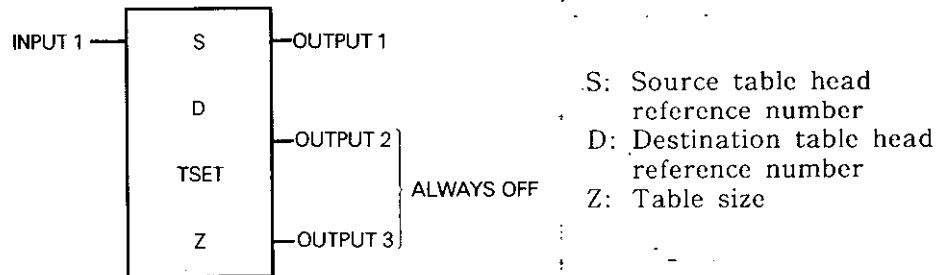


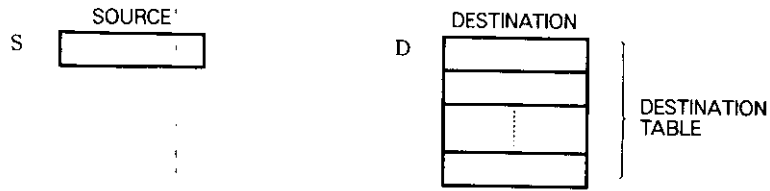
Fig. 5.58 TSET General Form

- Fig. 5.58 shows the form of table set.
- TSET is the symbol denoting the table set.
- TSET operation requires three elements placed vertically (top, middle and bottom). Referring to Table 5.73, specify the needed number for each element.

Table 5.73 Elements of TSET

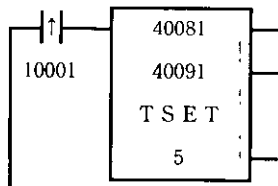
Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024)
Middle	Destination table head reference number	<ul style="list-style-type: none"> • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-100)

(3) Operation

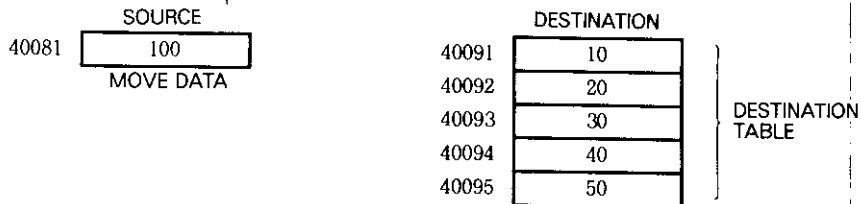


- When the input 1 is ON, TSET moves the source content to all destination tables.
- When the input 1 is ON, the output 1 is ON (Copy of the input 1.)
The outputs 2 and 3 are always OFF.

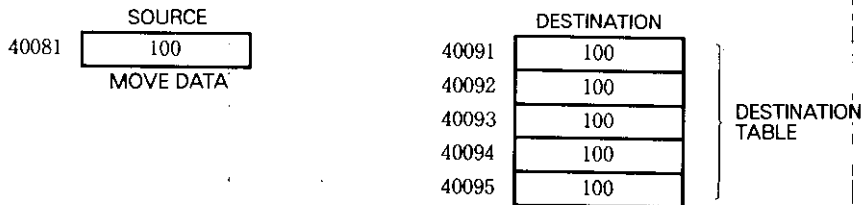
(4) Example



(a) Ladder



① Data before Move



② Data after Move

(b) TSET Operation

If the input relay 10001 is turned to ON, TSET in (a) moves the source register content to all registers of destination table as shown in (b).

5.9.11 Get Controller System Status (STAT)

(1) Function

This function provides the user with access to the GL60H or GL70H system status. The STAT is useful in the design of system diagnostics.

(2) Form

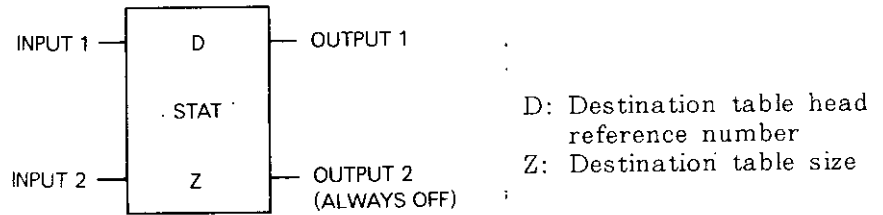


Fig. 5.59 STAT General Form

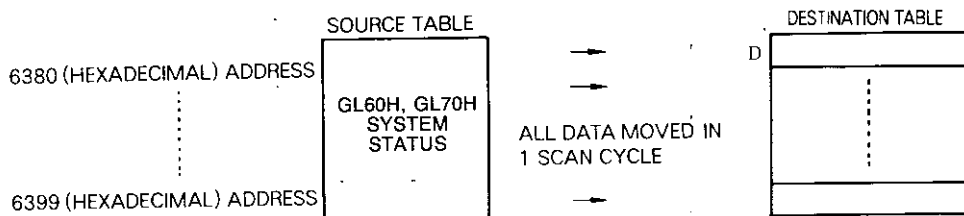
- Fig. 5.59 shows the form of system status.
- STAT is the symbol denoting the system status.
- STAT operation requires two elements placed vertically (top and bottom). Referring to Table 5.74, specify the needed number for each element.

Table 5.74 STAT Elements

Element	Description	Specified Number
Top	Destination table head reference number	<ul style="list-style-type: none"> • Coil (00001-08192) • Link coil (D0001-D1024) • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-128)

(3) Operation

128 registers worth of status can be obtained every scan. The length of the status move can be adjusted; however, it will always start with the GL60S machine status. The source is fixed by the GL60H's or GL70H's internal design and is not under user control.



(a) Inputs

Of the two available inputs, only the input 1 affects the operation of the STAT function. When this input node receives power flow, all statuses are moved to the destination in one scan.

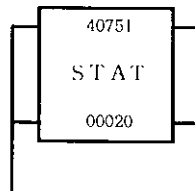
(b) Outputs

Of the two available outputs, only the output 1 is used. The output 2 has no significance and will be OFF under all conditions. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows Function Blocks to be cascaded or chained horizontally within a network.

(c) System Status

The moved system statuses are located in the destination as shown in Par. 9.6.

(4) Example



Status information can be obtained constantly because the input 1 is connected to the power rail at left. It is stored in 20 registers of 40751 to 40770, in the order.

5.9.12 Programming Data Move Circuit and Precautions

- (1) Inputs to the data move circuit may be outputs of relays, timers, counters, arithmetic operations, matrices, and other data-move circuits.
- (2) Coils need not be connected to three output nodes (1, 2 and 3) of a data move circuit. It is permitted to connect a relay contact to the output nodes at right or connect the output node directly to an input node of an arithmetic circuit, except relays.
- (3) To execute an operation constantly, connect the input directly to the power rail at left. To execute it only during one scanning cycle, use a transitional contact as an input.
- (4) The range of the source or destination table specified by a table size must be within the range of the reference numbers of input relays, coils, or registers.
- (5) Since the output 1 supplies power flow whenever the input 1 receives power flow, cascaded operation is possible. It is possible to OR the outputs by connecting a vertical shunt element.
- (6) When coil $0 \times \times \times \times$ is to be specified as destination, the number of the coil group cannot be the same as the number of a coil group used in another circuit. If it is attempted, the message "COIL IS USED" is displayed on the screen of the programming panel and input operation will be rejected. Coil 02049 is used for battery monitor and the destination table cannot include coils 02049-02064.
- (7) When coil $0 \times \times \times \times$ is specified as destination, the coil is turned on and off by data move function even if it is disabled.
- (8) When a pointer is used, its value does not exceed the table size normally. If the value is made greater than the table size forcibly by another circuit, move is not executed (SRCH executes it after resetting the pointer to 0) and the pointer becomes equal to the table size.
- (9) The source remains unchanged data after the move.

5.9.13 Example – Application Circuits of Data Move

(1) Setting and Reading Data (Applications of $R \rightarrow T$ and $T \rightarrow R$)

This is an example to set and read a time of externally-preset timers. Fig. 5.60 shows a sample layout of switches and indicator.

Setting:

Set a timer number and time by the digital switches and push the SET pushbutton switch to store the set time in the register associated with the timer number and indicate the time.

Reading:

Set a timer number and push the READ pushbutton switch to read out the set time corresponding to the timer number from a table and indicate the time. It is assumed that the timer number will be N and the set time will be T_N .

(a) Hardware Configuration

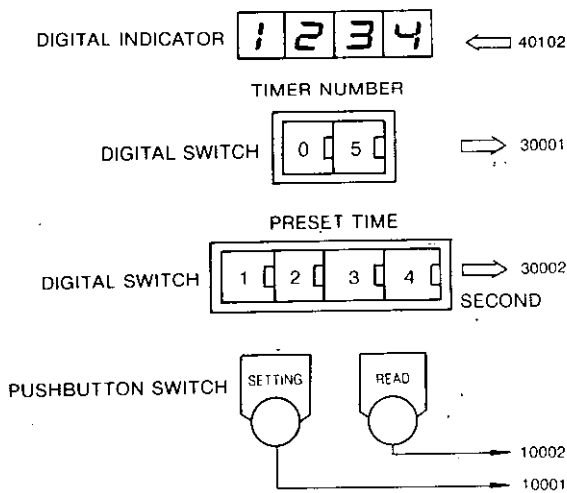


Fig. 5.60 External Devices and Assignments

(b) Ladder Diagram

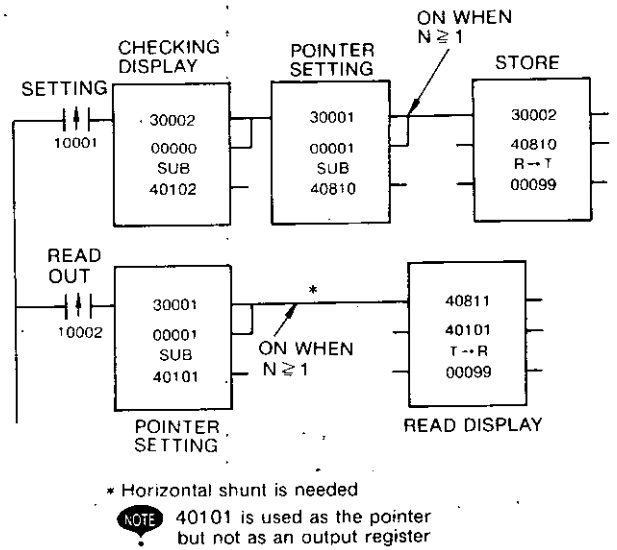


Fig. 5.61 Setting/Reading Circuits

(c) Data Flow

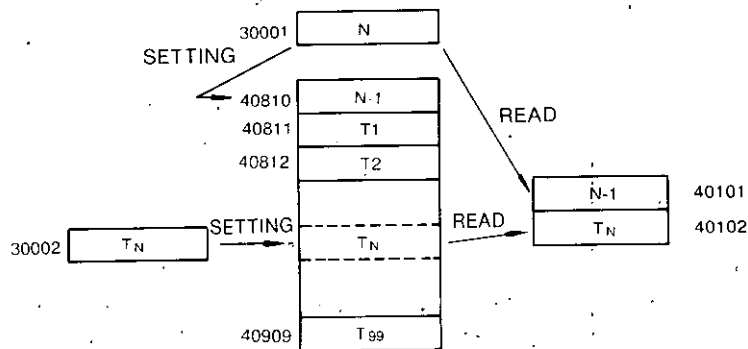


Fig. 5.62 Data Flow

(3) Reservation (Applications of FIN and FOUT)

In this example, articles are grouped by destination and placed on delivery tracks. It is assumed that branching and interruption do not happen during transportation.

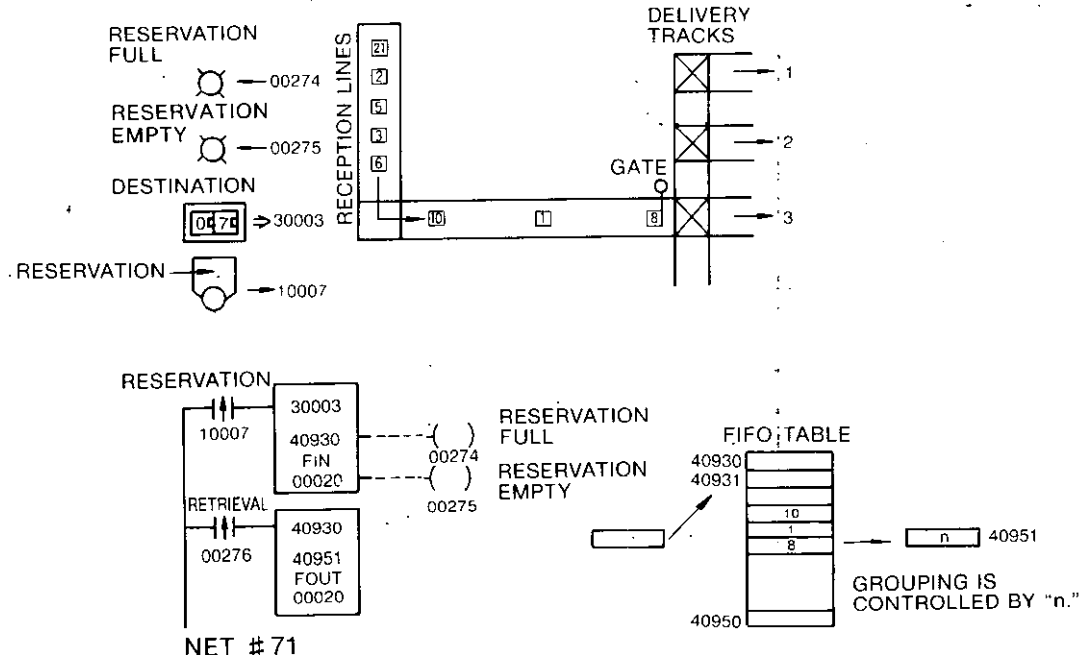


Fig. 5.65 Reservation Circuit

(4) Prevention of Double Reservation (Application of SRCH)

This is an example to prevent double reservation in the system shown in (3) above.

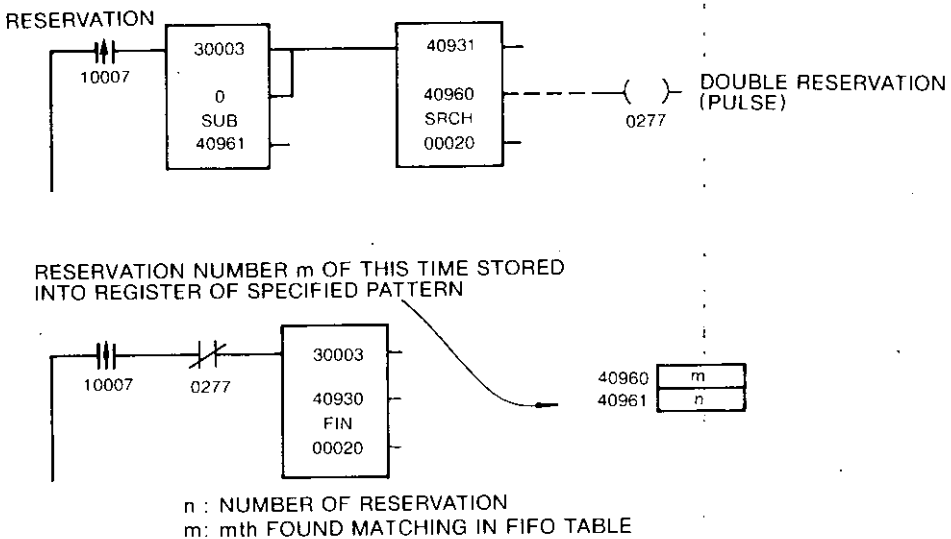


Fig. 5.66 Prevention Circuit of Double Reservation

5.10 INDEXED BLOCK MOVE

Indexed block move is a table-to-table move executed in one scanning cycle, like block move (BLKM), described in 5.9.3. In addition, there are following functions:

- The source or destination table can be specified according to the value (index) of the pointer.
- A group of input relays can be specified as the destination table.

5.10.1 Types of Indexed Block Move

Four types of the indexed block move are available as shown in Table 5.75.

Table 5.75 Types of Indexed Block Move

Type	Symbol	Description	Page
Block Move 1 with Destination Index	DIBT	Permits to specify destination table (input relay groups) according to pointer.	167
Block Move 2 with Destination Index	DIBR	Permits to specify destination table (holding register groups) according to pointer.	172
Block Move 1 with Source Index	SIBT	Permits to specify source table (coils, input relays and input register groups) according to pointer.	175
Block Move 2 with Source Index	SIBR	Permits to specify source table (holding register groups) according to pointer.	179

5.10.2 Block Move 1 with Destination Index (DIBT)

(1) Function

This function can transfer all data of source table to the destination table specified by the pointer.

(2) Form

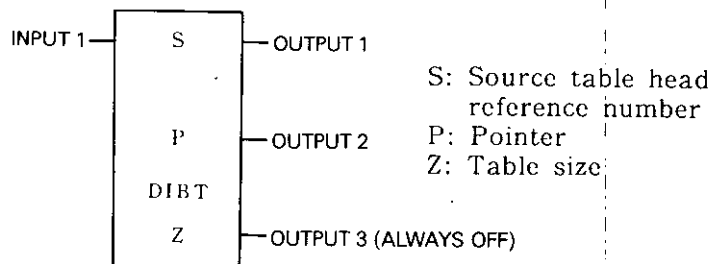


Fig. 5.67 DIBT General Form

5.10.2 Block Move 1 with Destination Index (DIBT) (Cont'd)

- Fig. 5.67 shows the form of the block move 1 with destination index (DIBT).
- DIBT is the symbol denoting the block move 1 with destination index.
- DIBT requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.76, specify any of constant K, reference numbers of discrete group and reference numbers of various registers for each of the elements.

Table 5.76 DIBT Elements

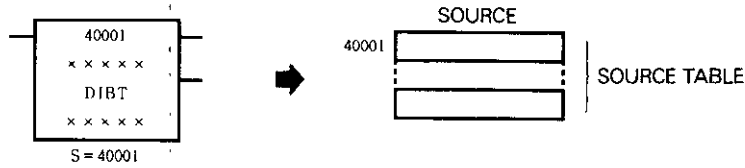
Element	Description	Specified Number
Top	Source table head reference number	Any of the following: <ul style="list-style-type: none"> • Coil (00001-08177) • Input relay (10001-14081) • Step relay (S001-S497) • Link coil (D0001-D1009) • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512)
Middle	With pointer, specified head number of destination table	<ul style="list-style-type: none"> • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Size of source table and destination table	<ul style="list-style-type: none"> • Where step relay, constant (1-32) • Where link coil, constant (1-64) • Others (1-100)

- NOTE**
1. When a relay reference No. is to be specified in the top stage, the lower-place 4 digits (3 digits in case of step relay) of the number that can be specified will be limited to $[16m + 1]$. ($m = 0, 1, 2, \dots$)
 2. In case coil or relay is specified in the top stage when transfer of data is executed, data are stored in the registers in the sequence of the latest number starting with the highest order of register.

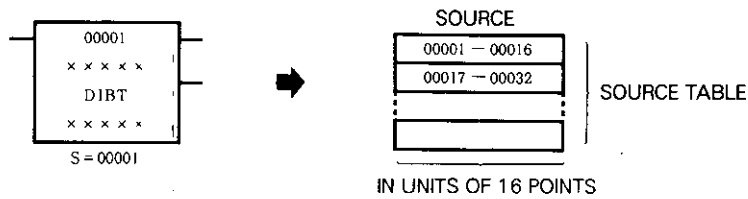
(3) Operation

The relation between source table and its head number is as follows:

Example 1:

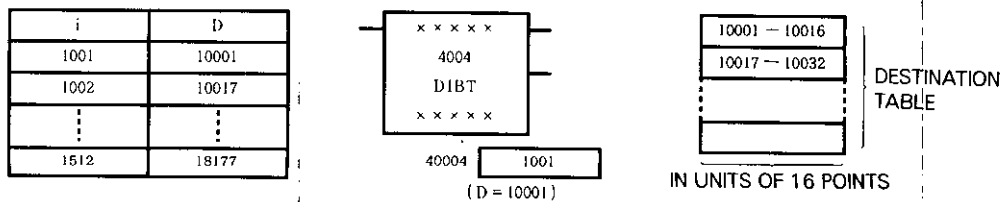


Example 2:

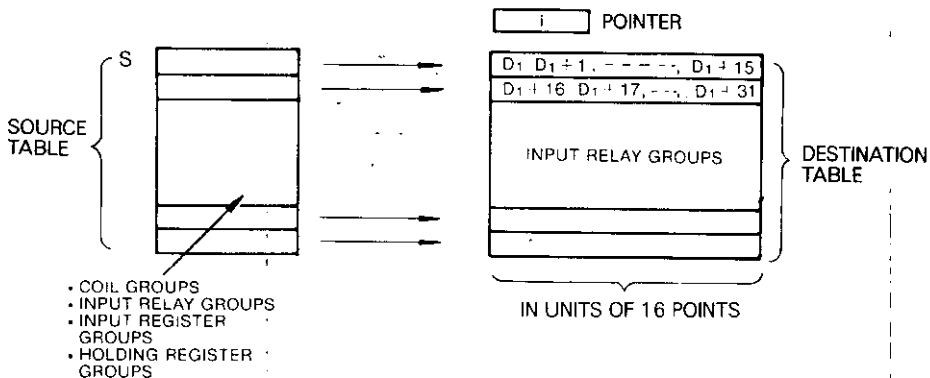


If a relay table or a coil table is to be specified, $n = 16m + 1$ ($m = 0, 1, 2, \dots$) must hold, where n represents the lower-place 4 digits (lower-place 3 digits in case of step relay). The relation between pointer content (i) and the head number (D) of destination table (input relay table) is shown below.

Example:



- By DIBT, all data of the source table will be moved to the destination table (input relay groups) specified with the value i of the pointer when the input 1 is ON. The output 1 is turned ON. The move will be completed in one scanning cycle.



5.10.2 Block Move 1 with Destination Index (DIBT) (Cont'd)

- In the following cases, the move will not be executed and the output 2 is turned on.

(a) When the value i of the pointer is out of the range of 1001-1256:

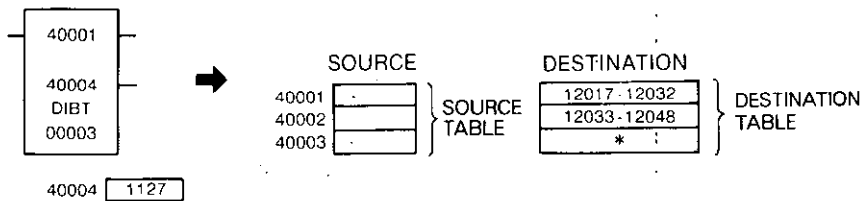
Example:



NOTE There is no input relay group corresponding to $i=0$. Therefore, no destination tables corresponding to the source tables 40001-40003 exist.

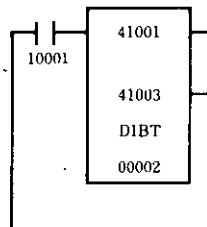
(b) When the value i of the pointer is within the range of 1001-1256 but no destination table to be specified by the value exists:

Example:

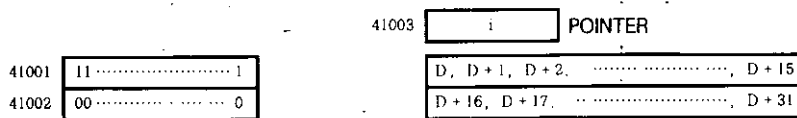


* No input relay groups of destination corresponding to the source 40003 exist.

(4) Example

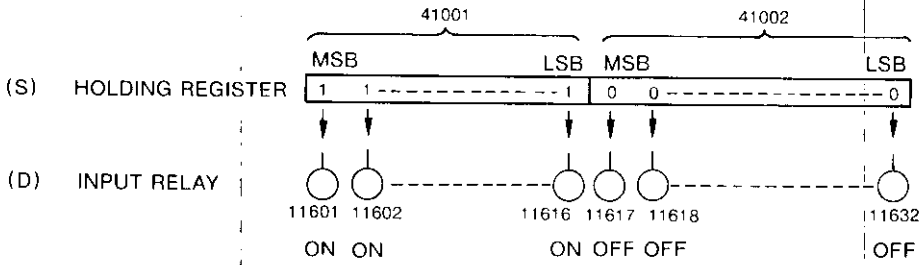


(a) Ladder Circuit

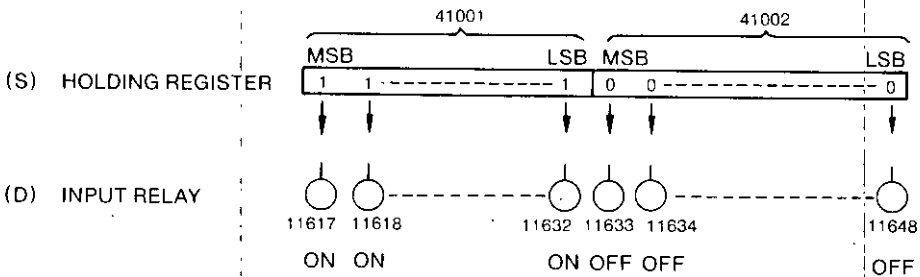


(b) Move Contents

- When $i = 1101$, $D1 = 11601$. If the input relay 10001 is turned ON at this time, the bit patterns of the holding registers 41001 and 41002 are moved to the input relay groups 11601-11632 and the output 1 is turned ON.



- When $i = 1102$, $D1 = 11617$. If the input relay 10001 is turned ON at this time, the bit patterns of the holding registers 41001 and 41002 are moved to the input relay groups 11617-11648 and the output 1 is turned ON.



- When $i = 1128$, $D1 = 12033$. Because the table size is 2, no input relay groups of destination corresponding to the holding register 41002 exist. Therefore, even if the input relay 10001 is turned ON, the move will not be performed and the output 2 is turned ON.

5.10.3 Block Move 2 with Destination Index (DIBR)

(1) Function

This function can transfer all data of source table to the destination table (a group of holding register) specified by the pointer.

(2) Form

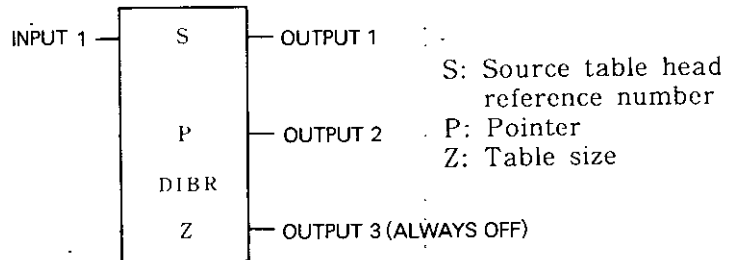


Fig. 5.68 DIBR General Form

- Fig. 5.68 shows the form of the block move 2 with destination index (DIBR).
- DIBR is the symbol denoting the block move 2 with destination index.
- DIBR requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.77, specify any of number for each of the elements.

Table 5.77 DIBR Elements

Element	Description	Specified Number
Top	Source table head reference number	Any of the following: • Coil (00001-08177) • Input relay (10001-14081) • Step relay (S001-S497) • Link coil (D0001-D1009) • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512)
Middle	With pointer, specified head number of destination table	• Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Size of source table and destination table	• For step relay specified in top, constant (1-32) • For link coil specified in top, constant (1-64) • Others (1-100)

NOTE

1. When a relay reference No. is to be specified in the top stage, the lower-place 4 digits (3 digits in case of step relay) of the number that can be specified will be limited to $(16m + 1)$. ($m=0.1.2....$)
2. In case coil or relay is specified in the top stage when transfer of data is executed, data are stored in the registers in the sequence of the latest number starting with the highest order of register.

(3) Operation

- By DIBR, all data of the source table will be moved to the destination table (holding register groups) specified with the value i of the pointer when the input 1 is ON. The output 1 is turned ON. The move will be completed in one scanning cycle.

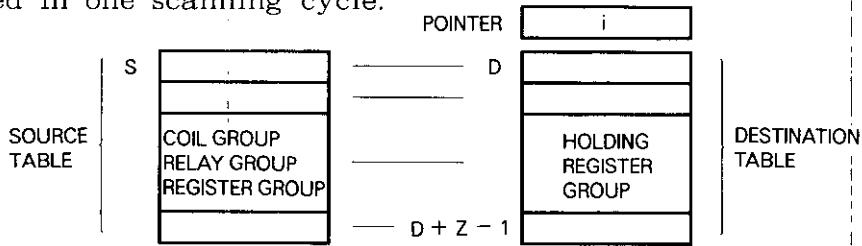
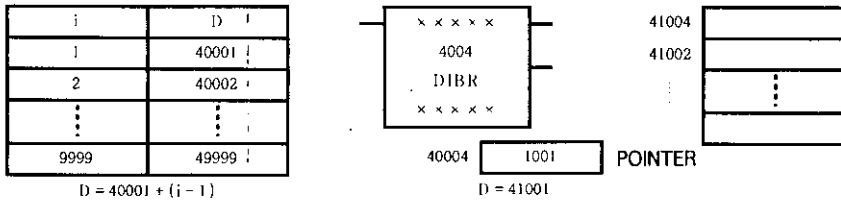


Fig. 5.69 DIBR Function

The following shows the relation between the value i of the pointer and the head number of the holding register group at the destination.



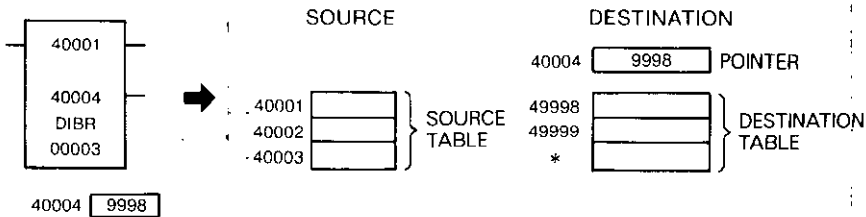
- In the following cases, the move will not be executed and the output 2 is turned ON.

(a) When the value i of the pointer is out of the range of 1-9999.



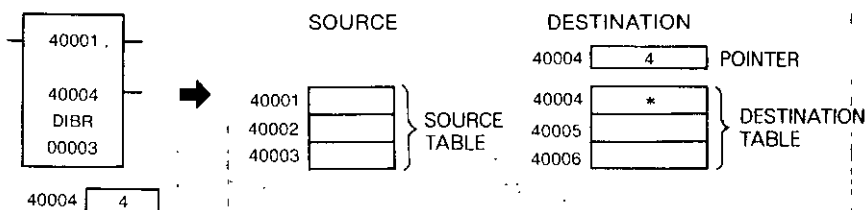
NOTE There is no holding register corresponding to $i = 0$. Therefore, no destination tables corresponding to the source tables 40001-40003 exist.

(b) When the value i of the pointer is within the range of 1-9999 but no destination table to be specified by the value exist:



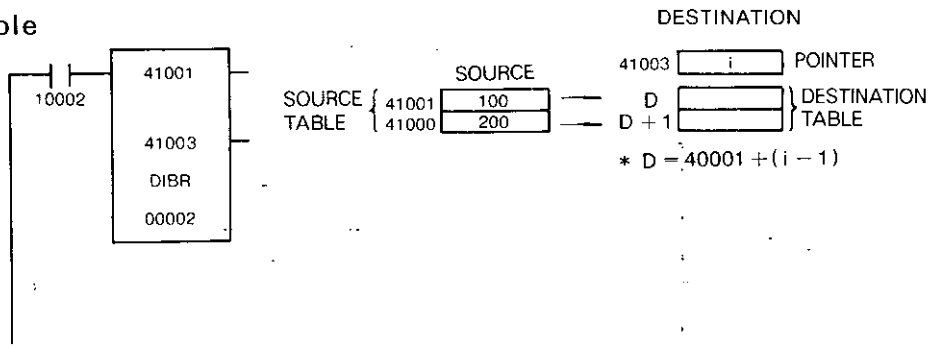
*No destination corresponding to the source 40003 exists.

(c) When the pointer is included in the destination table:

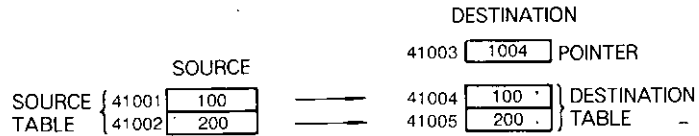


* The pointer 40004 is included in the destination table.

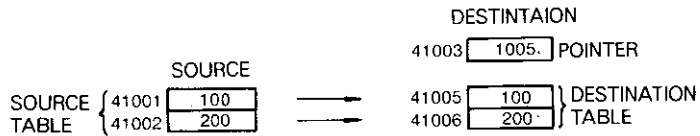
(4) Example



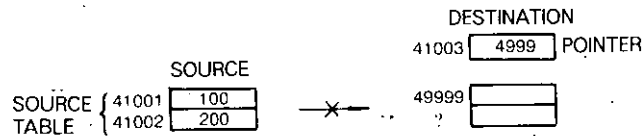
(a) When $i = 1004$, $D1 = 41004$. If the input relay 10002 is turned ON at this time, the contents of the holding registers 41001 and 41002 is moved to the holding registers 41004 and 41005, and the output 1 is turned ON.



(b) When $i = 1005$, $D1 = 41005$. If input relay 10002 is turned ON at this time, the contents of the holding registers 41001 and 41002 is moved to the holding registers 41005 and 41006, and the output 1 is turned ON.



(c) When $i = 9999$, $D1 = 49999$. Because the table size is 2, no holding register of destination corresponding to the source holding register 41002 exists. Therefore, even if the input relay 10002 is turned ON, the move will not be performed and the output 2 is turned ON.



(d) When $i = 1003$, $D1 = 41003$. Because the pointer 41003 is included in the destination table, the move will not be performed and the output 2 is turned ON even if the input relay 1002 is turned ON.



5.10.4 Block Move 1 with Source Index (SIBT)

(1) Function

This function transfers all the data of the source table (coil table, relay table, register table) specified by the pointer content (i) to the destination table (register table) in 1 scan.

(2) Form

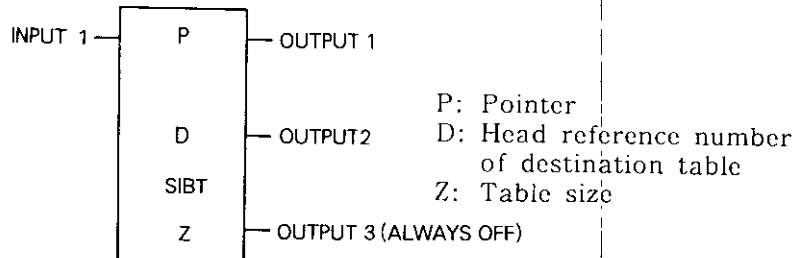


Fig. 5.70 SIBT General Form

- Fig. 5.70 shows the form of the block move 1 with source index (SIBT).
- SIBT is the symbol denoting the block move 1 with source index.
- SIBT requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.78, specify either constant or reference number of various registers for each of the elements.

Table 5.78 SIBT Elements

Element	Description	Specified Number
Top	Using pointer, head number of the source table is specified	<ul style="list-style-type: none"> • Holding register (40001-49999) • Link register (R0001-R1024)
Middle	Head number of the destination table	<ul style="list-style-type: none"> • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-100)

(3) Operation

By SIBT, all data of the source table (coils, input relays, input register groups) specified by the value i of the pointer will be moved to the destination table (holding register groups) when the input 1 is ON. The output 1 is turned on. The move will be completed in one scanning cycle. Table 5.79 shows relation between the value i of the pointer and the source table.

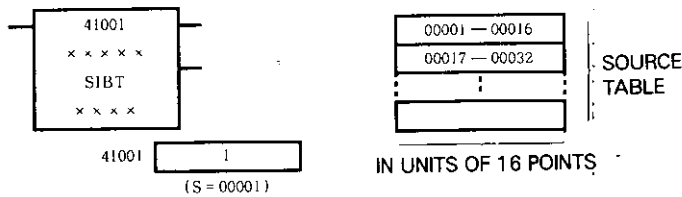
Table 5.79 Relation between Pointer i and Source Table

Coil		Input Relay		Input Register	
i	S	i	S	i	S
1	00001	1001	10001	3001	30001
2	00017	1002	10017	3002	30002
⋮	⋮	⋮	⋮	⋮	⋮
512	08177	1256	14081	3512	30512

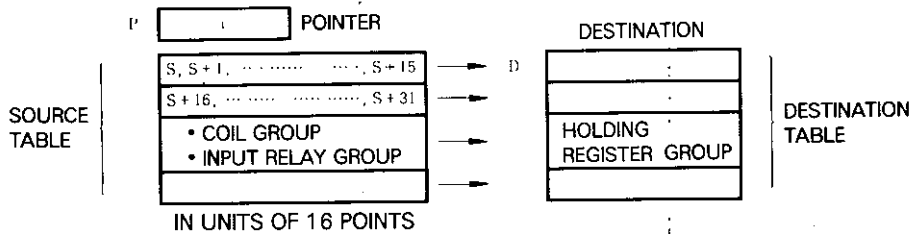
$S = 1 + 16(i-1)$ $S = 1001 + 16(i-1001)$ $S = 30001 + (i-3001)$

5.10.4 Block Move 1 with Source Indx (SIBT) (Cont'd)

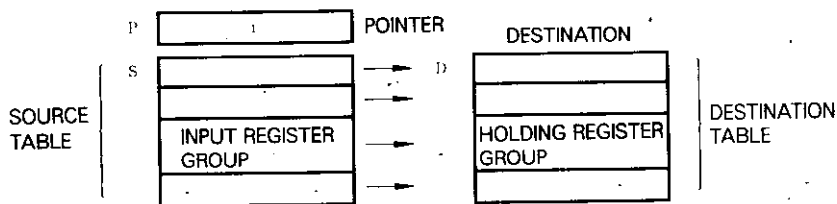
Example:



(a) When the source table is coil and input relay groups:



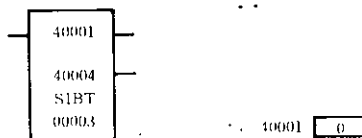
(b) When the source table is input register groups:



- In the following cases, the move will not be executed and the output 2 is turned ON.

(a) When the value i of the pointer is out of the ranges of 1-512, 1001-1256, and 3001-3512:

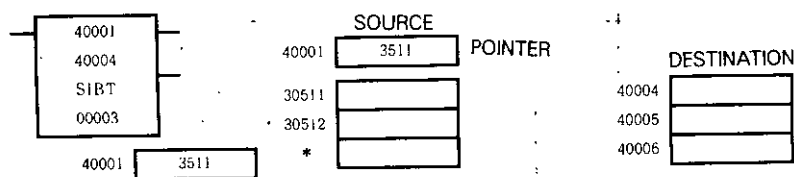
Example:



NOTE There are no coil group, input relay group and input registers corresponding to $i = 0$. Therefore, no source table corresponding to the destination tables 40004-40006 exist.

(b) When the value i of the pointer is within the ranges of 1-512, 1001-1256, and 3001-3512 but no source table specified by the value exists:

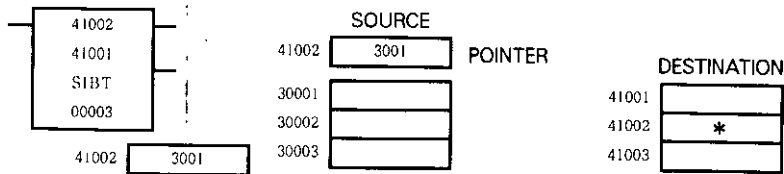
Example:



*No source input register corresponding to the destination 40006 exists.

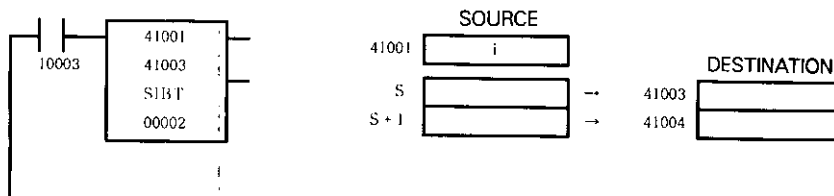
(c) When the pointer is included in the destination table:

Example:

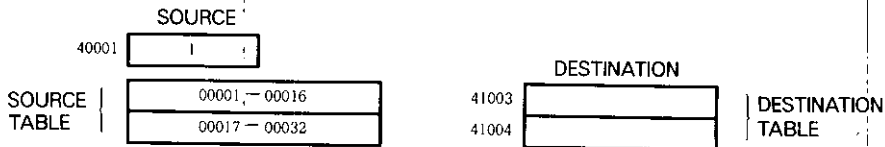


*The pointer 41002 is included in the destination table.

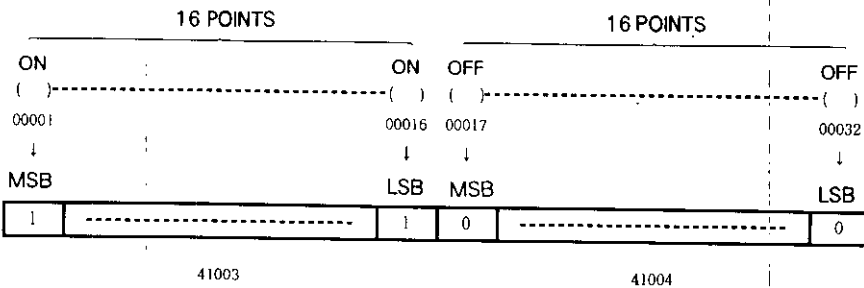
(4) Example



(a) When $i = 1$, $S1 = 00001$. If the input relay 10003 is turned ON at this time, the ON/OFF status of the coil groups 00001-00032 is moved to the holding registers 41003 and 41004 and the output 1 is turned ON.



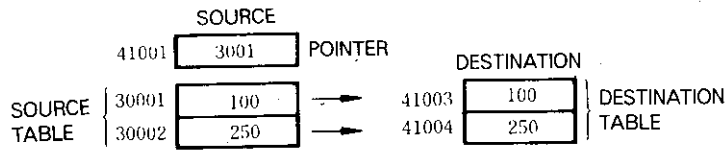
(a) Coil



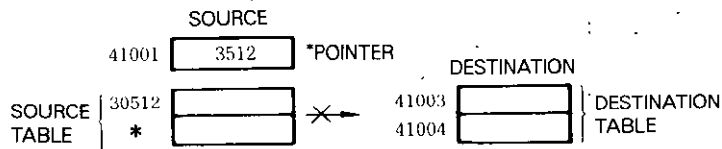
(b) Holding Register

(b) When $i = 3001$, $S1 = 30001$. If the input relay 10003 is turned ON at this time, the contents of the input registers 30001 and 30002 is moved to the holding registers 41003 and 41004, respectively.

5.10.4 Block Move 1 with Source Index (SIBT) (Cont'd)



(c) When $i = 3256$, $S = 30256$. Because the table size is 2, no source input register corresponding to the destination holding register 41004 exists. Therefore, even if the input relay 10003 is turned ON, the move will not be performed and the output 2 is turned ON.



* No source input register corresponding to the destination 41004 exists.

5.10.5 Block Move 2 with Source Index (SIBR)

(1) Function

This function can transfer all data of source table (holding register group) specified by the value i of the pointer to the destination table (holding register group).

(2) Form

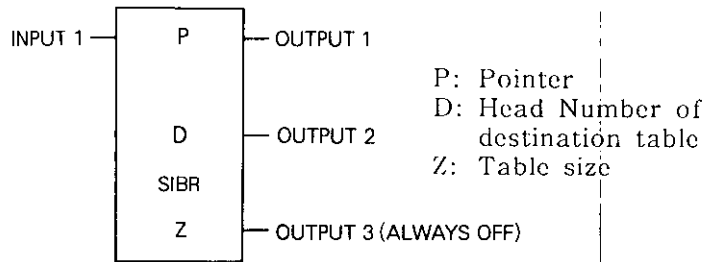


Fig. 5.71 SIBR General Form

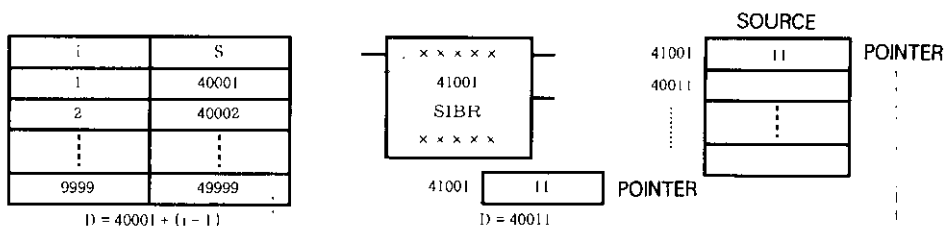
- Fig. 5.71 shows the form of block move 2 with source index (SIBR).
- SIBR is the symbol denoting the block transfer 2 with source index.
- SIBR requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.80, specify the needed number for each of the elements.

Table 5.80 SIBR Elements

Element	Description	Specified Number
Top	With pointer, specified head number of source table.	• Holding register (40001-49999) • Link register (R0001-R1024)
Middle	Head number of destination table	• Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	Constant (1-100)

(3) Operation

The following figure shows relation between the value i of the pointer and the head number of source table (holding register group).

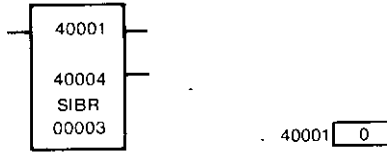


5.10.5 Block Move 2 with Source Index (SIBR) (Cont'd)

- In the following cases, the move will not be executed and the output 2 is turned ON.

(a) When the value i of the pointer is out of the range of 1-9999:

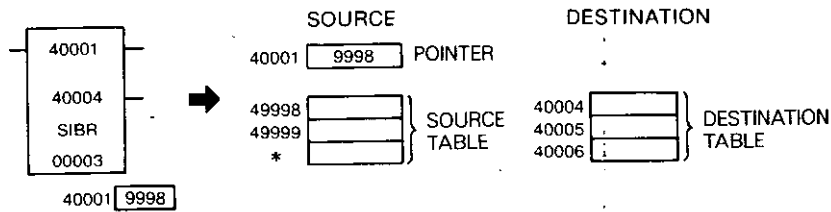
Example:



NOTE No holding register corresponding to $i = 0$ exists and therefore no source table corresponding to the destination table 40004-40006 exists.

(b) When the value i of the pointer is within the range of 1-9999 but no source table specified by the value exists:

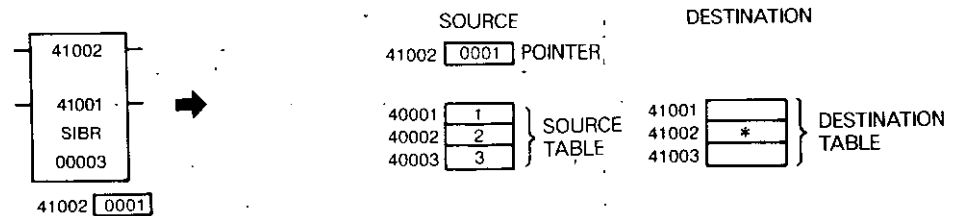
Example:



* No source holding register corresponding to the destination 40006 exists.

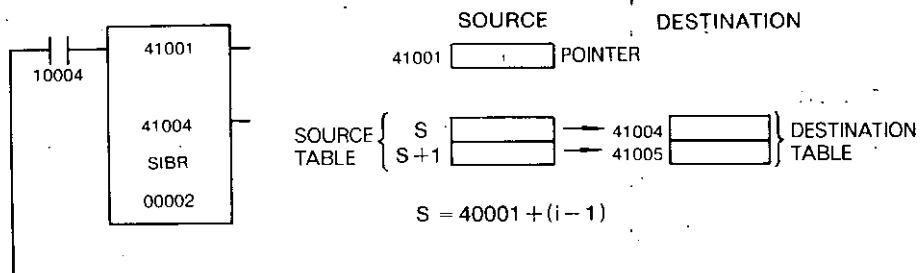
(c) When the pointer is included in the destination table:

Example:

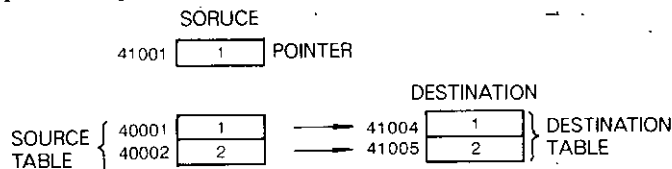


* The pointer 41002 is included in the destination table.

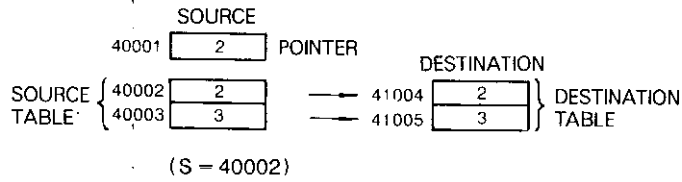
(4) Example



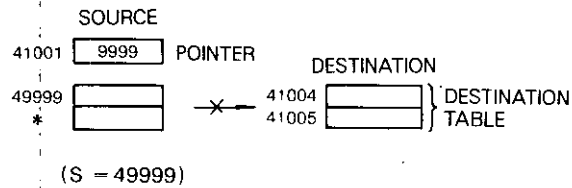
- When $i = 1$, $S = 40001$. If the input relay 10004 is turned ON at this time, the contents of the holding registers 40001 and 40002 is moved to the holding registers 41004 and 41005, respectively and the output 1 is turned ON.



- When $i = 2$, $S = 40002$. If the input relay 10004 is turned ON at this time, the block move will be performed as follows.



- When $i = 9999$, $S = 49999$. Because the table size is 2, no holding register corresponding to the destination 41005 exists. Therefore, even if the input relay 10004 is turned ON, the move will not be performed and the output 2 is turned ON.

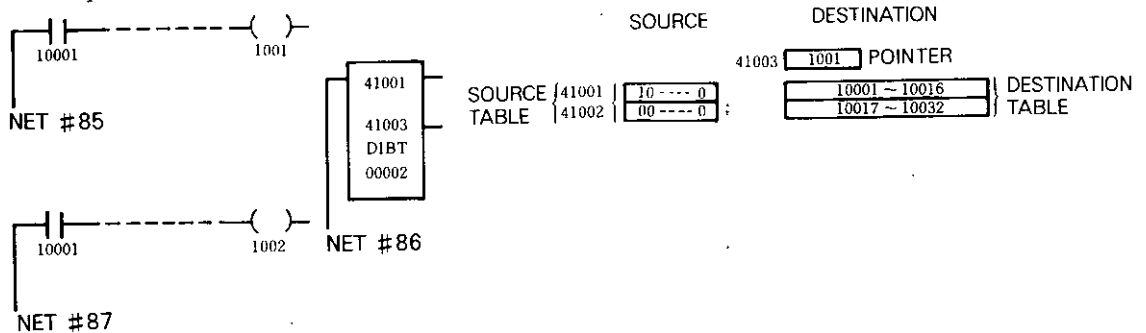


*No source holding register corresponding to the destination 41005 exists.

5.10.6 Programming Indexed Block Move Circuit and Precautions

- (1) Inputs to the indexed block move circuit may be outputs of relays, timers, counters, arithmetic operations, data transfer matrixes, and other indexed block move circuits.
- (2) Coils need not be connected to two output nodes, (1 and 2) of an indexed block move circuit. It is permitted to connect a relay contact to the output nodes at right or connect the output node directly to an input node of an arithmetic circuit, except relays.
- (3) To execute the move constantly, connect the input directly to the power rail at left. To execute it only during one scanning cycle, use a transitional contact as an input.
- (4) The range of the source or destination table specified by a table size must be within the range of the reference numbers of input relays, coils, or registers.
- (5) It is possible to OR the outputs by connecting a vertical shunt element.
- (6) The source or the pointer remain unchanged data or values after the move.
- (7) Be sure to disable the input relay groups used as destination of the block move 1 with destination index (DIBT) (DISABLE ON or DISABLE OFF). There are following differences between the cases the input relay groups are disabled and enabled.
 - (a) When the input relay groups are disabled:
The input relays are turned ON and OFF according to the results of execution of DIBT.
 - (b) When the input relay groups are enabled:
The input relays are turned ON and OFF according to the results of execution of DIBT then all of them are turned OFF at the beginning of the next scanning cycle. But, if an input module is connected and I/O allocation is made, they are turned ON and OFF according to the actual status of input signals at the beginning of the next scanning cycle.

Example:



NOTE If the input relay 10001 is disabled, the each normally open (NO) contact of the input relay 10001 of NET #85 and #87 is turned ON. If the input relay 10001 is enabled, the NO contact of the input relay 10001 of NET #85 is turned off and that of NET #87 is turned ON.

- (8) Do not use transitional contacts of input relay group used in DIBT destination. Because of the followings:
 - (a) $\overline{1} \text{---} | \text{---} 1$ provides a function of $\overline{1} \text{---} | \text{---} 1$.
 - (b) $\overline{1} \text{---} | \text{---} 1$ keeps always OFF.

5.11 DATA CONVERSION

Data conversion is designed to swap, sort, compose, split and/or convert the data in the data table.

5.11.1 Types of Data Conversion

There are 7 types of data conversion as follows:

Table 5.81 Types of Data Conversion

Type	Symbol	Functions	Reference Page
BCD→Binary Conversion	BIN	Converted to binary	183
Binary→BCD Conversion	BCD	Converted to BCD	186
Swap	SWAP	Replacement of high-order byte and low-order byte.	189
Sort	SORT	Rearrangement of data in UP order and DOWN order.	191
Byte Split	BYSL	Splitting of word data into byte data.	194
Byte Composition	BYCM	Composition of byte data into word data.	196
Block Addition	BADD	Addition in block unit.	198

5.11.2 BCD → Binary Conversion (BIN)

(1) Function

This function converts the content of source table represented in BCD into a binary number in 1 scan cycle and transfers it to the destination.

(2) Form

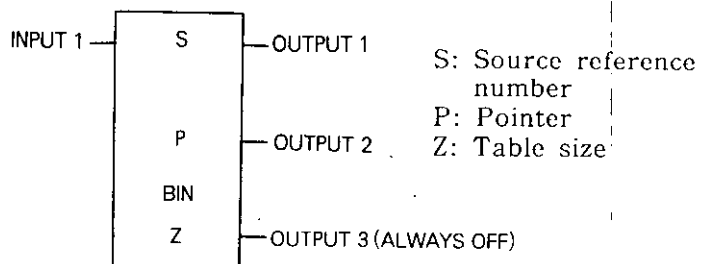


Fig. 5.72 BIN General Form

- Fig. 5.72 shows the form of BCD → BIN conversion.
- BIN is the symbol denoting BCD → BIN conversion.
- BCD → BIN conversion requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.82, specify the needed number for each element.

5.11.2 BCD → Binary Conversion (BIN) (Cont'd)

Table 5.82. BIN Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) • Link register (R0001-R1024)
Middle	Pointer	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-16)

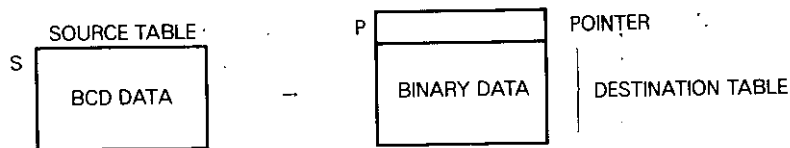
- NOTE**
1. Destination reference number starts with the next to pointer reference number.
 2. Table size does not contain the pointer.

(3) Operation

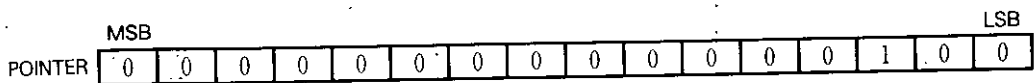
When the input 1 is in the ON state, this function converts all of the data of source table from BCD into binary number and transfers it to the destination table.

When the data of source table are not BCD data, the function indicates the order of that data counted from the head as the order counted from below the pointer (or from LSB). The conversion is performed down to the end.

The data which are not BCD data (the number exceeding 9999) are tentatively converted as BCD data into a binary number and stored in the destination table, but the data are not correct.



If the data 3rd from the head of source table are not BCD table, the content of the pointer after the conversion will be as follows:



The output 1 will become ON when the input 1 is ON and will become OFF (Copy of the input 1) when the input 1 is OFF.

The output 2 will become ON when there are any data other than BCD data in the source table.

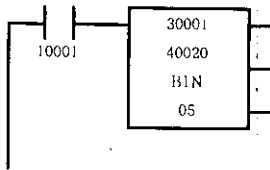
The output 3 is always OFF.

The operations of BIN are tabulated in Table 5.83.

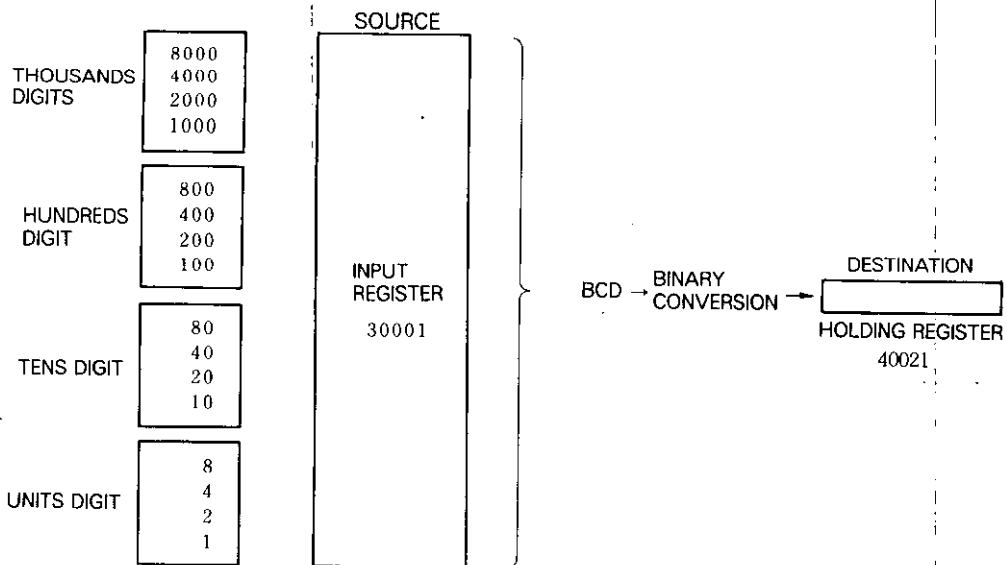
Table 5.83 BIN Operations

Input 1	Operations	Output 1	Output 2
ON	No.15 bit (8000)	ON	OFF
	No.14 bit (4000)		
	No.13 bit (2000)		
	No.12 bit (1000)		
	No.11 bit (800)		
	No.10 bit (400)		
	No.9 bit (200)		
	No.8 bit (100)		
	No.7 bit (80)		
	No.6 bit (40)		
	No.5 bit (20)		
	No.4 bit (10)		
	No.3 bit (8)		
No.2 bit (4)			
No.1 bit (2)			
No.0 bit (1)			
	BCD→binary conversion		
	Converted (other type of data in source table...not BCD data)	ON	ON
OFF	Not operated.	OFF	OFF

(4) Example

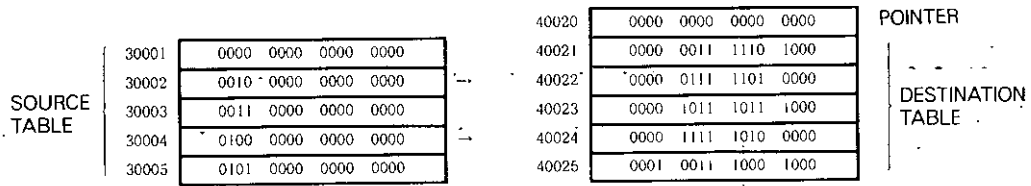


(a) Ladder



(b) Input Conversion (using 30001)

5.11.2 BCD → Binary Conversion (BIN) (Cont'd)



(c) Content of Conversion

BIN shown in (a) will execute the conversion shown in (C) throughout the duration when input relay 10001 is in the ON state.

BIN is mainly used to convert BCD data in the input register that are input, if the input device is a device of BCD representation.

Suppose that the data of 30004 are not BCD data, 0110 1111 0101 0000, this function will place 1 in the 4th bit counted from below 40020 of the pointer and continue conversion down to the last data of the source table. The data of 40024 will be tentatively converted into a binary number, but the data will not be correct. From the value of the pointer, it can be determined in which register of the source table the data which are not BCD data are stored. As described above, when certain data which are not BCD data are included in the source table, the output 2 will turn ON.

5.11.3 Binary → BCD Conversion (BCD)

(1) Function

This function converts the content of source table represented in binary into a BCD in 1 scan and transfers it to the destination.

(2) Form

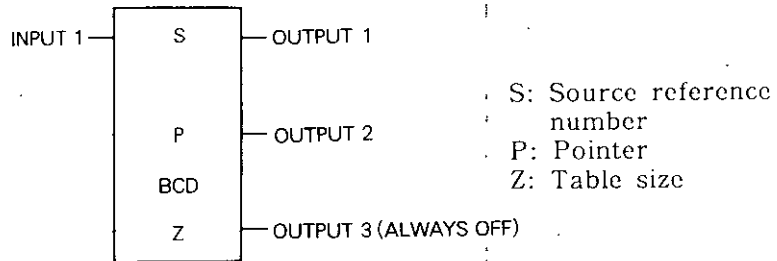


Fig. 5.73 BCD General Form

- Fig. 5.73 shows the form of Binary → BCD conversion.
- BCD is the symbol denoting Binary → BCD conversion.
- Binary → BCD conversion requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.84, specify the needed number for each element.

Table 5.84 BCD Elements

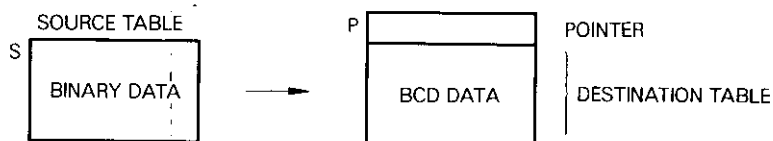
Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) • Link register (R0001-R1024)
Middle	Pointer	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-16)

- NOTE**
1. Destination reference number starts with the next to pointer reference number.
 2. Table size does not contain the pointer.

(3) Operation

When the input 1 is in the ON state, this function converts all of the data of source table from Binary into BCD and transfers it to the destination table.

When the data of source table are not of BCD conversion range (the number exceeding 9999 or negative number), the function indicates the order of that data counted from the head as the order counted from below the pointer (or from LSB). The conversion is performed down to the end. The data are tentatively converted into a BCD and stored in the destination table, but the data are not in BCD.



If the data 3rd from the head of source table are not binary table, the content of the pointer, after the conversion will be as follows:

	MSB																		LSB		
POINTER	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

The output 1 will become ON when the input 1 is ON and will become OFF (Copy of the input 1) when the input 1 is OFF.

The output 2 will become ON when there are any data other than binary data in the source table.

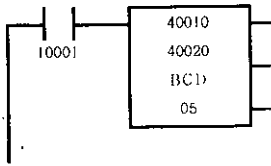
The output 3 is always OFF.

The operations of BCD are tabulated in Table 5.85.

Table 5.85 BCD Operations

Input 1	Operations		Output 1	Output 2
ON	Binary → BCD Conversion	No.15 bit (8000) No.14 bit (4000) No.13 bit (2000) No.12 bit (1000) No.11 bit (800) No.10 bit (400) No.9 bit (200) No.8 bit (100) No.7 bit (80) No.6 bit (40) No.5 bit (20) No.4 bit (10) No.3 bit (8) No.2 bit (4) No.1 bit (2) No.0 bit (1)	ON	OFF
		Converted (other type of data in source table...not binary data)	ON	ON
OFF	Not operated.		OFF	OFF

(4) Exmample



(a) Ladder

SOURCE TABLE	
40010	1000 0000 1010 0000
40011	0000 1000 0000 0000
40012	0000 0000 0000 1111
40013	0000 0000 0001 0000
40014	0000 0001 0000 0000

DESTINATION		POINTER
40020	0000 1000 0000 0000	
40021		
40022		
40023	ANY PATTERN	
40024		
40025		

① Before Conversion

SOURCE TABLE		
40010	1000 0000 1010 0000	(-160)
40011	0000 1000 0000 0000	(2048)
40012	0000 0000 0000 1111	(15)
40013	0000 0000 0001 0000	(16)
40014	0000 0001 0000 0000	(256)

DESTINATION		POINTER
40020	0000 0000 0000 0001	
40021	WRONG DATA	
40022	0010 0000 0100 1000	
40023	0000 0000 0001 0101	
40024	0000 0000 0001 0110	
40025	0000 0010 0101 0110	

② After Conversion

(b) Content of Conversion

BCD shown in (a) will execute the conversion shown in (b) throughout the duration when input relay 10001 is in the ON state.

BCD is mainly used to output the internal operation result after converting its binary data into BCD, if the output device is a device of BCD representation.

Since the content of 40010 in the source table (A negative number is anticipated for this data) is, if converted into BCD; a 5-digit data of 32928 which does not come into the register, incorrect data will be stored. In this case, the function places 1 in the least significant bit of pointer 40020 and continues the conversion down to the last data of the source table. From the value of the pointer, it can be determined in which register of the source table the wrong data are stored. As described above, when any data which cannot be converted into a 4-digit BCD are included in the source table, the output 2 will turn ON.

5.11.4 Swap (SWAP)

(1) Function

It stores the content of source table in the destination, after replacing the high-order byte with the low-order byte.

(2) Form

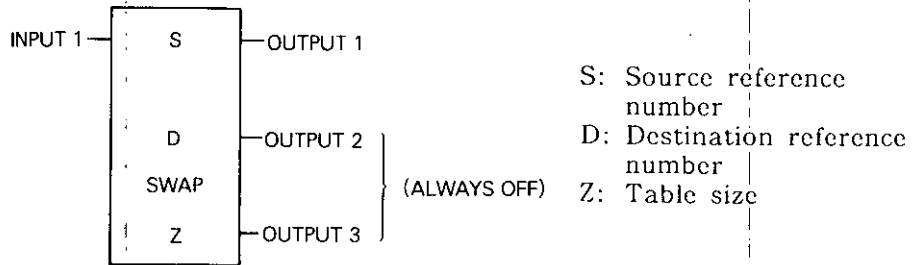


Fig. 5.74 SWAP General Form

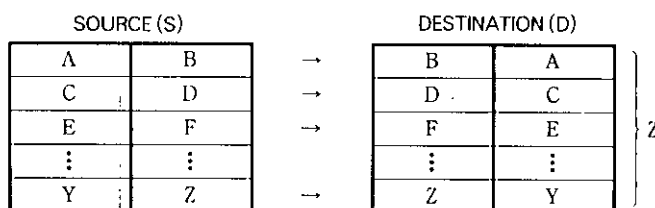
- Fig. 5.74 shows the form of swap.
- SWAP is the symbol denoting swap.
- SWAP requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.86, specify the needed number for each element.

Table 5.86 SWAP Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024)
Middle	Destination reference number	<ul style="list-style-type: none"> • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-100)

(3) Operation

When the input is ON, the function stores the content of source table after replacing the high-order byte with the low-order byte. It converts and transfers the whole table in 1 scan cycle.



5.11.4 Swap (SWAP) (Cont'd)

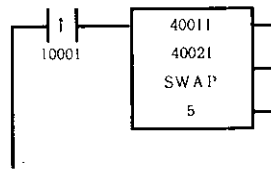
(a) Inputs

The input 1 executes SWAP and stores the result in the destination table. The inputs 2 and 3 are not used.

(b) Outputs

The operation of the output 1 is the same as the input 1 ON/OFF status. The outputs 2 and 3 are always OFF.

(4) Example



(a) Ladder

SOURCE					DESTINATION				
40011	0101	1111	0000	1111	40021	ANY PATTERN			
40012	1111	1111	0000	0000	40022				
40013	1010	1010	0101	0101	40023				
40014	0001	1000	0100	0010	40024				
40015	1000	1100	0011	0011	40025				

① Before Swap

SOURCE					DESTINATION				
40011	0101	1111	0000	1111	40021	0000	1111	0101	1111
40012	1111	1111	0000	0000	40022	0000	0000	1111	1111
40013	1010	1010	0101	0101	40023	0101	0101	1010	1010
40014	0001	1000	0100	0010	40024	0100	0010	0001	1000
40015	1000	1100	0011	0011	40025	0011	0011	1000	1100

② After Swap

(b) Content of Swap

This function is effective for data rearrangement in use of Memobus data communication.

5.11.5 Sort (SORT)

(1) Function

It stores the content of source table in the destination table after rearranging it either in the order from the largest value down or in the order from the smallest value up. Or, with the source table as an index, it rearranges the destination table in the order from the largest value down or on the order from the smallest value up.

(2) Form

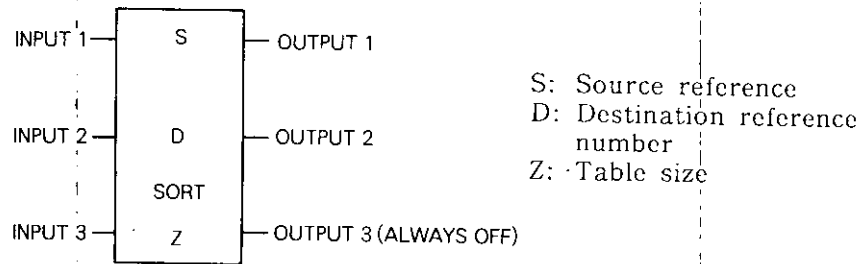


Fig. 5.75 SORT General Form

- Fig. 5.75 shows the form of sort.
- SORT is the symbol denoting sort.
- SORT requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.87, specify the needed number for each element.

Table 5.87 SORT Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024)
Middle	Destination reference number	<ul style="list-style-type: none"> • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-100)

- NOTE**
1. If the input 3 is OFF; when source table is equal to destination table, the operation can be performed correctly. However, when they overlap, no correct operation can be performed.
 2. If the input 3 is ON; when source table and destination table overlap, no correct operation can be performed. When source table is equal to destination table, the output 2 turns ON, and no operation is performed.

(3) Operation

The function transfers the content of source table to the destination table after sorting it either in the order from the largest value down or in the order from the smallest value up. Or, it sorts the destination table in the order from the largest value down or in the order from the smallest value up, and at the same time, it sorts the source table in the same way.

It rearranges all tables in 1 scan. Note that the table cannot be sorted correctly when there is any negative number in the data.

(a) Inputs

- Input 1: Executes SORT when it is ON.
- Input 2: Sorts the data in the order from the largest value down when it is ON and in the order from the smallest value up when it is OFF.
- Input 3: When it is ON, it makes destination table the subject of SORT using source table as an index. When it is OFF, it makes the content of source table the subject of SORT and transfers the result of SORT to the destination table.

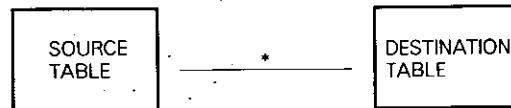
(b) Outputs

- Output 1: Same as the status of the input 1. (Copy of the input 1)
- Output 2: It is ON when the input 3 is ON and source table is same as destination table, and in this case, no operation is performed.
- Output 3: Always OFF.

When the input 3 is ON:



When the input 3 is OFF:



*Data are stored in the order from the smallest value up or in the order from the largest value down.

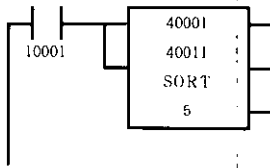
The operations of SORT are tabulated in Table 5.88.

Table 5.88 SORT Operations

Input 1	Input 2	Input 3	Operations	Output 1	Output 2
ON	ON	ON	Indexed SORT in the order from the largest value down.	ON	OFF
		OFF	Transfers data to the destination after sorting in the order from the largest value down.		
	OFF	ON	Indexed SORT in the order from the smallest value up.		
		OFF	Transfers data to the destination in the order from the smallest value up.		
OFF	—	ON	SORT and destination table coincide with each other.	ON	
OFF	—	—	Not operated.	OFF	OFF

(4) Example

Example 1:



(a) Ladder

SOURCE TABLE	
40001	100
40002	200
40003	300
40004	400
40005	500

DESTINATION TABLE	
40011	
40012	
40013	ANY PATTERN
40014	
40015	

① Before Sort

SOURCE TABLE	
40001	100
40002	200
40003	300
40004	400
40005	500

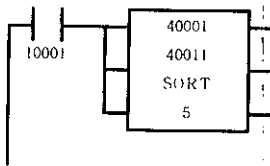
DESTINATION TABLE	
40011	500
40012	400
40013	300
40014	200
40015	100

② After Sort

(b) Content of Sort

The SORT shown in (a) executes the SORT which transfers the source table to destination table in the order from the largest value down, because the inputs 1 and 2 will turn ON when input relay 10001 turns ON.

Example 2:



(a) Ladder

SOURCE TABLE	
40001	1
40002	2
40003	3
40004	4
40005	5

DESTINATION TABLE	
40011	50
40012	3000
40013	10
40014	1
40015	200

① Before Sort

SOURCE TABLE	
40001	2
40002	5
40003	1
40004	3
40005	4

DESTINATION TABLE	
40011	3000
40012	200
40013	50
40014	10
40015	1

② After Sort

(b) Content of Sort

The SORT shown in (a) executes Indexed SORT, because the inputs 1, 2 and 3 turn ON when input relay 10001 turns ON.

5.11.6 Byte Split (BYSL)

(1) Function

It splits the word data of source table into byte data and stores them in destination table.

(2) Form

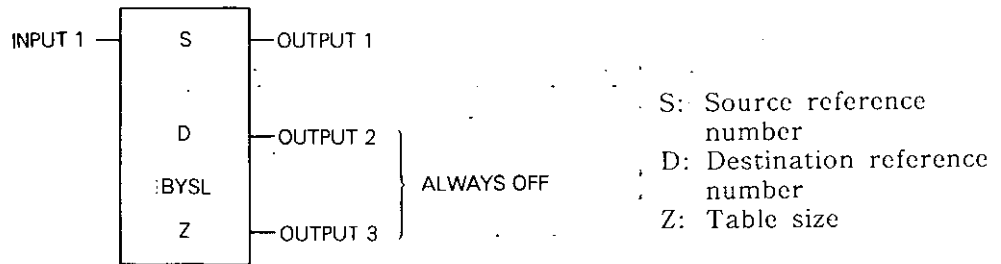


Fig. 5.76 BYSL General Form

- Fig. 5.76 shows the form of byte split.
- BYSL is the symbol denoting byte split.
- BYSL requires three elements placed vertically (top, middle, and bottom). Referring to Table 4.89, specify the needed number for each element.

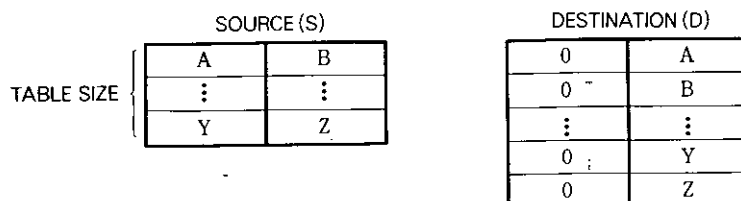
Table 5.89 BYSL Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024)
Middle	Destination reference number	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-100)

(3) Operation

It splits the word data of source table into the high-order byte data and low-order byte data and stores the split data in the destination table. All of the split data are stored in the low-order bytes, and all of the high-order bytes become 0 (zero).

It splits all data in 1 scan cycle and transfers them to the destination table.



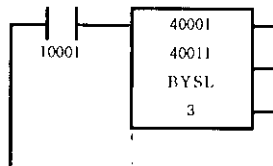
(a) Inputs

- Input 1: Executes byte split and stores the result in the destination table.
- Input 2: } Not used.
- Input 3: }

(b) Outputs

- Output 1: Same as the input 1 ON/OFF status (Copy of the input 1)
- Output 2: } Always OFF.
- Output 3: }

(4) Example



(a) Ladder

SOURCE TABLE				
40001	0001	0000	0010	0000
40002	0011	0000	0100	0000
40003	0101	0000	0110	0000

DESTINATION TABLE	
40011	ANY PATTERN
40012	
40013	
40014	
40015	
40016	

① Before Byte Split

SOURCE TABLE				
40001	0001	0000	0010	0000
40002	0011	0000	0100	0000
40003	0101	0000	0110	0000

DESTINATION TABLE				
40011	0000	0000	0001	0000
40012	0000	0000	0010	0000
40013	0000	0000	0011	0000
40014	0000	0000	0100	0000
40015	0000	0000	0101	0000
40016	0000	0000	0110	0000

② After Byte Split

(b) Content of Byte Split

This function is effective for data conversion required when receiving data from ASCII module.

5.11.7 Byte Composition (BYCM)

(1) Function

It composes the byte data of source table into word data and stores them in the destination table. In other words, it converts the data inversely to BYSL in Par. 5.11.6.

(2) Form

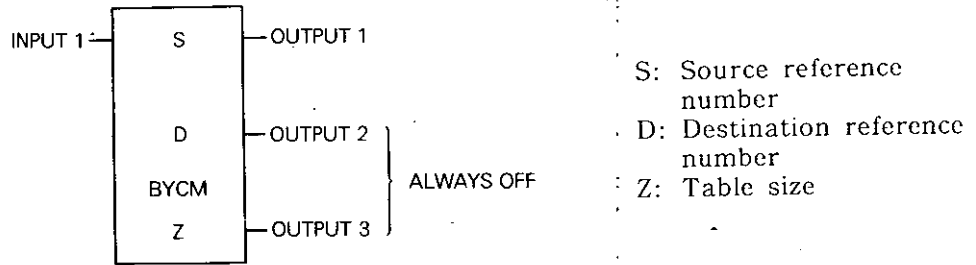


Fig. 5.77 BYCM General Form

- Fig. 5.77 shows the form of byte composition.
- BYCM is the symbol denoting byte composition.
- BYCM requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.90, specify the needed number for each element.

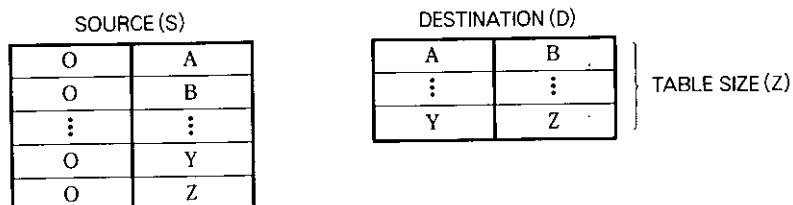
Table 5.90 BYCM Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Input register (30001-30511) • Holding register (40001-49998) • Constant register (31001-35095) • Link register (R0001-R1023)
Middle	Destination reference number	<ul style="list-style-type: none"> • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-100)

(3) Operation

It compounds two continuous byte data of source table into one-word data and stores them in destination table. The byte data of odd numbers of source table become the high-order byte, and the byte data of even numbers become the low-order byte.

All of the byte data of source table are composed into word data in 1 scan cycle and transferred to destination table.



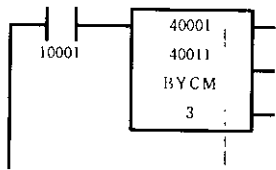
(a) Inputs

Input 1: Executes composition and stores the result in the destination table.
 Input 2: } Not used.
 Input 3: }

(b) Outputs

Output 1: Same as the Input 1 ON/OFF status.
 Output 2: } Always OFF.
 Output 3: }

(4) Example



(a) Ladder

SOURCE TABLE

40001	0000	0000	0001	000
40002	0000	0000	0010	0000
40003	0000	0000	0011	0000
40004	0000	0000	0100	0000
40005	0000	0000	0101	0000
40006	0000	0000	0110	0000

DESTINATION TABLE

40011	ANY PATTERN			
40012				
40013				

① Before Byte Composition

SOURCE TABLE

40001	0000	0000	0001	0000
40002	0000	0000	0010	0000
40003	0000	0000	0011	0000
40004	0000	0000	0100	0000
40005	0000	0000	0101	0000
40006	0000	0000	0110	0000

DESTINATION TABLE

40011	0001	0000	0010	0000
40012	0011	0000	0100	0000
40013	0101	0000	0110	0000

② After Byte Composition

(b) Content of Byte Composition

BYCM shown in (a) executes conversion (b) when input relay 10001 turns ON.

Even when data other than 0 are in the high-order byte of source table, this function executes the operation described above. And the same operation result is obtained.

5.11.8 Block Addition (BADD)

(1) Function

It adds the content of source table into words or bytes and stores the result in destination table.

(2) Form

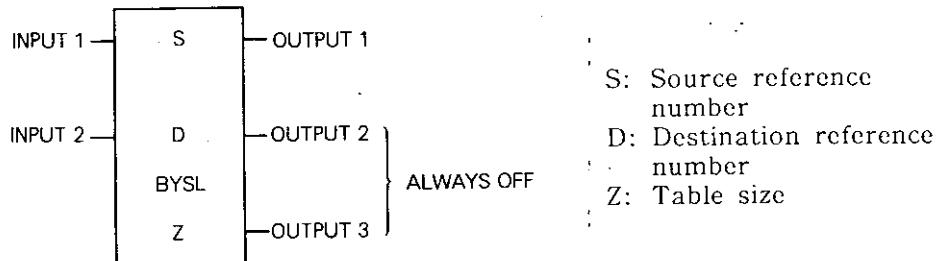


Fig. 5.78 BADD General Form

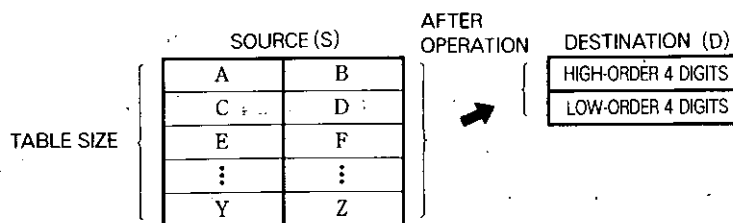
- Fig. 5.78 shows the form of block addition.
- BADD is the symbol denoting block addition.
- BADD requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.91, specify the needed number for each element.

Table 5.91 BADD Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024)
Middle	Destination reference number	<ul style="list-style-type: none"> • Holding register (40001-49998) • Link register (R0001-R1023)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-100)

(3) Operation

When the input 1 turns ON, this function adds the content of source table for each word or for each byte according to ON/OFF of the input 2 and stores the result in destination table. It adds up the whole table in 1 scan cycle.



Even when the table size is 100, operation result will not exceed 99,999,999 and operation will be performed correctly. However, operation will not be performed correctly if any negative number is included in the data.

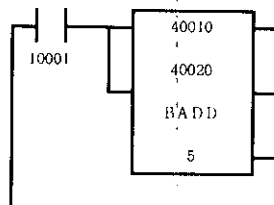
- (a) When the input 2 is ON,
Byte: A + B + C + D + . . . + Y + Z
- (b) When the input 2 is OFF,
Word: AB + CD + . . . + YZ
- (c) The input 3 is not used.
- (d) Outputs
 - Output 1: Same as the status of the input 1. (Copy of the input 1)
 - Output 2: } Always OFF
 - Output 3: }

Table 5.92 shows the operations of BADD.

Table 5.92 Operations of BADD.

Input 1	Input 2	Operation	Output 1
ON	ON	Performs addition per each byte.	ON
	OFF	Performs addition per each word.	
OFF	—	Not operated.	OFF

(4) Example



(a) Ladder

SOURCE TABLE					DESTINATION	
40010	0000	1010	0001	0100		
40011	0001	1110	0010	1000	10	
40012	0011	0010	0011	1100	20	
40013	0100	0110	0101	0000		
40014	0101	1010	0110	0100		

① Before Block Addition

SOURCE TABLE					DESTINATION	
40010	0000	1010	0001	0100		
40011	0001	1110	0010	1000	0	
40012	0011	0010	0011	1100	550	
40013	0100	0110	0101	0000		
40014	0101	1010	0110	0100		

② After Block Addition

(b) Content of Block Addition

BADD shown in (a) performs the following operation, because the input 2 turns ON when input relay 10001 turns ON:

$$10 + 20 + 30 + 40 + 50 + 60 + 70 + 80 + 90 + 100 = 550,$$

and stores the operation result in the destination table.

5.12 MATRIX

This function group allows matrices to be built in consecutively numbered registers. These matrices are similar to tables previously discussed in that they can be groups of input registers or holding registers depending upon the application requirements. In fact, these same registers can also be operated upon by Move function. Whereas the Move functions operate upon individual registers as elements of tables, the matrix function will operate upon bit patterns within the matrix. Since all registers contain 16 bits, the size of a matrix in bits will be even multiples of 16 (e.g., 32, 48, 64, 80, etc.). A bit can have one of two states: ON (one) or OFF (zero). Bits within a matrix each have their own identification as illustrated in Fig. 5.79. A matrix of 100 registers contains 1600 bits.

	MSB					LSB
41001	1,	2,	3,	4,	16
41002	17,	18,	19,	32
41003	33,	34,	48
41004	49,	64

Fig. 5.79 Sample Matrix Bit Numbering

5.12.1 Types of Matrix

Table 5.93 Types of Matrix

Type	Symbol	Reference Page
Logical <u>AND</u>	AND	202
Logical <u>OR</u>	OR	202
Logical <u>Exclusive OR</u>	XOR	202
Logical <u>Complement</u>	COMP	205
Logical <u>Compare</u>	CMPR	207
Logical <u>Bit Modify</u>	MBIT	211
Logical <u>Bit Sense</u>	SENS	213
Logical <u>Bit Rotate</u>	BROT	217
Logical <u>Multiple Bit Rotate</u>	MROT	220
Logical <u>Byte Rearrangement</u>	TWST	223
Logical <u>Bit Count</u>	BCNT	225

5.12.2 Form of Matrix

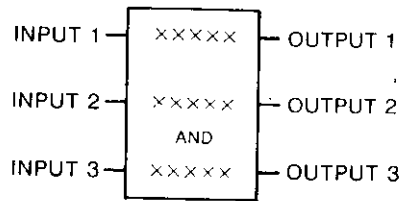


Fig. 5.80 Matrix General Form

The matrix requires three elements placed vertically (top, middle, and bottom), like arithmetic operations and data move. It can be used at any intersection of the 7 lines-by-10 columns matrix (except that the top element cannot be located on lines 6 and 7). AND written between the middle and bottom elements indicates the type of matrix.

NOTE Byte rearrangement (TWST) alone uses two elements (top and bottom places).

(1) Elements and Their Meanings

The top element is called source and the middle destination. The bottom indicates the size of the matrix table. Each has the same meaning as in the data move.

(2) Reference Number

(a) Numbers specified as reference numbers depend on the type of matrix. See paragraph of each matrix type.

(b) The content of a register (16-bit binary) for the register or the ON/OFF status for discrete signals is operated for the matrix, respectively.

NOTE When the discrete I/O is specified as a reference number, $n = 16m + 1$ ($m = 0, 1, 2, \dots$) is required where n is $\times \times \times \times$ of $1 \times \times \times \times$ or $0 \times \times \times \times$.

(3) Table Size

The constant $\times \times \times \times$ is specified as the table size. The range of the fixed value depends on the type of matrix. See paragraph of each matrix type.

NOTE Specify the number of registers (or of sets of 16 discrete signals) but not the number of bits as the table size.

Except for MBIT, SENS, MROT, and BCNT the source and destination tables have the same size.

(4) Pointer

Only CMPR, MBIT, and SENS functions use a pointer. Its function is the same as that described in Par. 5.9 MOVE.

5.12.3 AND, OR, XOR

Each logical function is provided with a similar function block, so these functions are described in this section.

(1) Function

These functions perform logical operations including AND, OR and XOR between source table and destination table and stores the results in the destination table.

(2) Form

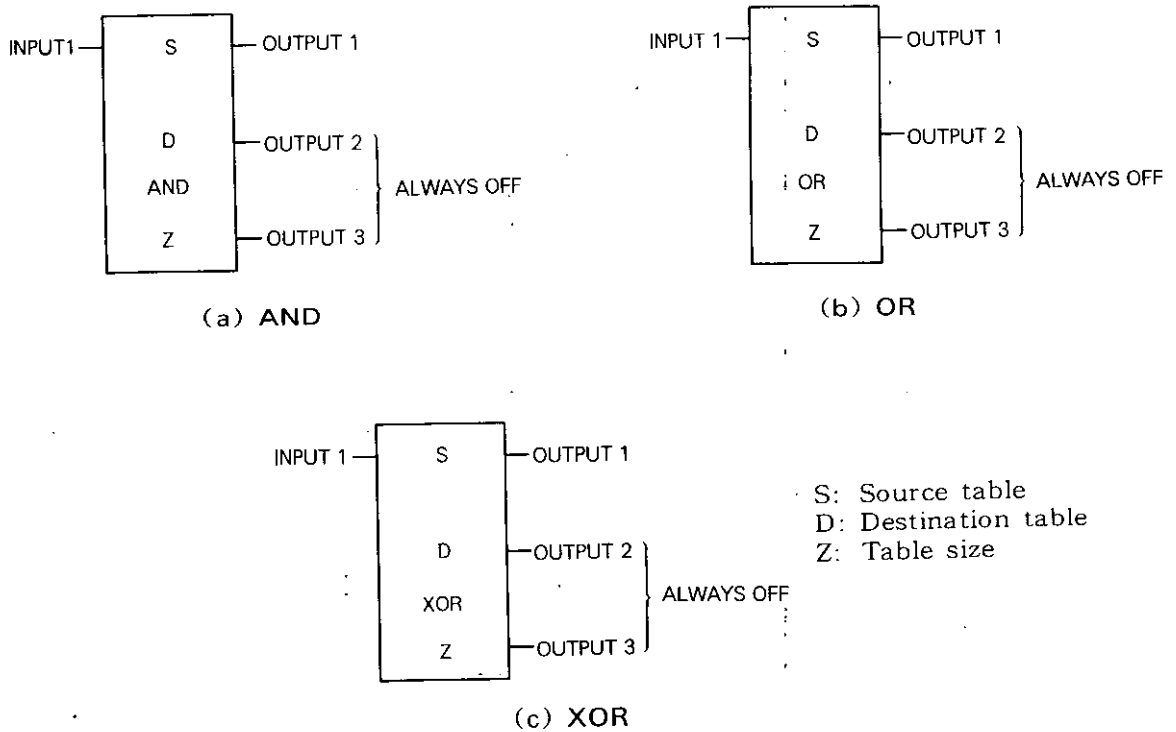


Fig. 5.81 AND, OR, XOR General Forms

- Fig. 5.81 shows each form of AND, OR or XOR.
- AND, OR and XOR are the symbols denoting AND, OR and exclusive-OR, respectively.
- AND, OR and XOR require three elements placed vertically (top, middle, and bottom). Referring to Table 5.94, specify the needed number for each element.

Table 5.94 AND, OR, XOR Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Input relay (10001-14081) • Step relay (S001-S497) • Link coil (D0001-D1009) • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512)
Middle	Destination reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Link coil (D0001-D1009) • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<p>The followings depend on number specified in top or middle.</p> <ul style="list-style-type: none"> • Coil (1-100) • Input relay (1-100) • Step relay (1-32) • Link coil (1-64) • Various registers (1-100)

(3) Operation

Fig. 5.82 shows the truth tables and equivalent relay circuits of AND, OR, and XOR applied to 1 bit signal.

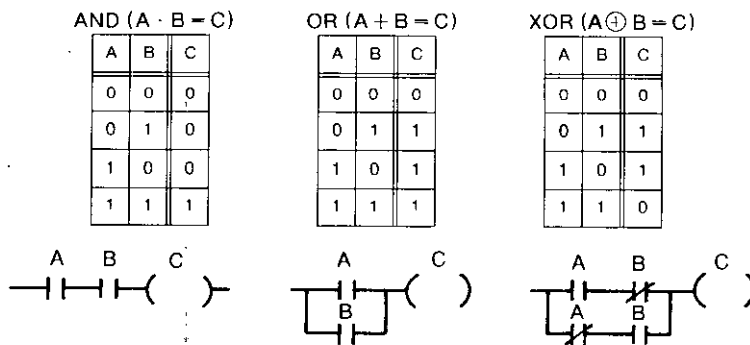
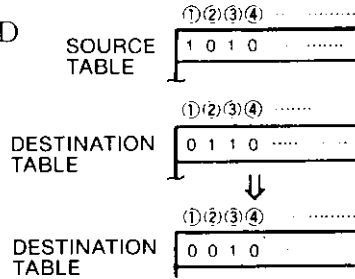


Fig. 5.82 Truth Tables and Equivalent Relay Circuits

This function causes two matrices of equal length to be logically AND'ed, R'ed or XOR'ed together and the results stored for reference by any other logic function. The result of the logical AND will be a one (ON) bit only if both bits (one from each matrix) are one bits; otherwise, the result will be zero (OFF) bit. One bit from each matrix with the same identification number will be combined in accordance to the truth table.

5.12.3 AND, OR, XOR (Cont'd)

Example: AND



NOTE The bit numbers are circled. AND is applied to the bit pair having the same bit number and result is stored in the bit having the same number.

All bits in the matrices will be operated upon every scan the function is enabled. The Source matrix is not altered only copied. The current content of the Destination matrix is used for the AND, OR, or XOR operation and then replaced with the result of the operation. The Destination matrix is altered by the AND, OR, or XOR operation. The Matrix AND operation is useful to clear large groups of registers to zero (when ANDed with a matrix of zeros) or to construct masks within the controller. The Matrix OR operation is useful to construct masks within the controller. The Matrix XOR operation is useful to detect differences between two matrices within one scan and, when operated upon the same matrix in both Source and Destination, to clear a matrix to zero. See Example 2.

(a) Inputs

Only the input 1 is used with these functions. When this input node receives power flow, the each operation is performed. Every scan, when enabled, the AND, OR or XOR will operate upon the entire content of the matrices. Transitional contact can be used if a single operation is desired.

(b) Outputs

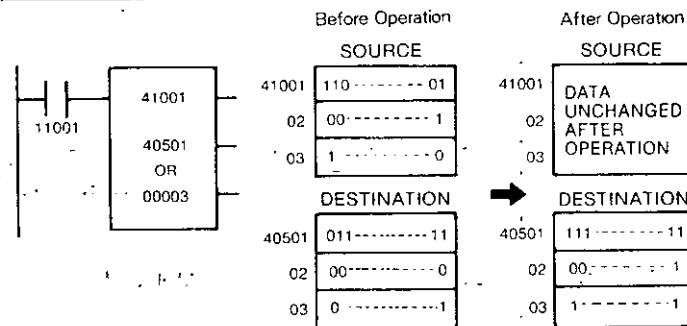
The each matrix function utilizes only the output 1. The lower two outputs have no significance and will be OFF under all conditions. The output 1 will supply power flow whenever the input 1 receives power flow. Thus, the output 1 allows Function Blocks to be cascaded or chained horizontally within a network. Table 5.95 shows operations of AND, OR and XOR.

Table 5.95 Operations of AND, OR and XOR

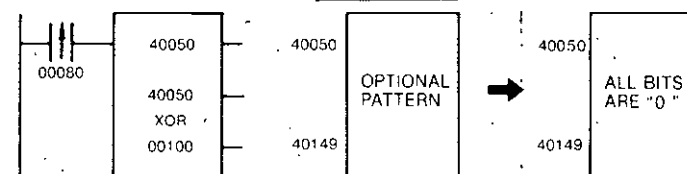
Input 1	Operations	Output 1
ON	AND, OR, XOR operated.	ON
OFF	Not operated.	OFF

(4) Example

Example 1:



Example 2:



This XOR is used to clear the source and destination tables when operated upon the same matrix in both Source and Destination.

5.12.4 Complement (COMP)

(1) Function

This function causes the content of one matrix to be complemented (all ones replaced by zeros, and zeros by ones) and placed in another matrix for reference by any other function. The entire matrix is operated upon every scan the function is enabled by power flow to the input. The result of the Complement operation is placed in the Destination matrix; the previous content of this matrix is lost. The Source matrix is not altered only copied.

The Matrix Complement is useful to alter normally closed inputs to the same base as normally open inputs or to move masks within the controller.

(2) Form

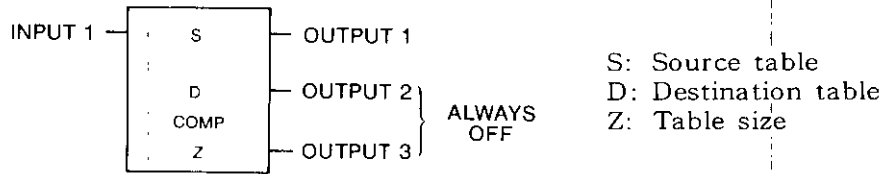


Fig. 5.83 COMP General Form

- Fig. 5.83 shows the form of complement.
- COMP is the symbol denoting complement.
- COMP requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.96, specify the needed number for each element.

Table 5.96 COMP Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Input relay (10001-14081) • Step relay (S001-S497) • Link coil (D0001-D1009) • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512)
Middle	Destination reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Link coil (D0001-D1009) • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<p>The followings depend on number specified in top or middle.</p> <ul style="list-style-type: none"> • Coil (1-100) • Input relay (1-100) • Step relay (1-32) • Link coil (1-64) • Various registers (1-100)

(3) Operation

(a) Inputs

Only the input 1 is used with the COMPLEMENT function. When this input node receives power flow, the COMPLEMENT operation is performed. Every scan, when enabled, the COMPLEMENT will operate upon the entire content of both matrices. Transitional contacts can be used if a single operation is desired.

(b) Outputs

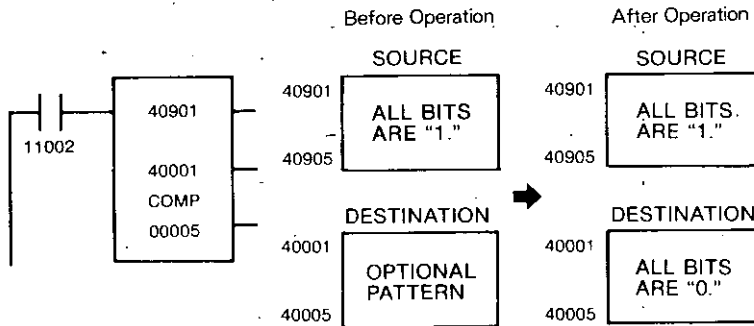
The COMPLEMENT matrix function utilizes only the output 1. The lower two outputs have no significance and will be OFF under all conditions. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows function Blocks to be cascaded or chained horizontally with a network.

Table 5.97 shows operation of COMP.

Table 5.97 Operation of COMP

Input 1	Operations	Output 1
ON	Complemented	ON
OFF	Not complemented.	OFF

(4) Example



5.12.5 Compare (CMPR)

(1) Function

This function causes two matrices to be compared on a bit-by-bit basis; their contents are not altered only examined. When enabled, the compare function will examine one bit from each matrix with the same identification number. If these bits agree (both zero or both ones) the next bit from each matrix is compared; however, if they do not agree, the compare function will halt.

(2) Form

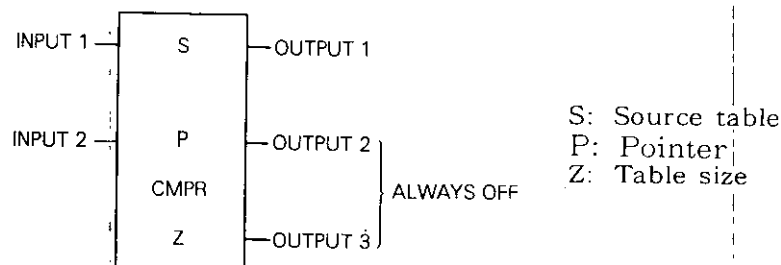


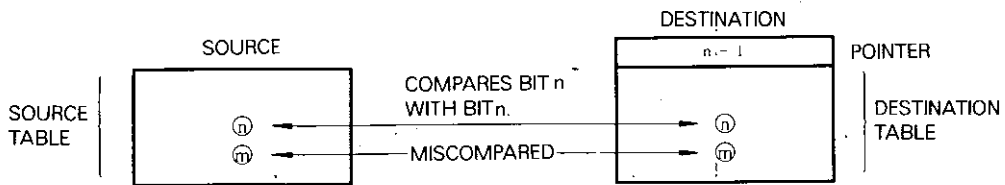
Fig. 5.84 CMPR General Form

- Fig. 5.84 shows the form of compare.
- CMPR is the symbol denoting compare.
- CMPR requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.98, specify the needed number for each element.

Table 5.98 CMPR Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Input relay (10001-14081) • Step relay (S001-S497) • Link coil (D0001-D1009) • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024) • Timer register (50001-50512)
Middle	Pointer	<ul style="list-style-type: none"> • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<p>The followings depend on number specified in top or middle.</p> <ul style="list-style-type: none"> • Coil (1-100) • Input relay (1-100) • Step relay (1-32) • Link coil (1-64) • Various registers (1-100)

(3) Operation



Each scan the compare is performed, either the end of the matrix is located or a miscompare is encountered. If both matrices are identical, the entire length, up to 100 registers or 1600 bits (such as discrete inputs/outputs), is compared each and every scan.

An output is used to indicate the result of compare. It will be ON if a miscompare is detected and OFF if the end of the matrices is reached. A pointer is the only register whose content is altered by the Compare. This pointer is referred to by the Destination block and indicates which specific bit was responsible for the miscompare, if the operation is terminated by a miscompare (output ON). The pointer can also be used to cause the comparison to begin on bits other than at the beginning; the compare function always proceeds towards the end (high bit number) of the matrices. Before the Compare, the pointer will be incremented; if the pointer is at the end of the matrix or longer, it will be reset to one prior to beginning the compare operation.

NOTE

1. The contents of the pointer can be changed by another logic circuit. To start compare at bit i , set $i-1$ to the pointer before starting.
2. If there are no miscompares, the pointer will be at the end of the matrices when the comparison is completed.

(a) Inputs

Only the upper two inputs are used with the COMPARE function. Every scan the input 1 receives power flow, the Compare operation is performed. Up to 1600 pairs of bits can be compared each scan the input is enabled. Transitional contacts can be used if a single operation is desired.

The input 2 controls the reset of the pointer. When this input receives power flow, the pointer will be reset to zero prior to the comparison; otherwise, the comparison begins at the current value of the pointer plus one. If the pointer is at the end of the matrix or greater, it will be reset to zero prior to the comparison regardless of the state of this input. Matrices begin at bit number one (not zero), thus resetting implies the value one.

(b) Outputs

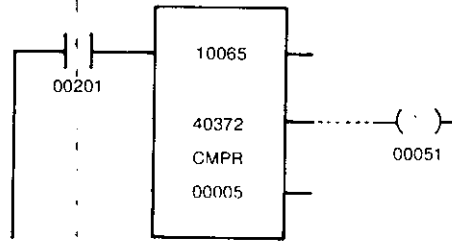
The COMPARE matrix function utilizes all three available outputs. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows Function Block to be cascaded or chained horizontally within a network. The output 2 will supply power flow if the comparison has detected a miscompare; this output will be OFF if either no compare is being performed (top input does not receive power flow), or if no miscompares have been detected and the end of the matrices have been reached. The output 3 senses the Source matrix bit if and only if a miscompare has been detected; this output will supply power flow when this bit is a one and be OFF if the bit is a zero. Using the lower two outputs, logic can be built to determine the exact bit configuration causing a miscompare (Source = one and Destination = zero, or vice versa).

Table 5.99 shows operation of CMPR.

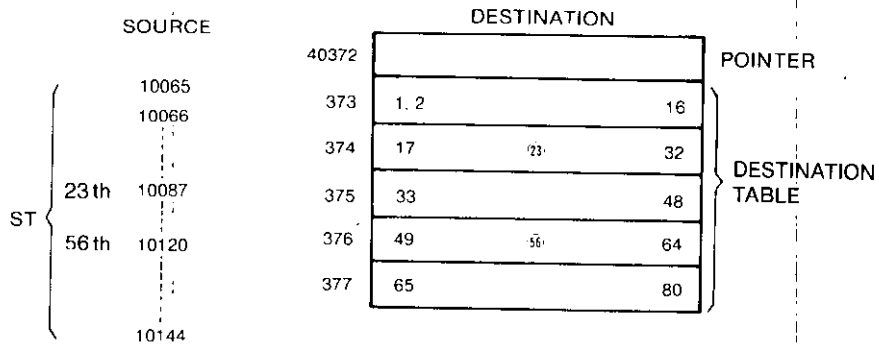
Table 5.99 Shows Operation of CMPR

Input 1	Input 2	Operations	Miscompare	Condition	Output 1	Output 2	Output 3
ON	OFF	Checks from the bit No. next to pointer.	No	—	ON	OFF	OFF
			Yes	S = 1 P = 0		ON	ON
				—		OFF	OFF
	ON	Turns pointer to 0 and checks from the bit No. at the head of table.	No	—		OFF	OFF
			Yes	S = 1 P = 0		ON	ON
				—		OFF	OFF
OFF	OFF	Not operated.	—	—	OFF	OFF	
	ON	Turns pointer to 0. Not operated.	—	—	OFF	OFF	

(4) Example



(a) Ladder



5.12.5 Compare (CMPR) (Cont'd)

Figure above illustrates a typical COMPARE Matrix function. If the pointer (register 40372) contains the value zero or one with input 10056 is energized, the comparison begins at input 10056 and bit 1. The entire matrix all 80 bits will be compared to the inputs unless a miscompare is detected. However, if bit 23 in register 40372 is energized. If input 10087 is ON and bit 23 in the destination matrix (40373-40377) is a zero, coil 00051 will be energized.

On the next scan with input 00201 still energized and the content of register 40372 not altered, the comparison will start at bit 24 and proceed towards the end of the matrix. If input 10120 is OFF and bit 56 set to a one value in the matrix 40373-40377, an additional miscompare is detected. The value in register 40372 is now 0056 and coil 00051 remains ON, 00051 since the Source bit is a zero (input 10120 OFF). The pointer value of the first miscompare is lost and replaced by the location of the second miscompare. To retain miscompare locations, they should be saved in another location, such as a table, with a register to table move function. This move function can be controlled by the coil 00051 in this example.

On the next scan, unless there are additional miscompares, the comparison begins at bit 57 and reaches the end of the matrix. The pointer will contain the value 81, coils 00051 will be OFF. On the next (fourth) scan, the comparison begins again and will detect bit 23 as a miscompare again unless the input or bit status is altered to prevent the miscompare. The compare function is very powerful and very fast. Unless action is taken, pointer values will be replaced by other values and repetitive miscompares detected. At best, all bits are compared every scan when they all agree, and at worst case one bit is compared each scan when they all disagree.

5.12.6 Modify (MBIT)

(1) Function

This function allows individual bits in a matrix to be altered. Only one bit per scan can be affected by this function; all other bits retain their state. Bits can be either set to a one (ON) condition or cleared to a zero (OFF) condition. A pointer is used to indicate which bit is to be modified.

(2) Form

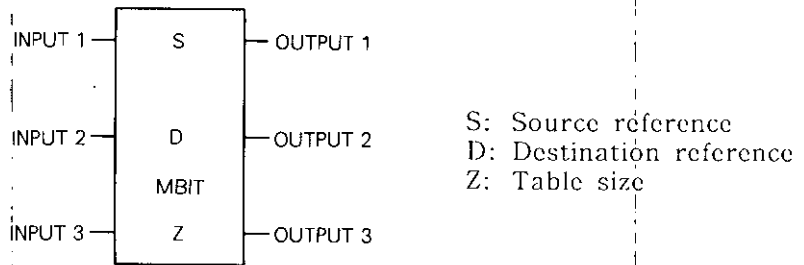


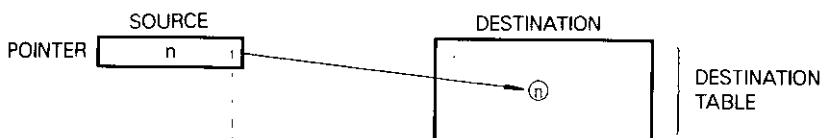
Fig. 5.85 MBIT General Form

- Fig. 5.85 shows the form of modify.
- MBIT is the symbol denoting modify.
- MBIT requires three elements placed vertically (top, middle, and bottom). Referring to table 5.100, specify the needed number for each element.

Table 5.100 MBIT Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Constant K (0001-9600) • Input register (30001-30512) • Holding register (40001-49999) • Link register (R0001-R1024)
Middle	Destination reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Link coil (D0001-D1009) • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	For coil specified in middle: Constant K (1-512) For link coil specified in middle: Constant K (1-64) For others specified: Constant K (1-600)

(3) Operation



NOTE Set bit n to "1" or "0".

5.12.6 Modify (MBIT) (Cont'd)

(a) Inputs

All three inputs are used with the Bit Modify function. Every scan the input 1 receives power flow, a bit is altered. Any of up to 9600 bits can be modified by this function. Transitional contacts can be used if a single operation is desired. The input 2 controls how that bit is to be modified.

When this input receives power flow, the bit will be set to a one (ON) condition; no power flow results in the bit being cleared to a zero (OFF) condition. The current status (ON/OFF) of the bit has no effect on the result after this function. The input 3 when receiving power flow will cause the pointer, if stored in holding register only, to be incremented after the bit is altered. The pointer is not incremented if it is a constant, stored in an input register, or there is no power flow to the input 3. The pointer, if incremented beyond the size of the matrix, will be reset to one automatically.

(b) Outputs

The Bit Modify matrix function utilizes all three outputs. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows Function blocks to be cascaded or chained horizontally within a network.

The output 2 supplies power flow whenever the bit is ON, one, after the operation is performed. Thus this output can be described as sensing the resultant bit or merely copying the state of the input 2 with a successful function. The output 3 will supply power flow if the pointer's magnitude is beyond the size of the matrix (pointer too large). Thus if the pointer in an input register or a holding register without automatic incrementing exceeds the matrix size, no operation is performed and this output is used to indicate the error condition.

MBIT function and status at I/O ON are shown in Tables 5.101 and 5.102, respectively.

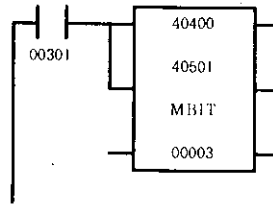
Table 5.101 MBIT Functions

Input ×	Functions	Remarks
Input 1	Sets or clears bits.	These actions are not performed, if the content of pointer is either 0 or exceeds the maximum bit number.
Input 2	Sets "1" at ON; clears "1" (sets to "0") at OFF.	Effective only when the input 1 is ON.
Input 3	Only when holding register is used as source, it adds + 1 to holding register content, after execution.	Effective only when the input 1 is ON.

Table 5.102 MBIT Output Status

Output ×	Status
Output 1	Same as the state of ON/OFF of the input 1 (Copy of input 1)
Output 2	Turns ON, when bit is "1" after execution.
Output 3	Turns ON, when bit No. exceeds the maximum bit No. determined according to table size. (regardless of ON/OFF state of the input 1)

(4) Example



(a) Ladder Circuit



① Before Modify



② After Modify

(b) Content of Modify

Figure above illustrates a typical Bit Modify Matrix function. If the pointer (register 40400) contains the value three, when coil 00301 is energized, the bit in the matrix 40501-40503 at location three will be set to a one. The control that sets the bit (in lieu of clearing the zero) is the vertical connection to the middle input. As long as coil 00301 is energized, bit three will be set every scan; unless other logic clears this bit, no change in the bit will be detectable.

5.12.7 Sense (SENS)

(1) Function

This function allows individual bits in a matrix to be examined, but not altered. An output is used to indicate one (ON) bits with power flow and a zero (OFF) bits without power flow. The operation of this function is similar to the Bit Modify previously discussed, except that no bits are modified. The status of only one bit can be obtained per scan.

(2) Form

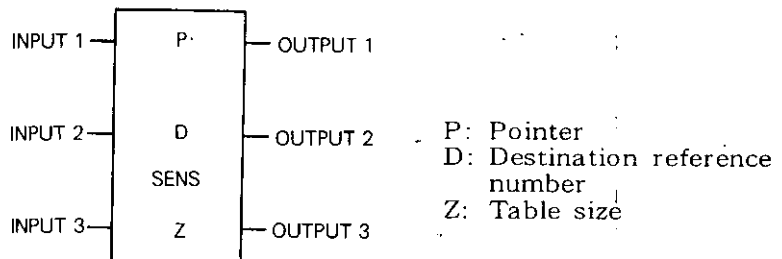


Fig. 5.86 SENS General Form

- Fig. 5.86 shows the form of sense.
- SENS is the symbol denoting sense.
- SENS requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.103, specify the needed number for each element.

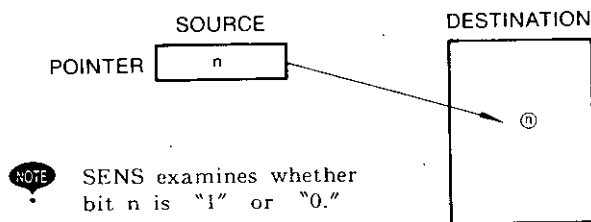
5.12.7 Sense (SENS) (Cont'd)

Table 5.103 SENS Elements

Element	Description	Specified Number
Top	Pointer	<ul style="list-style-type: none"> • Constant K (0001-9600) • Input register (30001-30512) • Holding register (40001-49999) • Link register (R0001-R1024)
Middle	Destination reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Input relay (10001-14081) • Link coil (D0001-D1009) • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024)
Bottom	Table size	<ul style="list-style-type: none"> • For coil specified in middle; Constant (1-512) • For input relay specified in middle; Constant (1-256) • For link coil specified in middle; Constant (1-64) • For register specified in middle; Constant (1-600)

(3) Operation

This function allows individual bits in a matrix to be examined, but not altered. An output is used to indicate one (ON) bits with power flow and a zero (OFF) bits without power flow. The operation of this function is similar to the Bit Modify previously discussed, except that no bits are modified. The status of only one bit can be obtained per scan.



(a) Inputs

All three inputs are used with the Bit Sense function. Every scan the input 1 receives power flow, a bit is located in a matrix and its status is obtained. Any of up to 9600 bits can be obtained with this function. Transitional contacts can be used if a single operation is desired.

The input 2 controls the incrementing of the pointer. When this input receives power flow, the pointer, if stored in a holding register only, will be incremented after the bit is examined. Both the inputs 1 and 2 must receive power flow for the pointer to be incremented. The pointer is not incremented if it is a constant, stored in an input register, or there is no power flow to the input 2. The pointer, if incremented beyond the size of the matrix, will be reset to one automatically. The pointer is also reset to one if the input 1 receives power flow; this resetting is accomplished prior to sensing the bit as required by the top input.

(b) Outputs

The bit Sense matrix function utilizes all three outputs. The output 1 will supply power flow whenever the input 1 receives power flow. Thus the output 1 allows Function Blocks to be cascaded or chained horizontally within a network. The output 2 supplies power flow whenever the bit being sensed (addressed by the pointer) is a one (ON) bit; this output will not supply power flow whenever the bit is a zero (OFF) bit. The input 2 is the resultant of the sense function. The output 3 will supply power flow if the pointer's magnitude is beyond the size of the matrix (pointer too large). Thus if the pointer is a holding register with automatic pointer incrementing and it exceeds the matrix size, no operation is performed and this output is used to indicate the error condition.

SENS function and status at I/O ON are shown in Tables 5.104 and 5.105, respectively.

Table 5.104 SENS Functions

Input ×	Functions	Remarks
Input 1	Executes SENS.	It will not execute SENS, if the pointer content is 0 or if it exceeds the maximum bit No. determined according to table size.
Input 2	Only when the pointer is a holding register, it adds + 1 to the content of the register after execution of ISENS.	The input 1 must be ON.
Input 3	Only when the pointer is a holding register, it turns the register content to 1.	Regardless of the input 1 ON and OFF

Table 5.105 SENS Output Status

Output ×	Status
Output 1	Same as the input 1 ON/OFF status.
Output 2	Turns ON, when the specified bit No. is "1".
Output 3	Turns ON, when the specified bit No. exceeds the maximum bit No. determined according to table size, (Regardless of the input 1 ON and OFF)

(4) Example

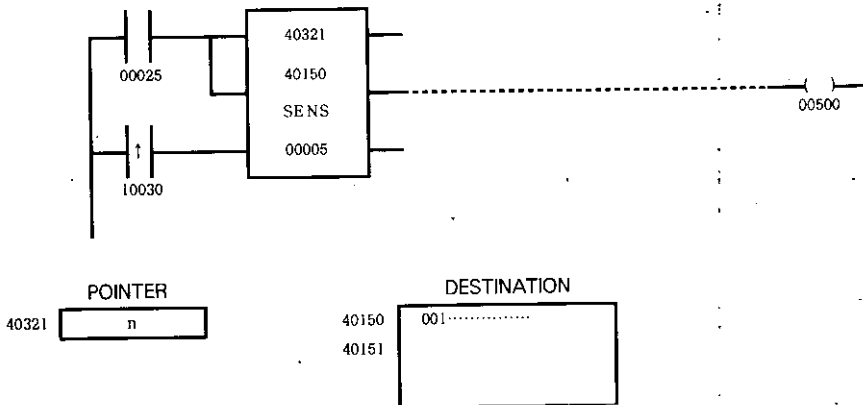


Figure above illustrates a typical Bit Sense Matrix function. If the pointer (register 40321) contains the value zero when input 00025 is energized, coil 00500 will be off during that first scan. Since there is a vertical connection for the power flow to input 2 the pointer will increment by one each scan input 00025 is energized.

Thus on the second scan that this input is ON, the pointer will be at the value one, and the first bit in the matrix (registers 40321-40324) will be sensed. This bit is a zero, and coil 00500 remains OFF; this coil will also be OFF for the third scan while bit two is sensed. When the pointer is incremented to the value three, coil 00500 will be ON sensing bit three as a one bit.

As long as input 00025 is energized, the sense function "walks" through the matrix at the rate of one bit per scan; coil 00500 indicates the status of each bit. The sensing will return to bit one after bit 80 automatically if input 00025 is energized for a sufficiently long period. Whenever input 10030 is energized, the sensing will return to bit one regardless of the pointer's value; since a transitional contact is used, input 1003 must be deenergized and then re-energized to affect the pointer.

5.12.8 Rotate (BROT)

(1) Function

BROT shifts all bits of the source table, one bit to the left or right, and stores the result in the destination table all in a scanning cycle.

(2) Form

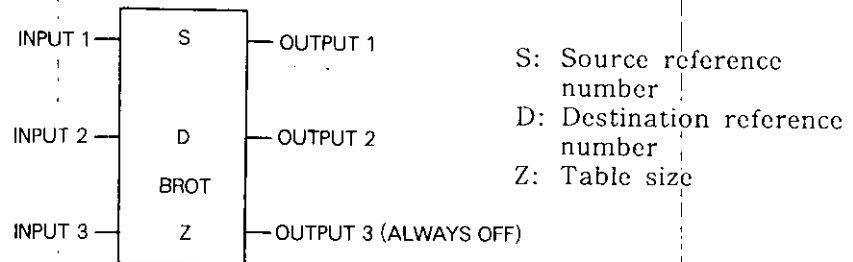


Fig. 5.87 BROT General Form

- Fig. 5.87 shows the form of rotate.
- BROT is the symbol denoting rotate.
- BROT requires three element placed vertically (top, middle, and bottom). Referring to Table 5.106, specify the needed number for each element.

Table 5.106 BROT Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Input relay (10001-14081) • Link coil (D0001-D1009) • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024)
Middle	Destination reference number	<ul style="list-style-type: none"> • Coil (00001-08177) • Link coil (D0001-D1009) • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<ul style="list-style-type: none"> • For link coil specified in top and middle; Constant (1-64) • For others; Constant (1-100)

(3) Operation

(a) Inputs

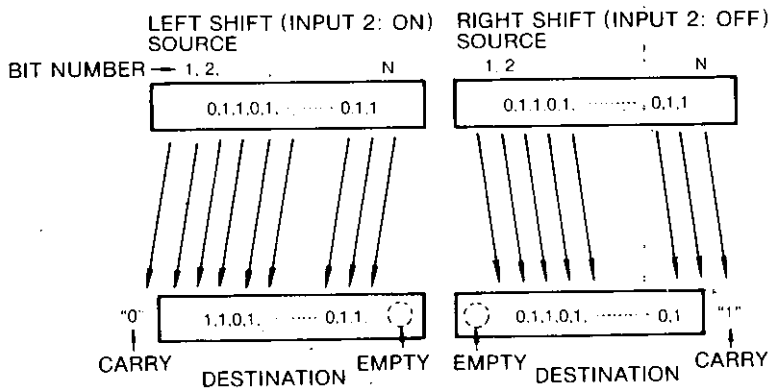
All three inputs are used with the Bit Rotate function. Every scan the input 1 receives power flow, all bits in the matrix will be rotated one position. Up to 1600 bits are moved with this function in one scan.

The input 2 controls the direction of the rotation. If this input receives power flow, the matrix will be rotated towards the left (bit 17 into 16, bit 16 into 15, ..., bit 3 into 2, bit 2 into 1, and bit 1 rotated out of the matrix). If the input 2 does not receive power flow, the matrix will be rotated towards the right (bit 1 into 2, bit 2 into 3, ..., bit 15 into 16, bit 16 into 17, etc.); the last bit will be rotated out of the matrix. The input 3 controls what happens to the single bit location vacated by the rotate to create either a shift operation or a true rotate. When this input does not receive power flow, the bit rotated out is ignored and vacant location at the opposite end of the matrix will be filled with zero; this is a shift operation. If this input receives power flow, the bit rotated out is carried around unchanged and entered into the opposite end of the matrix; this is a true rotate operation.

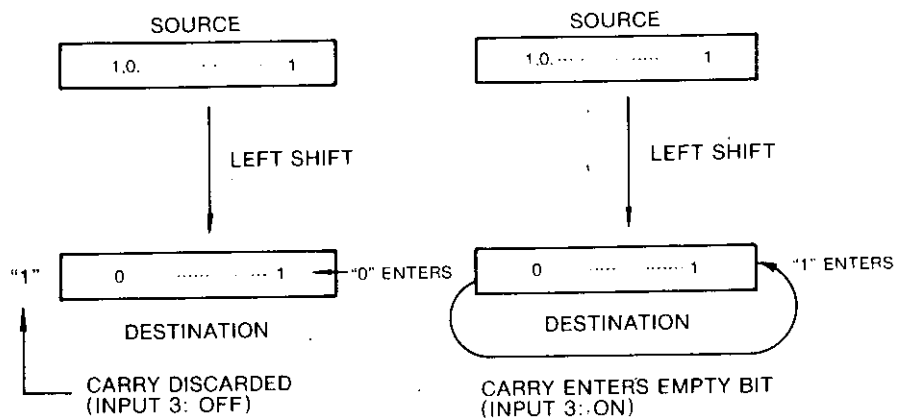
(b) Outputs

This function utilizes only the first two outputs. The output 3 has no significance and will be OFF (no power flow) under all conditions. The output 1 will supply power flow whenever the input 1 receives power flow. The output 2 supplies power flow when the carry is "1."

The direction of shift is determined by ON/OFF status of the input 2.



It is possible to select whether to fill the empty bit with 0 or with the carry, by ON/OFF status of input 3.



The source data remains unchanged unless the source and destination tables are the same. When the source and destination tables are the same, BROT realizes a 1-bit, N-stage (N is the total number of bits) shift register.

BROT function and status at I/O ON are shown in Tables 5.107 and 5.108, respectively.

Table 5.107 BROT Functions

Input ×	Functions
Input 1	Executes BROT.
Input 2	<ul style="list-style-type: none"> • Shifts to the left (LSB→MSB) when it is ON. • Shifts to the right (MSB→LSB) when it is OFF.
Input 3	<ul style="list-style-type: none"> • When it is ON, a carried bit enters an empty bit. • When it is OFF, "0" enters an empty bit.

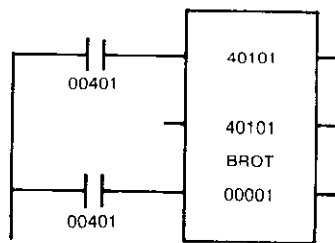
Table 5.108 BROT Output Status

Output ×	Status
Output 1	Same as the input 1 ON/OFF status.
Output 2	It turns ON, when "1" is forced out as a result of shifting (carrier).
Output 3	Always OFF.

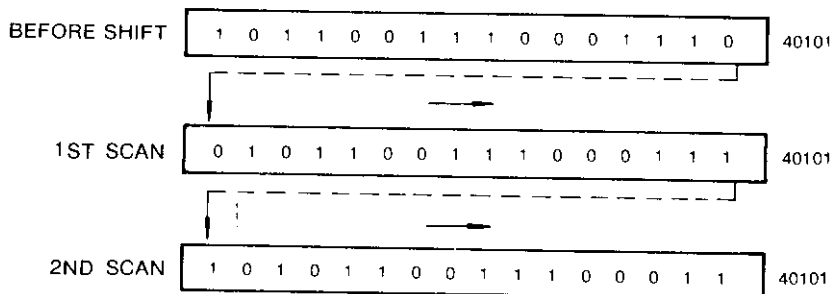
5

(4) Example

This is an example of a 1-bit, 16-stage ring counter. (The source and destination tables are the same.) This shifts to the right as the input 2 is OFF.



Ladder



5.12.9 Multi-Rotate (MROT)

(1) Function

MROT shifts all bits of the destination table, by the number of bits (1-15) stored in the source register to the left or right, and stores the result in the destination table all in a scanning cycle.

(2) Form

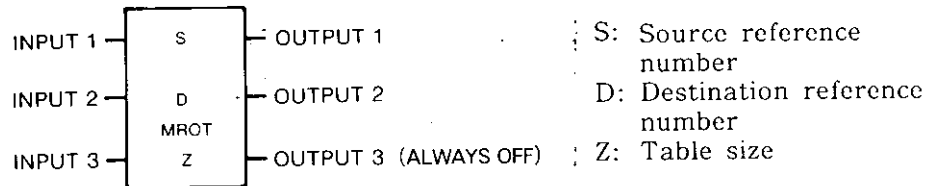


Fig. 5.88 MROT General Form

- Fig. 5.88 shows the form of multi-rotate.
- MROT is the symbol denoting multi-rotate.
- MROT requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.109, specify the needed number for each element.

Table 5.109 MROT Elements

Element	Description	Specified Number
Top	Source reference number	• Holding register (40001-4999) • Link register (R0001-R1024)
Middle	Destination reference number	• Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	• Constant (1-100)

(3) Operation

(a) Inputs

When the input 1 receives power flow, MROT executes a shift. The input 2 determines the direction of shift as follows.

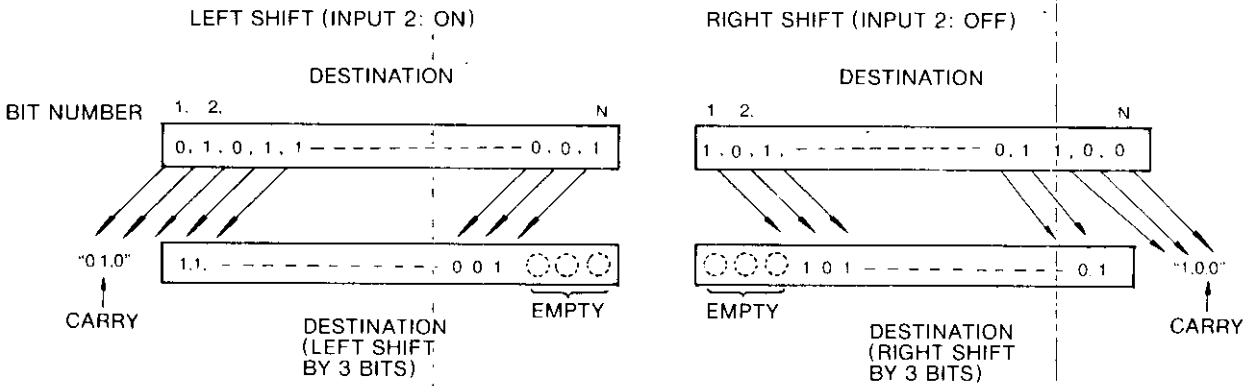
- When the inputs 1 and 2 are ON: Shift to left (to bit 1)
- When the input 1 is ON and 2 OFF: Shift to right (to bit N)
The input 3 determines what should fill the empty bits as follows.
- When the inputs 1 and 3 are ON, the carry bits fill the empty bits.
- When input 1 is ON and 3 OFF, 0's fill the empty bit.

(b) Outputs

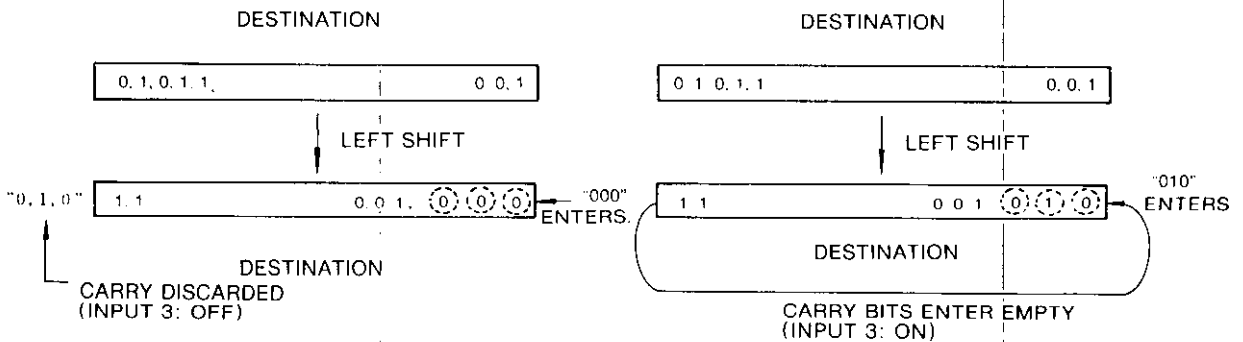
The output 1 is ON when the input 1 receives power flow and MROT executes a shift. The output 2 is ON when the input 1 receives power flow and MROT cannot execute a shift (the number of shift bits is 16 or more, or the source register is included in the destination table). The output 3 is always OFF.

NOTE When number of shift specified in source register is 0, the shift is not operated, but the output 1 is ON.

The input 2 determines the direction of shift.



It is possible to select whether to fill the empty bits with 0's or with the carry bits, by ON/OFF status of input 3.



Unlike BROT, the destination data are changed by the shift successively. MROT realizes an n-bit (n is the number of bits to shift), N-stage (N is the total number of bits) shift register.

5.12.9 Multi-Rotate (MROT) (Cont'd)

MROT function and status at I/O ON are shown in Tables 5.110 and 5.111, respectively.

Table 5.110 MROT Functions

Input ×	Functions
Input 1	Executes MROT.
Input 2	<ul style="list-style-type: none"> Shifts to the left (LSB→MSB) when it is ON. Shifts to the right (MSB→LSB) when it is OFF.
Input 3	<ul style="list-style-type: none"> When it is ON, a carried bit enters an empty bit. When it is OFF, "0" enters an empty bit.

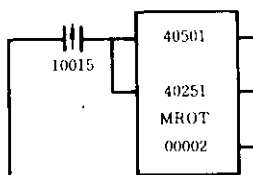
Table 5.111 MROT Output Status

Output ×	Status
Output 1	It turns ON when the input 1 is ON and MROT is executed.
Output 2	It turns ON when the input 1 is ON and MROT cannot be executed.
Output 3	Always OFF

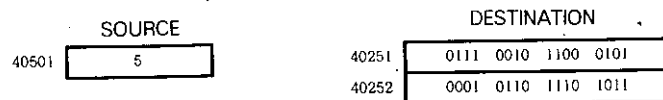
(4) Example

This is an example of 5-bit left shift. 0's fill the empty bits as the input 3 is OFF.

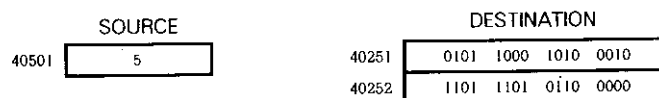
Example 1:



(a) Ladder



① Before Shift

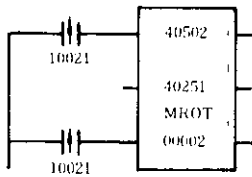


② After Shift

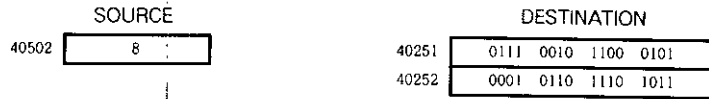
(b) Shift Operation

This is an example of 8-bit right shift. The carry bits fill the empty bits as the input 3 is ON.

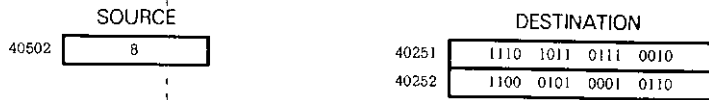
Example 2:



(a) Ladder



(1) Before Shift



(2) After Shift

(b) Shift Operation

5.12.10 Byte Rearrangement(TWST)

(1) Function

This function divides the registers in the destination table into the higher-place byte (8 bits) and the lower-place byte (8 bits). And bits in each byte are rearranged. All bits in the destination table are rearranged in one scan and stored in the destination table.

(2) Form

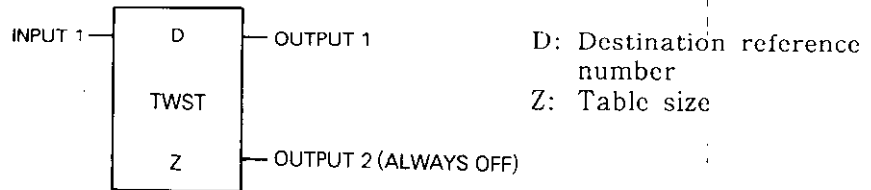


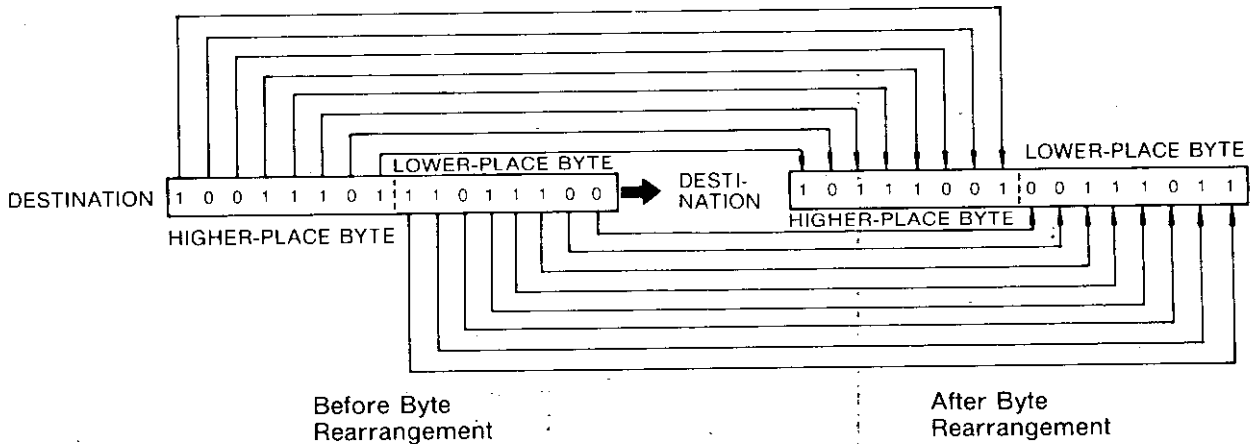
Fig. 5.89 TWST General Form

- Fig. 5.89 shows the form of byte rearrangement.
- TWST is the symbol denoting byte rearrangement.
- TWST requires two elements placed vertically (top and bottom). Referring to Table 5.112, specify the needed number for each element.

Table 5.112 TWST Elements

Element	Description	Specified Number
Top	Destination reference number	• Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	• Constant (1-100)

(3) Operation



When the statuses of coils and input relays are read with the GL60H or GL70H used as the MEMOBUS master, the information will come from a slave normally in such an order that a higher-place bit contains the status of a coil or input relay having a greater number. TWST can be used to rearrange the bits to the ordinary order.

SOURCE	
8-1	16-9
24-17	32-25
40-33	48-41

➔

SOURCE	
1-8	9-16
17-24	25-32
33-48	41-48

Before Byte Rearrangement After Byte Rearrangement

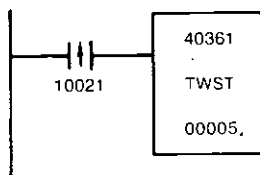
(a) Inputs

Only the input 1 is used for this function. When the input 1 receives power flow, the byte rearrangement operation can be performed.

(b) Outputs

TWST utilizes only the output 1 which will supply power flow whenever the input 1 receives power flow.

(4) Example



(a) Ladder

DESTINATION			
40361	0110	1010	1100 0001
40362	1110	0101	1101 1110
40363	1110	0101	1100 1100
40364	1110	1110	1011 0101
40365	0101	0011	1011 1111

➔

DESTINATION			
40361	0101	0110	1000 0011
40362	1010	0111	0111 1011
40363	1010	0000	0011 0011
40364	0111	0111	1010 1101
40365	1100	1010	1111 1101

Before Byte Rearrangement After Byte Rearrangement

(b) Byte Rearrangement Operation

The higher-place bits can be replaced with the lower-place bits by using TWST and SWAP.

5.12.11 Bit Count (BCNT)

(1) Function

It counts the number of bits of "1" or "0" in the source table and stores the result in the destination.

(2) Form

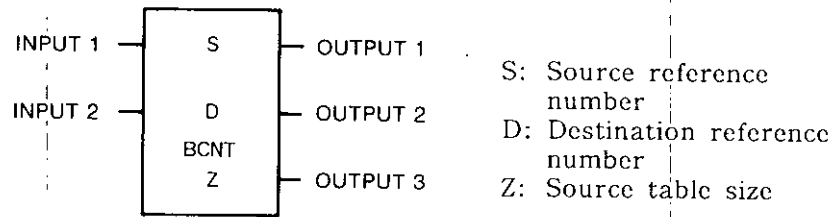


Fig. 5.90 BCNT General Form

- Fig. 5.90 shows the form of bit count.
- BCNT is the symbol denoting bit count.
- BCNT requires three elements placed vertically (top, middle, and bottom). Referring to Table 5.113, specify the needed number for each element.

Table 5.113 BCNT Elements

Element	Description	Specified Number
Top	Source reference number	<ul style="list-style-type: none"> • Input register (30001-30512) • Holding register (40001-49999) • Constant register (31001-35096) • Link register (R0001-R1024)
Middle	Destination reference number	<ul style="list-style-type: none"> • Holding register (40001-49999) • Link register (R0001-R1024)
Bottom	Table size	<ul style="list-style-type: none"> • Constant (1-100)

(3) Operation

When the input 1 turns ON, it counts the number of bits of "1" or "0" in the source table according to the ON/OFF status of the input 2 and stores the number in the destination table. It counts the numbers in all source tables in 1 scan.

BCNT function and status at I/O ON is shown in Tables 5.114 and 5.115, respectively.

Table 5.114 BCNT Functions

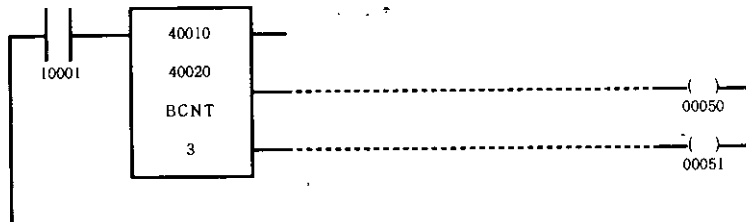
Input ×	Functions
Input 1	Counts the number of bits of "1" or "0."
Input 2	Counts the number of bits of "1" at ON, and of "0" at OFF.

5.12.11 Bit Count (BCNT) (Cont'd)

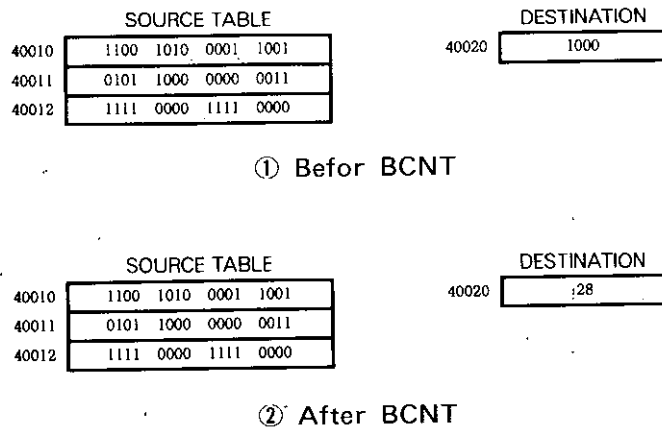
Table 5.115 BCNT Output Status

Output ×	Status
Output 1	Same as the input 1 ON/OFF status.
Output 2	It turns ON when the number of bits of "1" or "0" is an odd number.
Output 3	It turns ON when the number of bits of "1" or "0" is an even number.

(4) Example



(a) Ladder



(b) Bit Count Operation

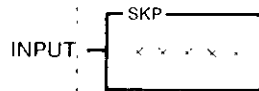
BCNT shown in (a) counts the number of "0" bits of the source table, because the input 2 turns OFF when input relay 10001 turns ON. Since the operation result is 28, which is an even number, the output 3, that is, output coil 00051 turns ON. This function can be used for parity check of data.

5.13 SKIP

This function allows logic in groups of networks to be skipped and thus not solved. Networks that are skipped will have their coils (if any) unchanged and register content unaltered; skipped networks are thus basically "frozen."

The Skip Function is useful to reduce the logic scan time. A skip of zero networks saves all of the time required to solve the logic remaining. A skip of non-zero networks reduces the time to solve the skipped logic to that of relay functions. The reduction in scan time cannot exceed that of the logic remaining. Since all logic must be examined to count and determine quantity of networks (whose size does vary greatly), non-zero skips will save less time than a zero skip. In either case, the controller's over all scan time cannot be reduced beyond that required to service I/O devices. In addition, the Skip Function can be used to select various groups of logic to be performed from a larger selection of logic.

(1) Form



Only one function block is used with the skip function, and input is available for control

- Reference number

The function block specifies the quantity of networks to be skipped. The value can be fixed quantity up to 9999 or a register reference ($3 \times \times \times \times$ or $4 \times \times \times \times$).

(2) Function and Operation

Assume the Skip instruction is stored in the network N and the number of networks to be skipped is J.

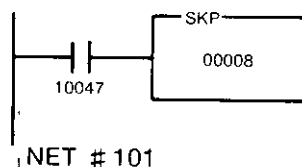
In a scanning cycle when the input is ON, processing of the J successive networks including network N (N, N+1, N+2, ..., N+J-1) is stopped.

If $J = 0$ or J is greater than the number of networks following network N, processing of all networks following network N will be skipped.

In network N, all logics that follow the element of Skip according to the order of network solving (see Par. 4.5.1) will be skipped.

- NOTE**
1. In case segment allocation is made in 2-level scan, no Skip is allowed exceeding the segments.
 2. Use of this function in action circuit or subroutine circuit is not allowed.

(3) Circuit Example



When the input relay 10047 is ON, networks 101-108 will be skipped. If the network 105 is the last one, networks 101-105 will be skipped.

5.14 SUBROUTINE

(1) Function

When the same circuit is to be formed in a ladder circuit many times (if it is stored as a subroutine circuit), it can be freely called up from the ladder circuit, and only one circuit has to be formed.

(2) Form

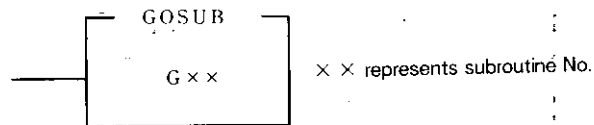
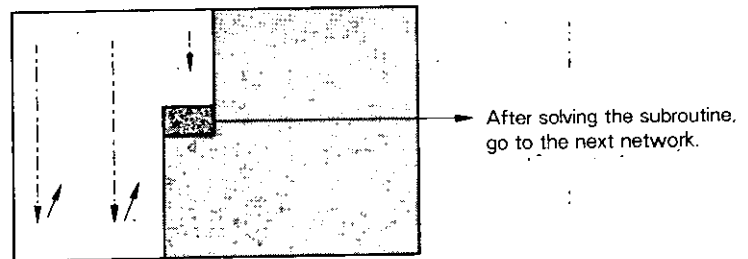


Fig. 5.91 GOSUB General Form

- Fig. 5.91 shows the form of subroutine.
- GOSUB is the symbol denoting subroutine.
- GOSUB requires subroutine No. as a component element. Specify a constant in the range from 0 to 99 according to the service condition of the subroutine circuit. No output is made.

(3) Operation



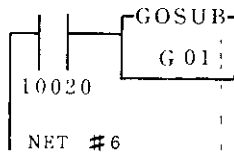
In the above diagram, suppose that GOSUB is at the position of . At this time, when solving is performed in sequence, the input of GOSUB is ON and GOSUB is solved, solving jumps to the specified subroutine circuit.

When solving of the subroutine circuit is finished, Solving will start from the head of the network next to the network in which GOSUB had been included. Consequently, since the part shown with in the above diagram will not be solved, any attempt to input coil or relay from P150 Programming Panel will fail. For details see "P150 Programming Panel Manual".

Furthermore, the same subroutine circuit can be called up any number of times by GOSUB, but in a subroutine circuit, it is not possible to further call up a subroutine by GOSUB (the so-called nesting). A subroutine circuit is stored in the unit of network, and it is formed within the scope of memory allocation of subroutine.

Use of this function in either an operation circuit or a transition condition circuit is not allowed.

(4) Example



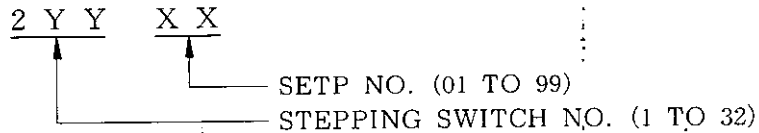
When input relay 10020 is ON, the subroutine circuit with subroutine No. 01 will be solved. Upon completing solving, this function restarts solving of the ladder circuit from NET #7.

NOTE For the state of coil used in the subroutine circuit and the content of register, in the case of the scan where the subroutine is not called up by GOSUB, the state in the scan called up last is held. Consequently, when the coil which is in the ON state is required to be turned OFF, the subroutine must be called up once again, after making an arrangement to allow the coil to turn OFF.

5.15 STEPPING SWITCH

5.15.1 Stepping Switch Functions and Operation

- (1) The GL60H or GL70H is provided with 32 stepping switches. Each switch can take up to 99 steps independently. The stepping switch reference Nos. and their meanings are as shown below:



- (2) Each stepping switch from 1 to 32 is controlled according to the content of holding registers 42001 to 42032 (numerical values). By controlling these register contents (called stepping switch control registers) by using functions such as timer, counter, addition or subtraction, the stepping switches can be permitted to step in proper order, return or jump.
- (3) Fig. 5.93 shows the equivalent circuit of the stepping switch 1. As shown in the figure, stepping switch 1 operates according to the holding register 42001 content n , as shown below:

- ① When n is in the range of 1 to 99;

Corresponding to the value of n , one of the simulated coils 20101 to 20199 becomes ON and its NO contact is conductive and NC contact non-conductive. For example, when $n = 1$, only simulated coil 20101 is turned on and its NO contact is conductive and NC non-conductive.

- ② When n is not in the range of 1 to 99;

All simulated coils 20101 to 20199 are turned off.

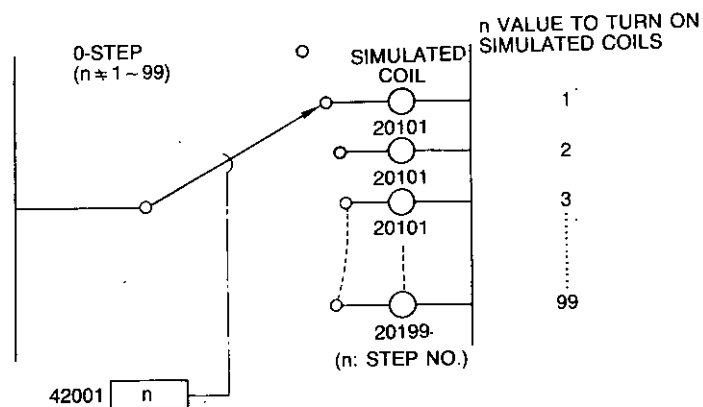


Fig. 5.93 Equivalent Circuit of Stepping Switch 1

(4) Tabel 5.116 shows the operation of stepping switches 1 to 32.

Table 5.116 Stepping Switch Operation

Stepping Switch No.	Stepping Switch Control Register No.	Stepping Switch Control Register Data (Numerical)				
		001	002	003	009
1	42001	Only 20101 "ON"	Only 20102 "ON"	Only 20103 "ON"	Only 20199 "ON"
2	42002	Only 20201 "ON"	Only 20202 "ON"	Only 20203 "ON"	Only 20299 "ON"
3	42003	Only 20301 "ON"	Only 20302 "ON"	Only 20303 "ON"	Only 20399 "ON"
4	42004	Only 20401 "ON"	Only 20402 "ON"	Only 20403 "ON"	Only 20499 "ON"
5	42005	Only 20501 "ON"	Only 20502 "ON"	Only 20503 "ON"	Only 20599 "ON"
...
31	42031	Only 23101 "ON"	Only 23102 "ON"	Only 23103 "ON"	Only 23199 "ON"
32	42032	Only 23201 "ON"	Only 23202 "ON"	Only 23203 "ON"	Only 23299 "ON"

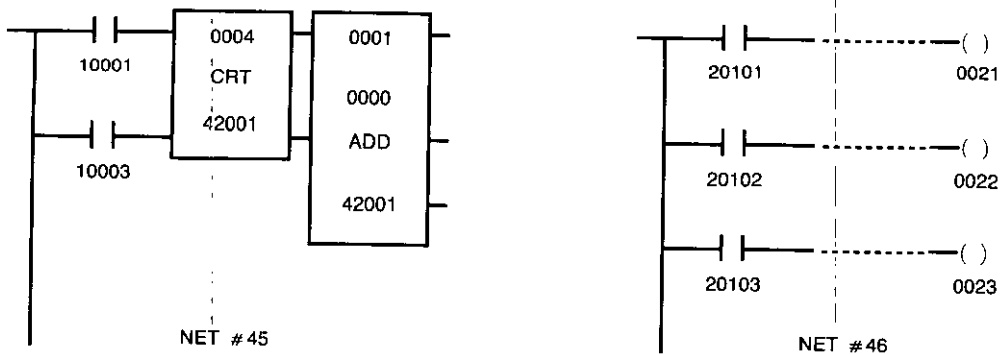
NOTE

1. When stepping switches are not used, switches 42001 to 42032 can be used as holding registers.
2. When a stepping switch is used in the network, its NO and NC contacts are used but the differential contact cannot be used.

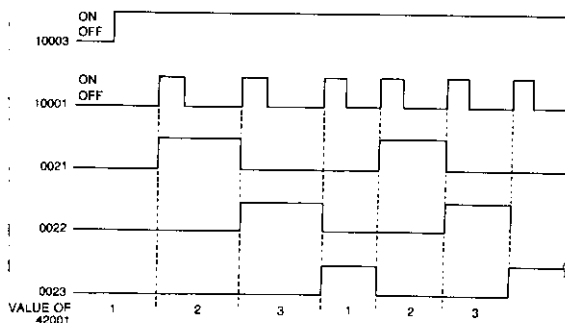
5.15.2 Example of Stepping Switches

Example 1: The following shows an example where stepping switch 1 is let step by counter. Coils 21 to 23 operate by the ladder in (a) as shown in (b).

(a) Ladder

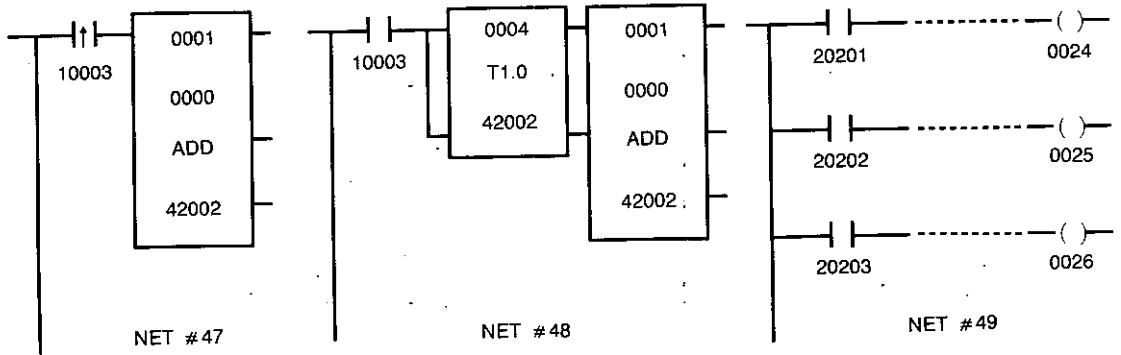


(b) Operation

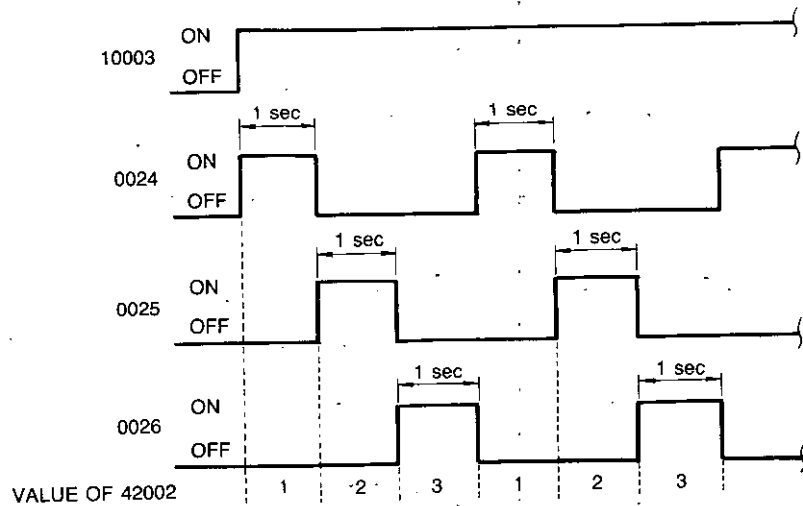


Example 2: The following shows an example where stepping switch 2 is permitted to step by timer. Coils 24 to 26 operate by the ladder in (a) as shown in (b).

(a) Ladder



(b) Operation



NOTE Refer to Par. 5.3.4 (7) "Timer Error".

5.16 ASCII COMMANDS

The ASCII commands are used to input/output ASCII devices (printer, CRT terminal, keyboard, bar-code reader, etc.) containing communication ports with EIA RS-232C specifications and ASCII characters. There are two types of ASCII commands: read (READ) command and write (WRIT) command. To operate the ASCII commands, remote I/O driver module (module type: JAMSC-IF62) and ASCII module (module type: JAMSC-IF71) are necessary.

For details of the ASCII devices, refer to the ASCII module user's manual (SIE-C815-14.4).

5.16.1 Read Command (READ)

(1) Function

The READ command is used to store the ASCII characters read from the ASCII devices in the holding registers according to the format specified by the message.

(2) Form

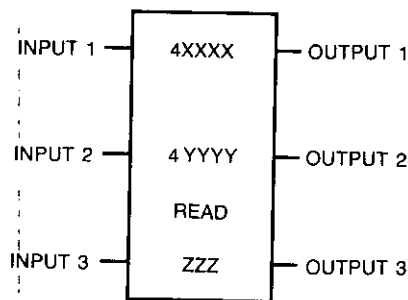


Fig. 5.94 READ Form

Table 4.117 READ Elements

Element	Description	Specified Number
Top	ASCII read control register	Holding register (40001 to 49993)
Middle	Destination reference number	Holding register (40001 to 49999)
Bottom	Table size	Constant (1 to 999)

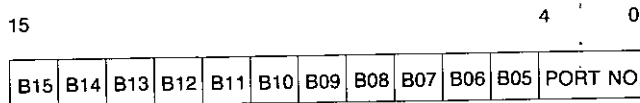
(a) Outline of top element

4 XXXX	PORT NO.
4 XXXX +1	MESSAGE NO.
4 XXXX +2	REQUIRED NO. OF REGISTERS
4 XXXX +3	INPUT NO. OF REGISTERS
4 XXXX +4	READ COMMAND EXECUTION STATUS
4 XXXX +5	ALWAYS SET TO "0"
4 XXXX +6	TOTAL SUM OF 4 XXXX-4 XXXX+5

NOTE 4XXXX+2 to 4XXXX+6 are used at the system side.

(b) Outline of 4XXXX

Upper-digit 11 bits of 4XXXX in which the port No. is stored are allocated in information bits of errors that have occurred at the READ command execution.



- B15: BCC check error in MEMOBUS mode
- B14: Input data differ from the specified format when non-keyboard type is used.
- B13: A transmission error occurred between ASCII devices and ASCII modules.
- B12: Buffer overrun error in ASCII modules
- B11: Ineffective format was in the execution message.
- B10: CR (carriage return) key was depressed and the READ command execution was aborted before all data items specified by the message have been input when the keyboard type was used.
- B09: The specified message has not been programmed in the ASCII module.
- B08: The ASCII modules with the specified port No. are not provided at the remote side.
- B07: 1731 (decimal) has been set in the port No. area of 4XXXX.
- B06: The remote I/O driver module having the relevant ASCII module has not been mounted in the GL60H or GL70H.
- B05: During ASCII message programming (ASCII module STOP status)

NOTE If any of the above errors except B14 and B10 is detected, the READ command is aborted for completion failure. In case of B10, the operation is normally completed.

(3) Operation

① Inputs

Input 1: Execution command. Effective when inputs 2 and 3 are turned off and input 1 is turned on. The READ command is executed immediately if the same port is not used by the other ASCII command.

Input 2: Pause command. The READ command under execution is paused. It is paused when input 3 is turned off and input 2 is turned on during READ execution command execution. The READ command restarts execution when input 2 is turned off again. Output 1 is turned off during the pause command execution.

Input 3: Abort command. The READ command under execution is aborted. The operation is completed as an error after the abort command execution.

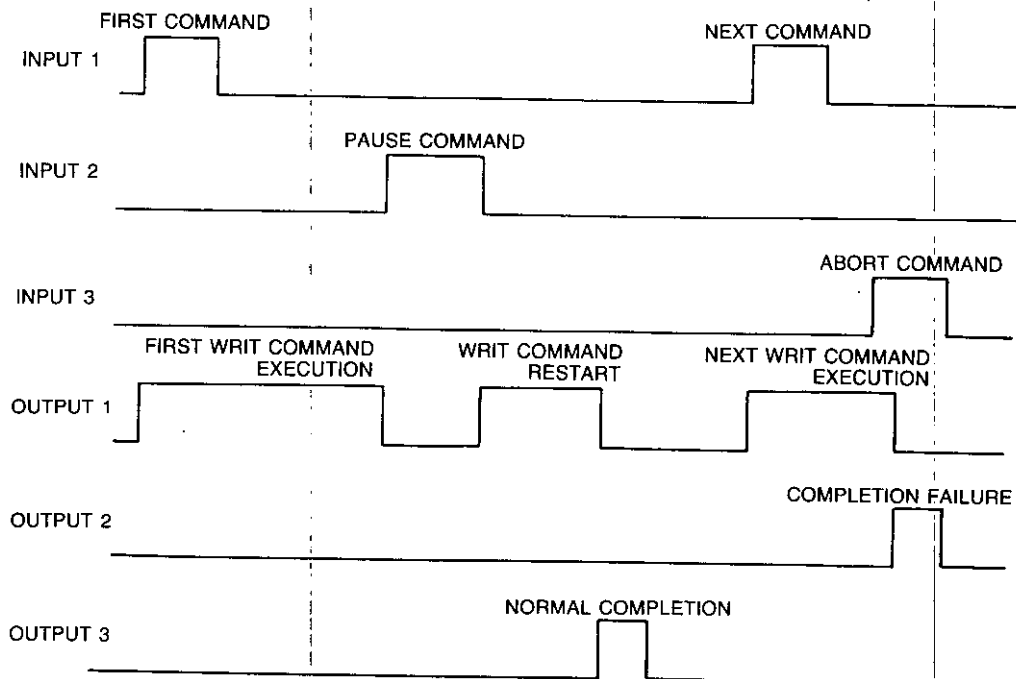
② Outputs

Output 1: Execution output. Indicates that the READ command is under execution when this output is turned on.

Output 2: Error output. If any error occurs during READ command execution, only one scan is turned on to inform that the operation has failed to be completed.

Output 3: Execution completion output. Only 1 scan is turned on after the completion of READ command execution to inform that the operation is completed normally.

③ Operation timing



5.16.2 Write Command (WRIT)

The write command (WRIT) is used to output data stored in input registers, constant registers or holding registers to the ASCII devices (CRT, printer, etc.) according to the format specified by the message.

(1) Form

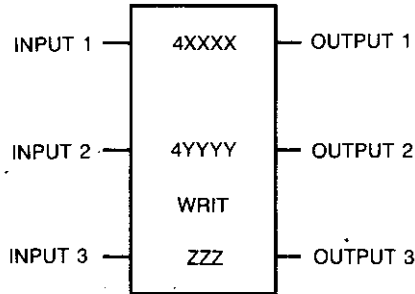


Fig. 5.95 WRIT Form

(2) Elements

Table 5.118 WRIT Elements

Element	Description	Specified Number
Top	Source reference number	Input register (10001 to 30512) Constant register (31001 to 35096) Holding register (40001 to 49999)
Middle	ASCII write control register	Holding register (40001 to 49999)
Bottom	Table size	Constant (1 to 999)

(3) Operation

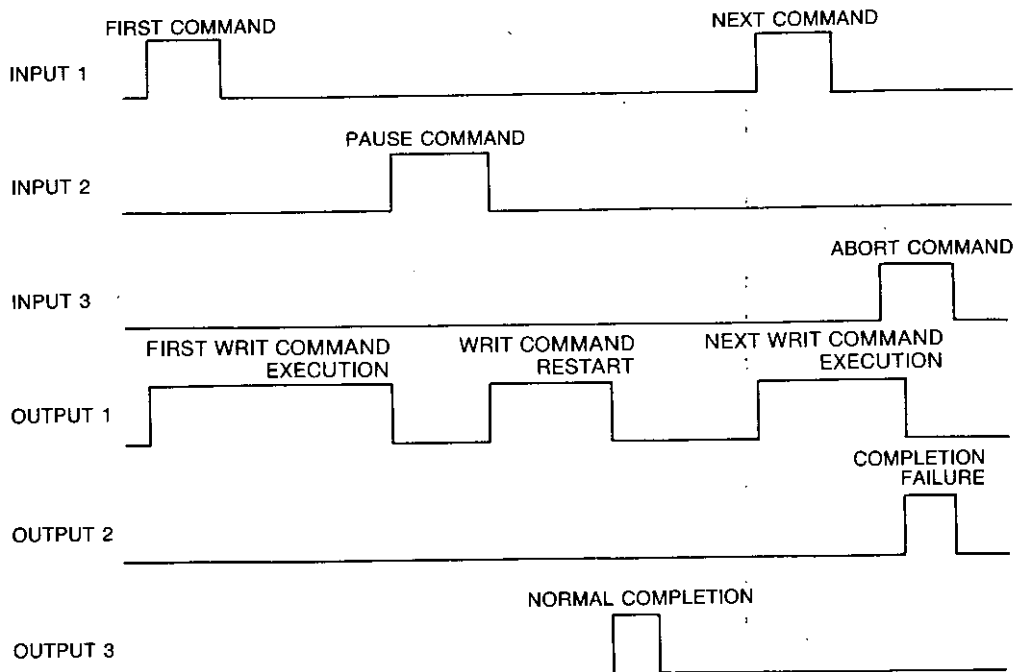
① Inputs

- Input 1: Execution command. Effective when inputs 2 and 3 are turned off and input 1 is turned on. The WRIT command is executed immediately if the same port is not used by the other ASCII command.
- Input 2: Pause command. The WRIT command under execution is paused. It is paused when input 3 is turned off and input 2 is turned on during WRIT execution command execution. The WRIT command restarts execution when input 2 is turned off again. Output 1 is turned off during the pause command execution.
- Input 3: Abort command. The READ command under execution is aborted. The operation is completed as an error after the abort command execution.

② Outputs

- Output 1: During execution. Indicates that the WRIT command is under execution when this output is turned on.
- Output 2: Error output. If any error occurs during WRIT command execution, only one scan is turned on to inform that the operation has failed to be completed.
- Output 3: Execution completion output. Only 1 scan is turned on after the completion of WRIT command execution to inform that the operation is completed normally.

③ Operation timing



5.17 COMM COMMAND

COMM command is used to activate communication from the GL60H or GL70H. By COMM command, it is possible to communicate via IOP of COMM RS-232C communication port as a MEMOBUS master. Therefore, the MEMOBUS master functions can be used without ASCII I/F.

By using each of four communication ports of IOP or COMM at a proper time, equal communication is possible between three GL series CPUs.

For the details of the COMM command, refer to the COMM command instruction (SIE-C815-14.5).

(1) Function

This command is used to output transmitting data specified by the holding register in the middle element from the port number specified by the holding register in the top element and receive the receiving data.

(2) Form

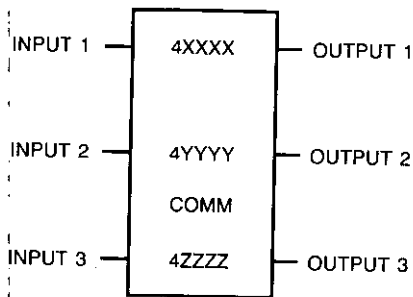
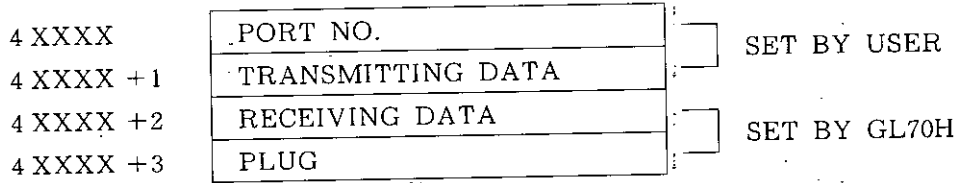


Fig. 5.96 COMM Form

Table 5.119 COMM Elements

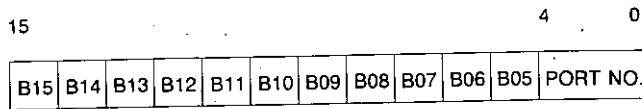
Element	Description	Specified Number
Top	COMM control register	Holding register (40001 to 49996)
Middle	Transmission buffer	Holding register (40001 to 49999)
Bottom	Receiving buffer	Holding register (40001 to 49999)

(a) Outline of top element



(b) Outline of 4XXXX

Upper-digit 11 bits of 4XXXX in which the port No. is stored are allocated in information bits of errors that have occurred at the READ command execution.



- B15: BCC check error in MEMOBUS mode
- B14: Transmission disabled from specified port
- B13: For future use
- B12: Receiving buffer overrun
- B11: Improper transmitting data
- B10: Matching device is not connected to specified port.
- B09: Improper port specified
- B08: Specified port IOP or COMM fault
- B07: Address or function code of received response does not match transmission message.
- B06: For future use
- B05: PP is connected to specified port.

NOTE Status of B04 to B15 (except B05) is updated at every execution of the command.

(3) Operation

① Input

Input 1: Execution command

When this input is turned on, COMM command execution is instructed. Input 3 must be turned off. When this input is accepted and the execution starts, output 1 is turned on.

NOTE Normally, use a differential contact for this input. When this input is turned on, the COMM command execution restarts.

Input 2: Mode designation

Either transmission/receiving mode or transmission mode is specified by this input.

ON : transmission mode

OFF: transmission/receiving mode

In the transmission mode, only transmission is performed and response is waited for from the matching device. When no response is provided from the matching device, waiting status is entered and the command is not completed. Therefore, time-out process is needed in the applications.

This input is checked only at COMM command activation.

Input 3: Abort command

The COMM command under execution is aborted. When this input is turned on, three outputs are turned off. This input has priority over any other inputs.

Used to abort the time-out command, etc. in the receiving mode.

NOTE Normally, use a differential contact for this input. The COMM command is not executed even if this input is turned on.

* : Command execution conditions

- Relevant port is not operating.
- Input 1 is turned off and then on.
- Input 3 is turned on.

② Output

Output 1: During execution

When input 1 is accepted and the COMM command execution starts, this output is turned on. It is turned off when operation is completed or aborted.

It is also turned off if an error occurs during COMM command execution.

Output 2: Error

Only 1 scan is turned on when the COMM command execution is completed with an error.

Output 3: Operation completion

Only 1 scan is turned on when the COMM command execution is completed normally. All status error bits are 0.

5.18 COMMUNICATION COMMANDS

The GL60H or GL70H is provided with eight types of communication exclusive-use commands for mutual communication between the GL series via the 3200IF module. By using the communication commands, communication between the GL series becomes very easy to perform.

These communication commands can execute five types of operations at the same time. (READ, WRIT and COMM can be executed independently from these commands.) When more than six types are activated, only the first activated five are executed and the others are in the waiting status to be executed in order.

- (I) MBUS
Supports MEMOBUS functions.
1:1 communication
- (II) PEER
Selective simultaneous broadcasting
The same data are transmitted simultaneously to more than one GL60S2, GL60S3, GL60H or GL70H.
- (III) BROD
All station simultaneous broadcasting
- (IV) BOOK
BOOK (booking) command
- (V) POLL
POLL command
Cannot be used without BOOK command.
- (VI) DIAG
Node diagnosis command
- (VII) SND
Data transmission command
- (VIII) RCV
Data receiving command

For details of the communication commands, refer to the YENET user's manual (SIE-C815-14.16).

5.18.1 MBUS

(1) Function

Communication is performed between one master and one slave station. Coils, input relays, holding registers or input registers are read or written. This command cannot perform broadcasting (simultaneous broadcasting).

(2) Form

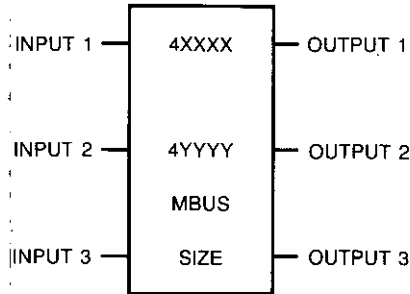


Fig. 5.97 MBUS Form

Table 5.120 MBUS Elements

Element	Description	Specified Number
Top	MBUS control register	Holding register (40001 to 49991)
Middle	Trasmission/receiving buffer	Holding register (40001 to 49999)
Bottom	Size	Constant (1 to 125)

(a) Outline of top element

Upper element control block.

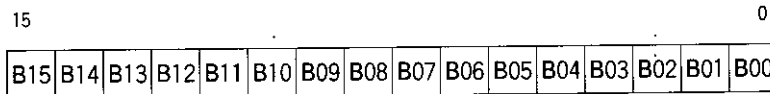
Based on the data information specified by this control block, data are read-out or written-in from or to the specified node.

4 XXXX	NETWORK ADDRESS
4 XXXX +1	DESTINATION ADDRESS
4 XXXX +2	FUNCTION CODE
4 XXXX +3	REFERENCE NO.
4 XXXX +4	DATA COUNT
4 XXXX +5	RETRY COUNT
4 XXXX +6	TIMER
4 XXXX +7	STATUS
4 XXXX +8	SYSTEM USE

NOTE 9 holding registers from 4XXXX are used exclusively.

(b) Outline of 4 XXXX + 7

Status after execution is stored. It is updated before MBUS execution start. 4 XXXX+7 is used.



B15 to B13: System use
B12: Error response receiving
B11: Receiving time exceeded
B10: Transmission time exceeded
B07: Address error
B06: Function error
B05: Reference error
B04: Parameter error
B03: Buffer size error
B02 to B00: Number of retry operations

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($-|\uparrow|-$). Once execution status is entered, execution continues until it is completed (however, the operation is not completed normally if a fault such as transmission error, time-out, etc. occurs.)

This input is not accepted during execution (before completion) even if the command is turned on or off.

Input 2: For future use

Input 3: Abort command

The command execution is aborted. This input signal is a level signal with priority over input 1.

② Outputs

Output 1: During execution

The command is under execution.

Output 2: Error

Indicates that the command execution is completed with an error. Only 1 scan is turned on and the error contents are reflected in the status.

Output 3: Completion

Only 1 scan is turned on at normal completion.

NOTE When the execution start command is input by NO contact ($-|\uparrow|-$), the contact is turned on at communication completion.

5.18.2 PEER

(1) Function

The same data (registers, files) are transferred to up to 8 matching stations in the same area in each station. Transmitted data are then stored in the holding registers as a transmission buffer.

There is no response since communication is performed by using simultaneous broadcasting (no response). This command is effective only for GL60S2, GL60S3, GL60H and GL70H.

(2) Form

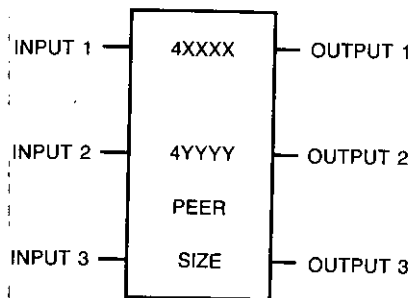


Fig. 5.98 PEER Form

Table 5.121 PEER Elements

Element	Description	Specified Number
Top	PEER control register	Holding register (40001 to 49987)
Middle	Transmission data	Holding register (40001 to 49994)
Bottom	Size	Constant (1 to 125)

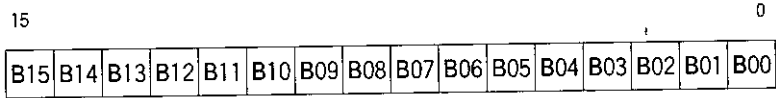
(a) Outline of top element

4 XXXX	NETWORK ADDRESS
4 XXXX + 1	DESTINATION ADDRESS #1
	...
4 XXXX + 8	DESTINATION ADDRESS #8
4 XXXX + 9	4ZZZZ
4 XXXX + 10	DATA COUNT
4 XXXX + 11	STATUS
4 XXXX + 12	SYSTEM USE

NOTE 13 holding registers from 4XXXX are used exclusively.

(b) Outline of 4XXXX +11

Status after execution is stored. It is updated before PEER execution start. 4XXXX +11 is used.



- B15 to B13: System use
- B10: Transmission time exceeded
- B07: Address error
- B05: Reference error
- B04: Parameter error
- B03: Buffer size error
- B12, B11, B06, B02 to B00: For future use

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact (—|↑|—). Once execution status is entered, execution continues until it is completed (However, the operation is not completed normally if a fault such as transmission error, time-out, etc. occurs.)

This input is not accepted during execution (before completion) even if the command is turned on or off.

Input 2: For future use

Input 3: Abort command

The command execution is aborted. This input signal is a level signal with priority to input 1.

② Outputs

Output 1: During execution

The command is under execution.

Output 2: Error

Indicates that the command execution is completed with an error. Only 1 scan is turned on and the error contents are reflected in the status.

Output 3: Completion

Only 1 scan is turned on at normal completion.

NOTE When the execution start command is input by NO contact(—| |—), the contact is turned on at communication completion.

5.18.3 BROD

(1) Function

MBUS writing-in function is performed by broadcasting (simultaneous broadcasting).

Used when the same data are transmitted to GL60H or GL70H (including self-station) via all network 3200IF modules. However, data are not transmitted to the units connected to NSU.

(2) Form

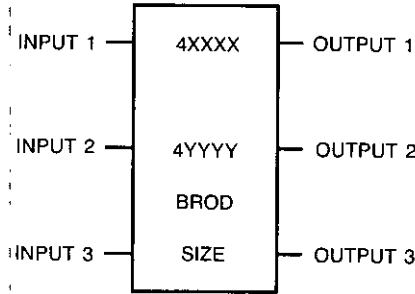


Fig. 5.99 BROD Form

Table 5.122 BROD Elements

Element	Description	Specified Number
Top	BROD control register	Holding register (40001 to 49987)
Middle	Transmission buffer	Holding register (40001 to 49994)
Bottom	Size	Constant (1 to 100)

(a) Outline of top element

Based on the data information specified by this control block, data are written-in to all stations.

4 XXXX	NETWORK ADDRESS
4 XXXX +1	FUNCTION CODE
4 XXXX +2	REFERENCE NO.
4 XXXX +3	DATA COUNT
4 XXXX +4	STAUS
4 XXXX +5	SYSTEM USE

NOTE 6 holding registers from 4XXXX are used exclusively.

(b) Outline of 4XXXX +4

Status after execution is stored. It is updated before BROD execution start. 4XXXX +4 is used.

15																			0
B15	B14	B13	B12	B11	B10	B09	B08	B07	B06	B05	B04	B03	B02	B01	B00				

B15 to B13: System use
B10: Transmission time exceeded
B07: Address error
B06: Function error
B05: Reference error
B04: Parameter error
B03: Buffer size error
B12, B11, B02 to B00: For future use

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). Once execution status is entered, execution continues until it is completed (however, the operation is not completed normally if a fault such as transmission error, time-out, etc. occurs).

This input is not accepted during execution (before completion) even if the command is turned on or off.

Input 2: For future use

Input 3: Abort command

The command execution is aborted. This input signal is a level signal with priority to input 1.

② Outputs

Output 1: During execution

The command is under execution.

Output 2: Error

Indicates that the command execution is completed with an error. Only 1 scan is turned on and the error contents are reflected in the status.

Output 3: Completion

Only 1 scan is turned on at normal completion.

NOTE

When the execution start command is input by NO contact ($\neg \uparrow \neg$), the contact is turned on at communication completion.

5.18.4 BOOK

(1) Operation

Communication is performed between one master and one slave station.

As well as MBUS, coils, input relays, holding registers or input registers are read out (only for reading-out).

When the same data are read out, this command must be used together with POLL command described in the next paragraph so that reading efficiency of coils, input relays or registers can be improved. By executing the BOOK command, the GL60H or GL70H which receives this command prepares every scan of data specified by the command. To read out these data, the POLL command must be executed. The latest data are available since data are updated at every scan.

However, this command is cancelled when MEMOBUS command is received from the node to which the command is sent after receiving this command.

This command is effective only for GL60H or GL70H connected by the 3200IF module. Broadcasting (simultaneous broadcasting) is not possible.

POINT

The BOOK command Performs booking and does not have response data, being different from the MBUS command. Therefore, the time until completion is short. After that, the POLL command is transmitted whenever data are required. Data can be obtained as many times as possible by the POLL command unless the BOOK command is released.

NOTE

GL60H or GL70H that has received the BOOK command keeps execution unless it is cancelled and the 3200IF module is reset. When the execution is not necessary, send the MEMOBUS command and cancel it (or execute the MBUS command).

(2) Form

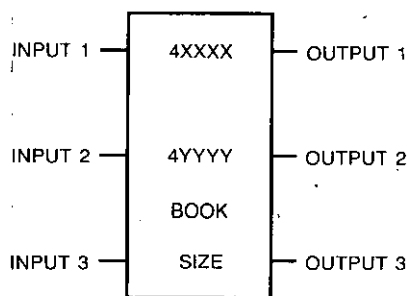


Fig. 5.100 BOOK Form

Table 5.123 BOOK Elements

Element	Description	Specified Number
Top	BOOK control register	Holding register (40001 to 49981)
Middle	Dummy buffer	Holding register (40001 to 49999)
Bottom	Size	Constant (1 to 125)

(a) Outline of top element

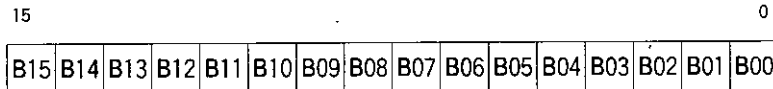
Based on the data information specified by this control block, data read-out is booked.

4 XXXX	NETWORK ADDRESS
4 XXXX +1	DESTINATION ADDRESS
4 XXXX +2	FUNCTION CODE
4 XXXX +3	REFERENCE NO.
4 XXXX +4	DATA COUNT
4 XXXX +5	RETRY COUNT
4 XXXX +6	TIMER
4 XXXX +7	STATUS
4 XXXX +8	SYSTEM USE

NOTE 9 holding registers from 4XXXX are used exclusively.

(b) Outline of 4XXXX +7

Status after execution is stored. It is updated before BOOK execution start. 4XXXX +7 is used.



- B15 to B13: System use
- B12: Error response receiving
- B11: Receiving time exceeded
- B10: Transmission time exceeded
- B07: Address error
- B06: Function error
- B05: Reference error
- B04: Parameter error
- B03: Buffer size error
- B02 to B00: No. of retry operations

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). Once execution status is entered, execution continues until it is completed (however, the operation is not completed normally if a fault such as transmission error, time-out, etc. occurs.).

This input is not accepted during execution (before completion) even if the command is turned on or off.

Input 2: For future use

Input 3: Abort command

The command execution is aborted. This input signal is a level signal with priority to input 1.

NOTE

When the execution start command is input by NO contact ($\neg \neg$), the contact is turned on at communication completion.

② Outputs

Output 1: During execution

The command is under execution.

Output 2: Error

Indicates that the command execution is completed with an error. Only 1 scan is turned on and the error contents are reflected in the status.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.18.5 POLL

(1) Function

This command cannot be used alone. It must be used with the BOOK command described in the former paragraph in the specified sequence. As described in the former paragraph, this command is very effective to improve the reading communication efficiency.

This command is effective only for GL60H or GL70H connected by the 3200IF module. Broadcasting (simultaneous broadcasting) is not possible.

The POLL command is an exclusive command to read out the data updated by GL60H or GL70H that has received the BOOK command.

(2) Form

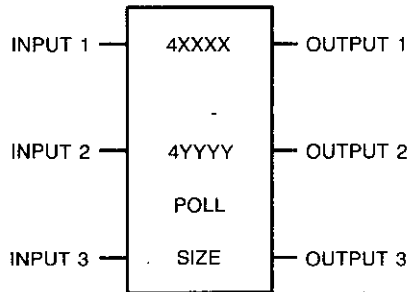


Fig. 5.101 POLL Form

Table 5.124 POLL Elements

Element	Description	Specified Number
Top	POLL control register	Holding register (40001 to 49987)
Middle	Transmission buffer	Holding register (40001 to 49994)
Bottom	Size	Constant (1 to 125)

(a) Outline of top element

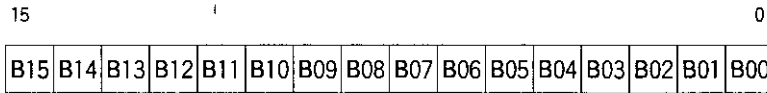
Based on the data information specified by the control block, data are read out from the specified node.

4 XXXX	NETWORK ADDRESS
4 XXXX +1	DESTINATION ADDRESS
4 XXXX +2	RETRY COUNT
4 XXXX +3	TIMER
4 XXXX +4	STATUS
4 XXXX +5	SYSTEM USE

NOTE 6 holding registers from 4XXXX are used exclusively.

(b) Outline of 4 XXXX +4

Status after execution is stored. It is updated at every POLL execution start. 4XXX +4 is used.



- B15 to B13: System use
- B12: Error response receiving
- B11: Receiving time exceeded
- B10: Transmission time exceeded
- B07: Address error
- B04: Parameter error
- B03: Buffer size error
- B02 to B00: No. of retry operations
- B06, B05: For future use

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). Once execution status is entered, execution continues until it is completed (however, the operation is not completed normally if a fault such as transmission error, time-out, etc. occurs).

This input is not accepted during execution (before completion) even if the command is turned on or off.

Input 2: For future use

Input 3: Abort command

The command execution is aborted. This input signal is a level signal with priority over input 1.

NOTE When the execution start command is input by NO contact ($\neg \neg$), the contact is turned on at communication completion.

② Outputs

Output 1: During execution

The command is under execution.

Output 2: Error

Indicates that the command execution is completed with an error. Only 1 scan is turned on and the error contents are reflected in the status.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.18.6 DIAG

(1) Function

Node diagnosis on LAN is performed. However, this command is effective only for the Memocon-SC devices (supporting MEMOBUS) connected to the 3200IF module.

Status whether node is ready or not is set to the specified register by the bit information.

Up to 8 nodes can be specified.

This command cannot perform broadcasting (simultaneous broadcasting).

(2) Form

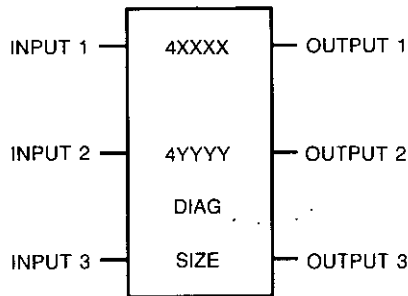


Fig. 5.102 DIAG Form

Table 5.125 DIAG Elements

Element	Description	Specified Number
Top	DIAG control register	Holding register (40001 to 49988)
Middle	Node status	Holding register (40001 to 49999)
Bottom	Constant	Constant (1)

(a) Outline of top element

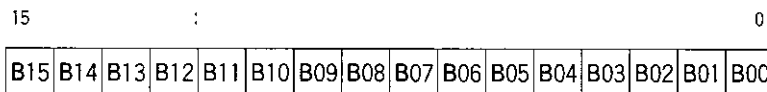
Based on the data information specified by the control block, up to 8 nodes are provided with diagnosis.

4 XXXX	NETWORK ADDRESS
4 XXXX + 1	DESTINATION ADDRESS #1
4 XXXX + 8	DESTINATION ADDRESS #8
4 XXXX + 9	TIMER
4 XXXX + 10	STATUS
4 XXXX + 11	SYSTEM USE

NOTE 12 holding registers from 4XXXX are used exclusively.

(b) Outline of 4 XXXX +10

Status after execution is stored. It is updated at every DIAG execution start. 4XXXX +10 is used.

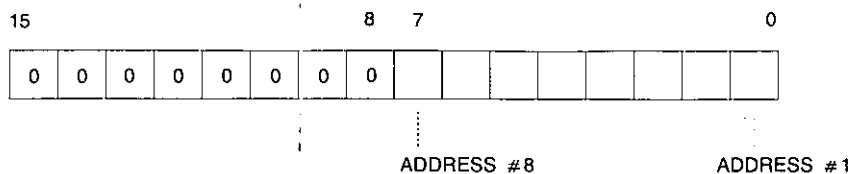


- B15 to B13: System use
- B12: Error response receiving
- B11: Receiving time exceeded
- B10: Transmission time exceeded
- B07: Address error
- B04: Parameter error
- B06, B05, B03 to B00: For future use

(c) Outline of middle element

Node status buffer is specified.

Buffer size (=1) is specified by the bottom element.



- 0 NO RESPONSE. NO DIAGNOSIS SPECIFIED
- 1 NORMAL RESPONSE

The bit arrangement of the relevant station corresponds to the top element destination address specified register in the order one by one.

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \mid$). Once execution status is entered, execution continues until it is completed (However, the operation is not completed normally if a fault such as transmission error, time-out, etc. occurs).

This input is not accepted during execution (before completion) even if the command is turned on or off.

Input 2: For future use

Input 3: Abort command

The command execution is aborted. This input signal is a level signal with priority over input 1.

NOTE

When the execution start command is input by NO contact ($\neg \mid$), the contact is turned on at communication completion.

② Outputs

Output 1: During execution

The command is under execution.

Output 2: Error

Indicates that the command execution is completed with an error. Only 1 scan is turned on and the error contents are reflected in the status.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.18.7 SND

(1) Funcion

SND command is used to transmit free data. Normally, it is used together with the "RCV" command described in the next paragraph.

This command can be used between GL60H and GL70H.

When the 3200IF module receives data by the SND command and the RCV command is not executed, data can be obtained immediately by executing the RCV command since data are stored in the 3200IF module receiving buffer. However, when more than one SND command is received, only the last data can be received.

(2) Form

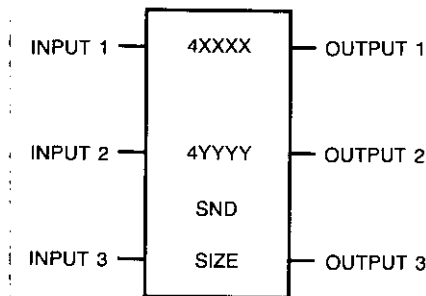


Fig. 5.103 SND Form

Table 5.126 SND Elements

Element	Description	Specified Number
Top	SND control register	Holding register (40001 to 49995)
Middle	Transmission buffer	Holding register (40001 to 49999)
Bottom	Size	Constant (1 to 250)

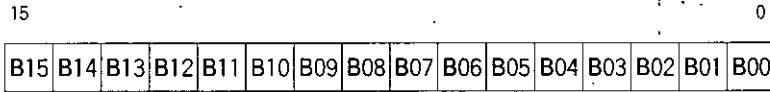
(a) Outline of top element

4 XXXX	NETOWRK ADDRESS
4 XXXX +1	DESTINATION ADDRESS
4 XXXX +2	DATA COUNT
4 XXXX +3	STATUS
4 XXXX +4	SYSTEM USE

NOTE 5 holding registers from 4XXXX are used exclusively.

(b) Outline of 4XXXX+3

Status after execution is stored. It is updated at every SND execution start. 4XXXX+3 is used.



- B15 to B13: System use
- B10: Transmission time exceeded
- B07: Address error
- B04: Parameter error
- B03: Buffer size error
- B12, B11, B06, B05, B02 to B00: For future use

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact (−↑|−). Once execution status is entered, execution continues until it is completed (however, the operation is not completed normally if a fault such as transmission error, timeout, etc. occurs).

This input is not accepted during execution (before completion) even if the command is turned on or off.

Input 2: For future use

Input 3: Abort command

The command execution is aborted. This input signal is a level signal with priority over input 1.

NOTE When the execution start command is input by NO contact (−| |), the contact is turned on at communication completion.

② Outputs

Output 1: During execution

The command is under execution.

Output 2: Error

Indicates that the command execution is completed with an error. Only 1 scan is turned on and the error contents are reflected in the status.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.18.8 RCV

(1) Function

The RCV command is used to receive free data.

It is often used together with the SND command, but it can be used alone only for receiving.

(2) Form

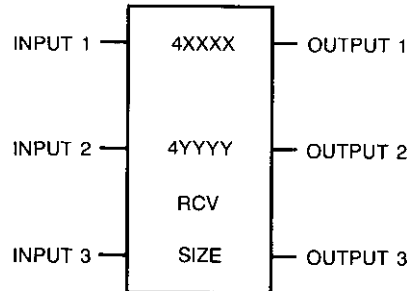


Fig. 5.104 RCV Form

Receiving data when the receiving command is not executed are not revoked but stored in the receiving buffer. Therefore, data are read-in immediately by executing the RCV command.

However, when the receiving buffer becomes full, no more data are received.

Table 5.127 RCV Elements

Element	Description	Specified Number
Top	RCV control register	Holding register (40001 to 49987)
Middle	Receiving buffer	Holding register (40001 to 49999)
Bottom	Size	Constant (1 to 250)

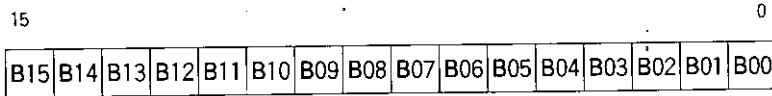
(a) Outline of top element

4 XXXX	NETOWRK ADDRESS
4 XXXX +1	SOURCE ADDRESS
4 XXXX +2	DATA COUNT
4 XXXX +3	TIMER
4 XXXX +4	STATUS
4 XXXX +5	SYSTEM USE

NOTE 6 holding registers from 4XXXX are used exclusively.

(b) Outline of 4 XXXX +4

Status after execution is stored. It is updated at every RCV execution start. 4 XXXX +4 is used.



B15 to B13: System use
B10: Receiving time exceeded
B07: Address error
B04: Parameter error
B03: Buffer size error
B12, B10, B06, B05, B02 to B00: For future use

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). Once execution status is entered, execution continues until it is completed (however, the operation is not completed normally if a fault such as transmission error, time-out, etc. occurs).

This input is not accepted during execution (before completion) even if the command is turned on or off.

Input 2: For future use

Input 3: Abort command

The command execution is aborted. This input signal is a level signal with priority over input 1.

② Outputs

Output 1: During execution

The command is under execution.

Output 2: Error

Indicates that the command execution is completed with an error. Only 1 scan is turned on and the error contents are reflected in the status.

Output 3: Completion

Only 1 scan is turned on at normal completion.

NOTE

When the execution start command is input by NO contact ($\neg \uparrow \neg$), the contact is turned on at communication completion.

5.19 MOTION COMMANDS

Table 5.128 shows the motion command function list. The motion commands execute motion control in a ladder circuit. These commands cannot activate more than one command simultaneously for one axis. However, the monitor command can be activated with the other commands.

Table 5.128 Motion Command Function List

No.	Name	Symbol	Function
1	1-axis Program Operation	MOVI	Performs single-axis program operation from specified block.
2	1-axis Single-block Operation	MVIS	Performs single-axis operation only for one specified block.
3	Independent Multi-axis Program Operation	MOVMM	Performs simultaneous start operation of multi-axis from specified block.
4	3-axis Program Operation	MOVL	Performs 2- or 3-axis program operation from specified block.
5	3-axis Single-block Operation	MVLS	Performs 2- or 3-axis program operation only for one specified block.
6	1-axis Zero-point Return Operation	ZRN	Performs 1-axis zero-point return operation.
7	Jog Operation	JOG	Performs jog operation.
8	Handle Operation	HNDL	Changes to handle operation mode.
9	Monitor	MON	Monitors alarms, parameters, program numbers.
10	Current Position Setting	POS	Changes current position data.
11	Parameter Setting	PRM	Sets one parameter.
12	Variable Value Setting	VAR	Sets a value to variable H1, H2, H3 or H4.
13	Alarm Reset	ARES	Resets alarm after servo alarm occurs.
14	Servo ON	SVON	Makes motor under current conduction. Makes motor not under current conduction.
15	Current Value Adjustment	ADJ	Adjusts current value before 3- or 2-axis operation is performed.

For the details of the motion commands, refer to the motion control user's manual (SIE-C815-15.30).

5.19.1 Outline of Motion Commands

5.19.1.1 1-axis Program Operation (MOV1)

Description

(1) Function

When the START command is input, the operation starts from the specified 0#N# block and continues execution according to the program contents until the program end command (M30) is provided.

It is necessary to load 1-axis motion program in the servopack before the START command is input.

(2) Form

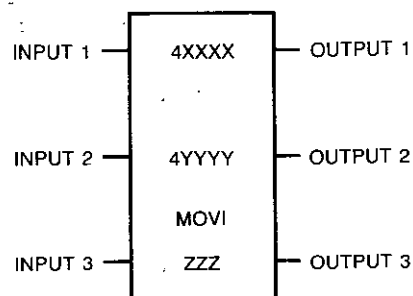


Fig. 5.105 MOV1 Form

Table 5.129 MOV1 Elements

Element	Description	Specified Number
Top	MOV1 control reference	Holding register (40001 to 49994)
Middle	Current value, status	Holding register (40001 to 49997)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 1-BIT
4 XXXX +2	0 NUMBER	MOTION PROGRAM 0#: 1 TO 20 (DEC)
4 XXXX +3	N NUMBER	MOTION PROGRAM N#: 1 TO 999 (DEC)
4 XXXX +4	FEEDHOLD SELECTION	1 WHEN INPUT 2 IS USED AS FEEDHOLD COMMAND
4 XXXX +5	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). If the command is turned on or off during execution, it is not accepted. If input 3 is turned on before this input, it is also not accepted.

Input 2: Continuous operation command when FEEDHOLD selection flag $\neq 1$

MFIN signal is output by startup differential contact ($\neg \uparrow \neg$). However, this input is effective only during MFIN request (M50 execution).

Stop command when FEEDHOLD selection flag = 1

The operation stops temporarily while this input is turned on. By turning it off, the rest of the program is executed. This input is not accepted unless output 1 is turned on.

Input 3: Abort command

The specified axis operation is aborted.

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at operation completion or STOP completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.2 1-axis Single-block Operation (MVIS)

(1) Function

Executes a program by one block in the 1-axis motion program.

(2) Form

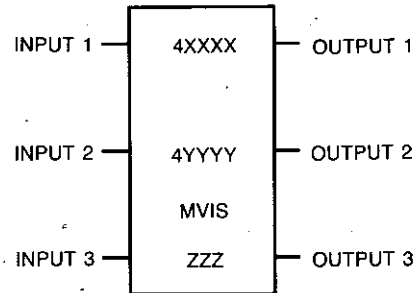


Fig. 5.106 MVIS Form

Table 5.130 MVIS Elements

Element	Description	Specified Number
Top	MVIS control reference	Holding register (40001 to 49995)
Middle	Current value, status	Holding register (40001 to 49997)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 1-BIT
4 XXXX +2	O NUMBER	MOTION PROGRAM O#: 1 TO 20 (DEC)
4 XXXX +3	N NUMBER	MOTION PROGRAM N#: 1 TO 999 (DEC)
4 XXXX +4	FEEDHOLD SELECTION	1 WHEN INPUT 2 IS USED AS FEEDHOLD COMMAND
4 XXXX +5	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). If the command is turned on or off during execution, it is not accepted. If input 3 is turned on before this input, it is also not accepted.

Input 2: Continuous operation command when FEEDHOLD selection flag $\neq 1$
MFIN signal is output by startup differential contact ($\neg \uparrow \neg$). However, this input is effective only during MFIN request (M50 execution).

Stop command when FEEDHOLD selection flag = 1

The operation stops temporarily while this input is turned on. By turning it off, the rest of the program is executed. This input is not accepted unless output 1 is turned on.

Input 3: Abort command

The specified axis operation is aborted.

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at operation completion or STOP completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.3 Independent Multi-axis Program Operation (MOVMM)

(1) Function

The MOVMM operation is used when each axis of the servopack is started simultaneously. Each axis starts operating at the same time and continues operation until its program M30 (operation end) is provided.

The program number (O number) and sequence block number (N number) are the common data for each axis. Therefore, a motion program with the same O number and the same sequence block number is needed. If even one axis does not have the operation program, axes that have already started operating stop the operation.

(2) Form

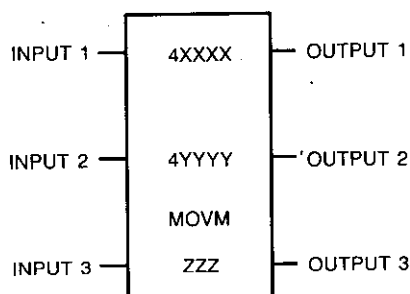


Fig. 5.107 MOVMM Form

Table 5.131 MOVMM Elements

Element	Description	Specified Number
Top	MOVMM control reference	Holding register (40001 to 49993)
Middle	Current value, status	Holding register (40001 to 49976)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE ANY BIT
4 XXXX +2	O NUMBER	MOTION PROGRAM O#: 1 TO 20 (DEC)
4 XXXX +3	N NUMBER	MOTION PROGRAM N#: 1 TO 999 (DEC)
4 XXXX +4	MFIN AXIS NUMBER	AXIS TO OUTPUT MFIN: 1 TO LOWER DIGIT BYTE ANY BIT
4 XXXX +5	FEEDHOLD SELECTION	1 WHEN INPUT 2 IS USED AS FEEDHOLD COMMAND
4 XXXX +6	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact $(-\uparrow-)$. If the command is turned on or off during execution, it is not accepted. If input 3 is turned on before this input, it is also not accepted.

Input 2: Continuous operation command when FEEDHOLD selection flag $\neq 1$
MFIN signal is output by startup differential contact $(-\uparrow-)$. However, this input is effective only during MFIN request (M50 execution).

Stop command when FEEDHOLD selection flag = 1

The operation stops temporarily while this input is turned on. By turning it off, the rest of the program is executed. This input is not accepted unless output 1 is turned on.

Input 3: Abort command

The specified axis operation is aborted.

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at operation completion or STOP completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.4 3-axis (2-axis) Program Operation (MOVL)

(1) Function

The MOVL operation is used when interpolation operation is performed by 3- or 2-axis.

Execute the MOVL after loading the same motion program to specified interpolation axes in the servopack and executing current value adjustment (ADJ) command.

(2) Form

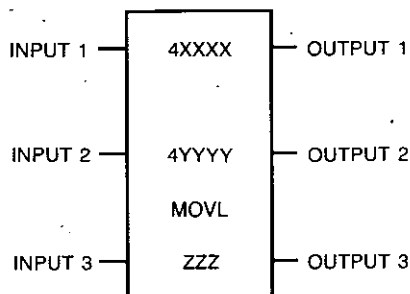


Fig. 5.108 MOVL Form

Table 5.132 MOVL Elements

Element	Description	Specified Number
Top	MOVL control reference	Holding register (40001 to 49994)
Middle	Current value, status	Holding register (40001 to 49991)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 2 OR 3 BITS
4 XXXX +2	O NUMBER	MOTION PROGRAM O#: 1 TO 20 (DEC)
4 XXXX +3	N NUMBER	MOTION PROGRAM N#: 1 TO 999 (DEC)
4 XXXX +4	FEEDHOLD SELECTION	1 WHEN INPUT 2 IS USED AS FEEDHOLD COMMAND
4 XXXX +5	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). If the command is turned on or off during execution, it is not accepted. If input 3 is turned on before this input, it is also not accepted.

Input 2: Continuous operation command when FEEDHOLD selection flag $\neq 1$
MFIN signal is output by startup differential contact ($\neg \uparrow \neg$). However, this input is effective only during MFIN request (M50 execution).

Stop command when FEEDHOLD selection flag = 1

The operation stops temporarily while this input is turned on. By turning it off, the rest of the program is executed. This input is not accepted unless output 1 is turned on.

Input 3: Abort command

The specified axis operation is aborted.

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at operation completion or STOP completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.5 3-axis (2-axis) Single-block Operation (MVLS)

(1) Function

In 3-axis (2-axis) motion program, one block is executed after another. It is necessary to execute the current value adjustment (ADJ) command for the specified axis and perform 3-axis (2-axis) position adjustment. When one block after another is executed continuously, activate the ADJ command once before the MVLS execution.

(2) Form

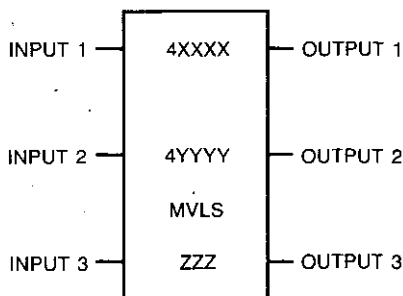


Fig. 5.109 MVLS Form

Table 5.133 MVLS Elements

Element	Description	Specified Number
Top	MVLS control reference	Holding register (40001 to 49994)
Middle	Current value, status	Holding register (40001 to 49991)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 2 OR 3 BITS
4 XXXX +2	O NUMBER	MOTION PROGRAM O# : 1 TO 20 (DEC)
4 XXXX +3	N NUMBER	MOTION PROGRAM N# : 1 TO 999 (DEC)
4 XXXX +4	FEEDHOLD SELECTION	1 WHEN INPUT 2 IS USED AS FEEDHOLD COMMAND
4 XXXX +5	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow |$). If the command is turned on or off during execution, it is not accepted. If input 3 is turned on before this input, it is also not accepted.

Input 2: Continuous operation command when FEEDHOLD selection flag $\neq 1$
MFIN signal is output by startup differential contact ($\neg \uparrow |$). However, this input is effective only during MFIN request (M50 execution).

Stop command when FEEDHOLD selection flag = 1

The operation stops temporarily while this input is turned on. By turning it off, the rest of the program is executed. This input is not accepted unless output 1 is turned on.

Input 3: Abort command

The specified axis operation is aborted.

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at operation completion or STOP completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.6 1-axis Zero Point Return Operation (ZRN)

(1) Funcion

1-axis zero-point return operation is performed.

There are two types of the zero-point return operation patterns: mode I and mode II. Select the mode in advance according to the parameter setting.

(2) Form

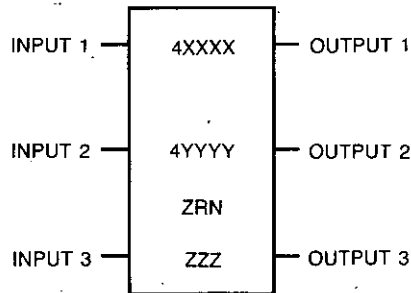


Fig. 5.110 ZRN Form

Table 5.134 ZRN Elements

Element	Description	Specified Number
Top	ZRN control reference	Holding register (40001 to 49997)
Middle	Current value, status	Holding register (40001 to 49997)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 1 BIT
4 XXXX +2	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). If the command is turned on or off during execution, it is not accepted. If input 3 is turned on before this input, it is also not accepted.

Input 2: Reverse run command

Rotating direction is specified. OFF: forward run direction.
ON: Reverse run direction. This input is effective only at execution command input.

Input 3: Abort command

The specified axis operation is aborted.

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at operation completion or STOP completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.7 Jog Operation (JOG)

(1) Funcion

The JOG command is used when it is required to move the machine by continuous manual feeding. The feeding speed is specified among the parameter set speed values of 1 to 4.

(2) Form

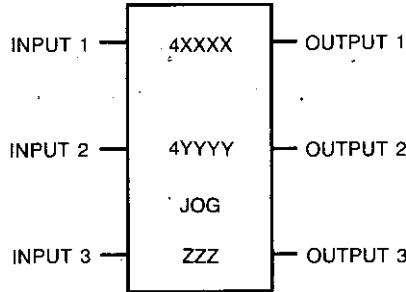


Fig. 5.111 JOG Form

Table 5.135 JOG Elements

Element	Description	Specified Number
Top	JOG control reference	Holding register (40001 to 49997)
Middle	Current value, status	Holding register (40001 to 49997)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 1 BIT
4 XXXX +2	SPEED NUMBER	FEEDING SPEED NO.: 1 TO 4
4 XXXX +3	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). If the command is turned on or off during execution, it is not accepted. If input 3 is turned on before this input, it is also not accepted.

Input 2: For future use

Input 3: Reverse run command

Rotating direction is specified. OFF: forward run direction.

ON: Reverse run direction. This input is effective only at execution command input.

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at operation completion or STOP completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.8 Handle Operation (HNDL)

(1) Function

The HNDL command is used for handle operation and the handle operation mode is entered. Servopack parameter setting must be (P19-b3=1) that uses a pulse reference input. By rotating the handle, the machine moves in the plus or minus direction according to the rotating direction. Three steps of moving amount per one handle scale ($\times 1$, $\times 10$, $\times 100$) can be selected by changing the servopack external input (SP2ND, SP3RD).

(2) Form

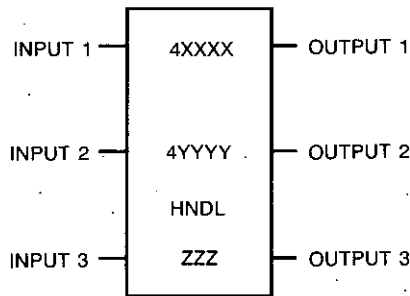


Fig. 5.112 HNDL Form

Table 5.136 HNDL Elements

Element	Description	Specified Number
Top	HNDL control reference	Holding register (40001 to 49997)
Middle	Current value, status	Holding register (40001 to 49997)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 1 BIT
4 XXXX +2	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($-|\uparrow|-$). If the command is turned on or off during execution, it is not accepted. Once execution status is entered, execution (current value, status monitoring) continues until it is released.

Input 2: For future use

Input 3: Release command

Handle mode is released.

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at completion of mode release.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion (code release completion).

5.19.1.9 Monitor (MON)

(1) Function

By executing the monitor command, the specified axis status can be monitored.

(2) Form

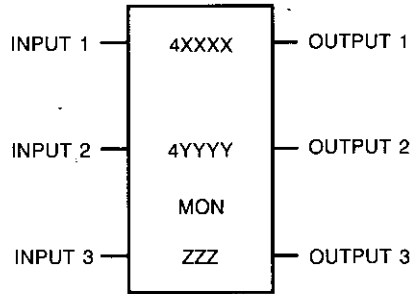


Fig. 5.113 MON Form

Table 5.137 MON Elements

Element	Description	Specified Number
Top	MON control reference	Holding register (40001 to 49997)
Middle	Current value, status	Holding register (40001 to 49997)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 1 BIT
4 XXXX +2	MONITOR NUMBER	MONITOR NO.: REFER TO TABLE 5. 138.
4 XXXX +3	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

Monitor Number

The contents to be monitored are specified.

Table 5.138 Monitor No. and Its Contents

Monitor No.	X	Contents
000X	1	Current position
	2	Position deviation
	3	Current speed
	4	Reference speed
	5	Torque reference
	6	Status
001X	0 ~ 9	0: Current alarm 1 ~ 9: Past alarms 1 to 9 times before
002X	1 ~ 6	IN1 to IN6 status
003X	1 ~ 3	OUT1 to OUT3 status
01XX	1 ~ 99	Refer to the servo parameter list.
0200		Program No. under execution at present
0300		Variables H1, H2, H3, H4 data

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). If the command is turned on or off during execution, it is not accepted. Once execution status is entered, execution (current value, status monitoring) continues until it is released.

Input 2: For future use

Input 3: For future use

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.10 Current Value Setting (POS)

(1) Function

This command rewrites the current position. After the command execution, it changes to a new coordinate axis.

(2) Form

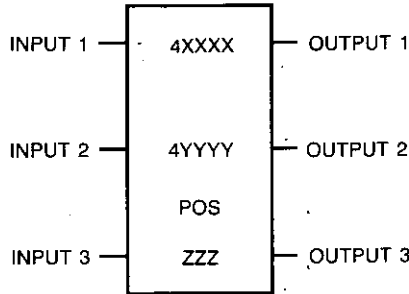


Fig. 5.114 POS Form

Table 5.139 POS Elements

Element	Description	Specified Number
Top	POS control reference	Holding register (40001 to 49995)
Middle	Current value, status	Holding register (40001 to 49999)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 1 BIT
4 XXXX +2	SET VALUE-H	<input type="checkbox"/> 0~+9999 9999 (DEC) [] AT "1" TO SET VALUE-H
4 XXXX +3	SET VALUE-L	<input type="checkbox"/> SET VALUE-L SET VALUE-L MSB
4 XXXX +4	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($-\uparrow-$). If the command is turned on or off during execution, it is not accepted.

Input 2: For future use

Input 3: For future use

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.11 Parameter Setting (PRM)

(1) Function

This command rewrites the servopack parameter number PXX parameters.

Parameters can be changed during operation. There is a limitation of parameter numbers that can be changed by the PRM command.

(2) Form

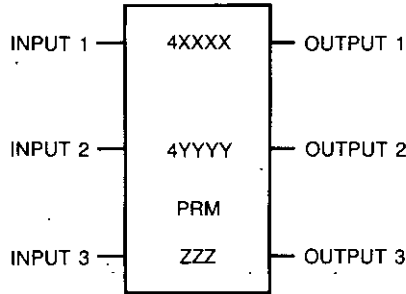


Fig. 5.115 PRM Form

Table 5.140 PRM Elements

Element	Description	Specified Number
Top	POS control reference	Holding register (40001 to 49995)
Middle	Current value, status	Holding register (40001 to 49999)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4									
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 1 BIT									
4 XXXX +2	PARAMETER NUMBER	SERVOPACK PARAMETER NO.: 1 TO 100 (DEC)									
4 XXXX +3	DATA-H	<input type="checkbox"/> <table style="display: inline-table; vertical-align: middle;"> <tr> <td style="text-align: center;">0~+9999</td> <td style="text-align: center;">9999 (DEC)</td> <td style="text-align: center;">[-] AT "1" TO</td> </tr> <tr> <td style="text-align: center;">SET VALUE-H</td> <td style="text-align: center;">SET VALUE-L</td> <td style="text-align: center;">SET VALUE-H</td> </tr> <tr> <td></td> <td></td> <td style="text-align: center;">MSB</td> </tr> </table>	0~+9999	9999 (DEC)	[-] AT "1" TO	SET VALUE-H	SET VALUE-L	SET VALUE-H			MSB
0~+9999	9999 (DEC)		[-] AT "1" TO								
SET VALUE-H	SET VALUE-L	SET VALUE-H									
		MSB									
4 XXXX +4	DATA-L										
4 XXXX +5	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM									

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). If the command is turned on or off during execution, it is not accepted.

Input 2: For future use

Input 3: For future use

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

NOTE

When any data other than the setting range are set by parameter data, the servopack has the maximum value of the parameter effective data range as a set value. Errors are not processed.

5.19.1.12 Variable Value Setting (VAR)

(1) Function

By using variables H1, H2, H3 and H4, a motion program can be created. VAR is a command to set the value of these variables.

(2) Form

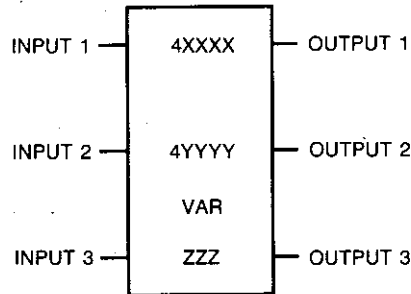


Fig. 5.116 VAR Form

Table 5.141 VAR Elements

Element	Description	Specified Number
Top	VAR control reference	Holding register (40001 to 49989)
Middle	Current value, status	Holding register (40001 to 49999)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX + 1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 1 BIT
4 XXXX + 2	H1 DATA-H	<input type="checkbox"/> 0~±9999 SET VALUE-H 9999 (DEC) SET VALUE-L [] AT "1" TO SET VALUE-H MSB
4 XXXX + 3	H1 DATA-L	
4 XXXX + 4	H2 DATA-H	
4 XXXX + 5	H2 DATA-L	
4 XXXX + 6	H3 DATA-H	
4 XXXX + 7	H3 DATA-L	
4 XXXX + 8	H4 DATA-H	
4 XXXX + 9	H4 DATA-L	
4 XXXX + 10	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($-|\uparrow|-$). If the command is turned on or off during execution, it is not accepted.

Input 2: For future use

Input 3: For future use

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.13 Alarm Reset (ARES)

(1) Function

When the servopack generates an alarm, this command resets the alarm. (The same function as that of servopack alarm reset.)

(2) Form

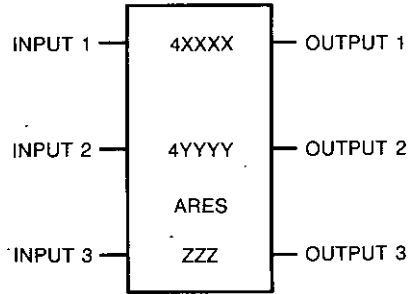


Fig. 5.117 ARES Form

Table 5.142 ARES Elements

Element	Description	Specified Number
Top	ARES control reference	Holding register (40001 to 49997)
Middle	Current value, status	Holding register (40001 to 49999)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 1 BIT
4 XXXX +2	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($-\uparrow|$). If the command is turned on or off during execution, it is not accepted.

Input 2: For future use

Input 3: For future use

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.14 Servo ON (SVON)

(1) Function

By the SVON command, the servopack main circuit power drive circuit operates and the motor is under current conduction.

(2) Form

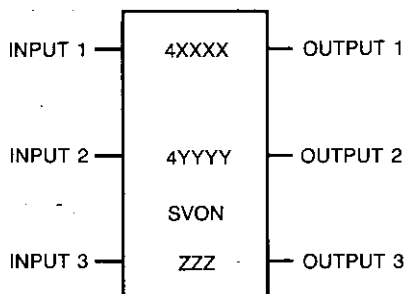


Fig. 5.118 SVON Form

Table 5.143 SVON Elements

Element	Description	Specified Number
Top	SVON control reference	Holding register (40001 to 49997)
Middle	Current value, status	Holding register (40001 to 49997)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 1 BIT
4 XXXX +2	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow \neg$). If the command is turned on or off during execution, it is not accepted.

Input 2: For future use

Input 3: Current conduction stop command

When input 1 is turned on while this input is turned on, the servo OFF command is provided.

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.1.15 3-axis (2-axis) Current Value Adjustment (ADJ)

(1) Function

When interpolation operation is performed by 3-axis (2-axis) adjustment, this command matches the current position between three (two) axes.

Since each servopack controls only the current position of its own axis under operation, it is necessary to teach the current positions of three axes, X, Y and Z in the interpolation program of G01 XX1 YY1 ZZ1 FF1; so that the servopack can operate its own axis feeding speed correctly according to the X- Y- or Z- axis moving amount.

(2) Form

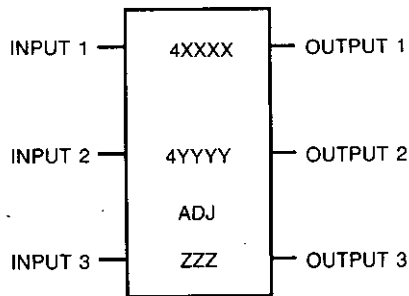


Fig. 5.119 ADJ Form

Table 5.144 ADJ Elements

Element	Description	Specified Number
Top	ADJ control reference	Holding register (40001 to 49997)
Middle	Current value, status	Holding register (40001 to 49991)
Bottom	Constant	Constant (1 to 999)

Setting Data

4 XXXX	MODULE NUMBER	IF66 NO.: 1 TO 4
4 XXXX +1	AXIS NUMBER	SERVOPACK AXIS NO.: 1 TO LOWER DIGIT BYTE 2 OR 3 BITS
4 XXXX +2	SYSTEM USE	USED AS EXECUTION FLAG IN SYSTEM

(3) Operation

① Inputs

Input 1: Execution command

The command execution is started by the command startup differential contact ($\neg \uparrow | -$). If the command is turned on or off during execution, it is not accepted.

Input 2: For future use

Input 3: For future use

When input 1 is turned on while this input is turned on, the servo OFF command is provided.

② Outputs

Output 1: During operation

This output is turned on during command execution. It is turned off at completion.

Output 2: Error

Only 1 scan is turned on at error completion.

Output 3: Completion

Only 1 scan is turned on at normal completion.

5.19.2 Status

Status after start is displayed in a holding register status.

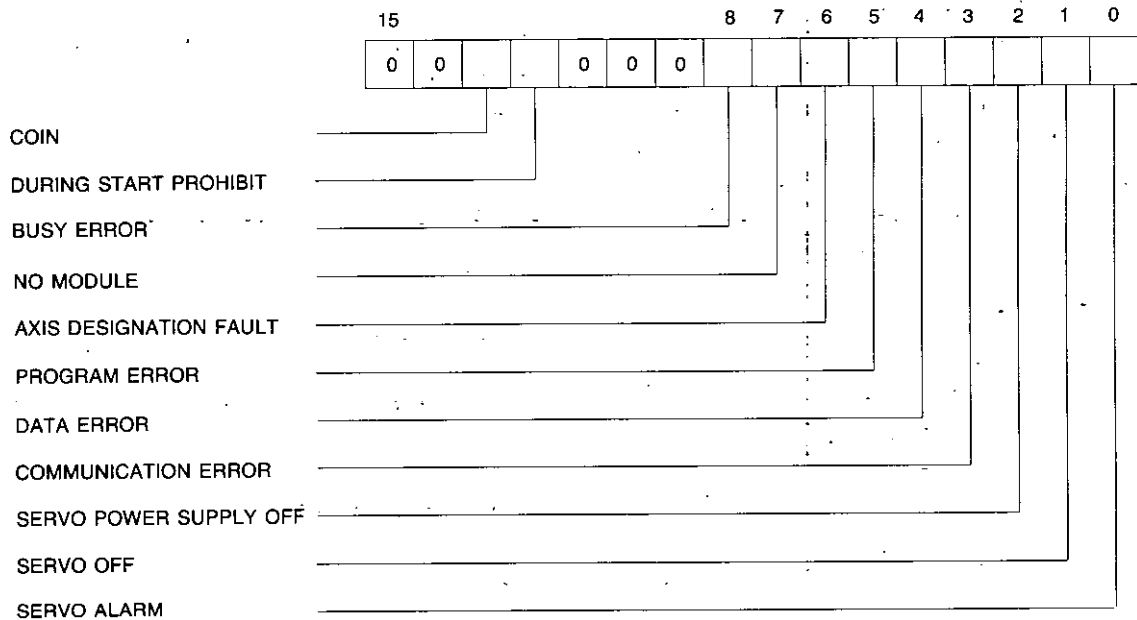


Table 5.145 Status

	Item	Bit Data	Contents
b0	Servo alarm	1	An alarm occurs at servo side. The alarm contents are monitored by MON command.
b1	Servo OFF	1	Motor is not under current conduction. Servo OFF atatus
b2	Servo power supply OFF	1	Servo control power supply is not turned on.
b3	Communication error	1	An error occurs in communication between IF66 and servo.
b4	Data error	1	An error occurs in setting dta. It occurs when; out of module No. range (except 1 to 4), out of O number range (except 1 to 20), out of N number range (except 1 to 999) or relevant 0 or N No. is not provided.
b5	Program error	1	An error occurs in the motion program.
b6	Axis designation error	1	Axis designation is out of the axis No. range or 3-axis and 2-axis designation does not meet the program.
b7	No module	1	Specified IF66 module is not provided.
b8	Busy error	1	Other motion command is under execution.
b12	During start prohibit	1	Occurs during motion program and parameter loading/saving.
b13	Coin	1	Difference between current value and aimed value given by reference data (reference unit) is within the range set by parameter 6 (P06).

5.19.3 M Relay, N Relay

M relays and N relays which can be used in the system are available for the GL70H.

(1) M50 relay

MFIN relay

┌ ─┐
M50XY X: IF66 module Nos. 1 to 4
 Y: Axis Nos. 1 to 8

When MFIN required motion program M50 is executed by Y-axis of module No. X, M50 relay (reference No.: M50XY) corresponding to the program is turned on. At this time, the Y-axis is in waiting status for MFIN. In this status, when input 2 of the motion commands (MOV1, MOV M, MOV L) is turned on (or MFIN is output), the next block is executed and the M50 relay is turned off.

(2) M51 relay

Notch signal relay

┌ ─┐
M51XY X: IF66 module Nos. 1 to 4
 Y: Axis Nos. 1 to 8

When M51 is executed by Y-axis of module No. X, the M51 relay (reference No.: M51XY) corresponding to the program is turned on. The program is executed without waiting, being different from the M50. When the next program is executed, the M51 relay is turned off.

(3) N00 relay

NEAR relay

┌ ─┐
N00XY X: IF66 module Nos. 1 to 4
 Y: Axis Nos. 1 to 8

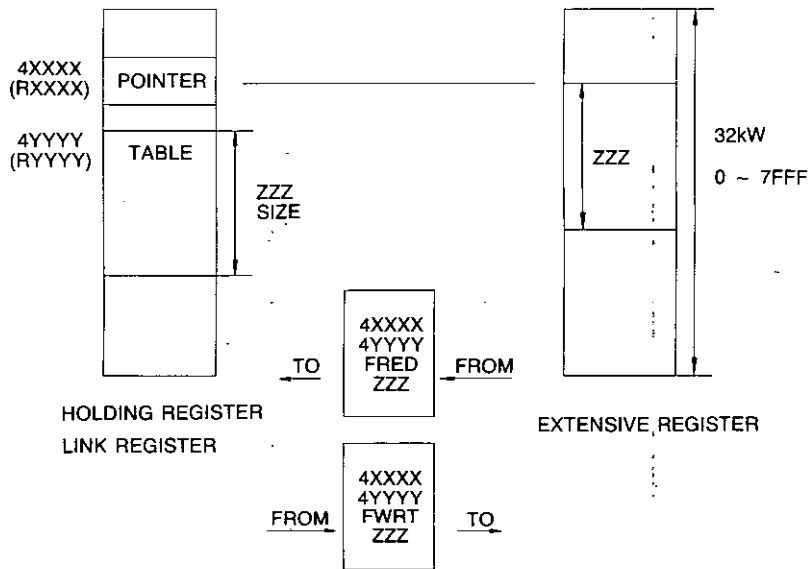
When the difference between the current value of X module Y-axis and aimed value given by reference data (reference unit) becomes lower than the value determined by the positioning NEAR range of parameter 45 (P45), N00 relay (reference No: NOOXY) is turned on.

(4) Supplement

M or N relay ON/OFF timing is delayed more than actual status, the number of control axes × approx. 100ms.

5.20 EXTENSIVE REGISTER ACCESS COMMANDS

32kW extensive registers are available for the Memocon-SC GL60H or GL70H. The extensive registers can be accessed in a ladder by using FRED (FILE READ) command or FWRT (FILE WRITE) command.



NOTE

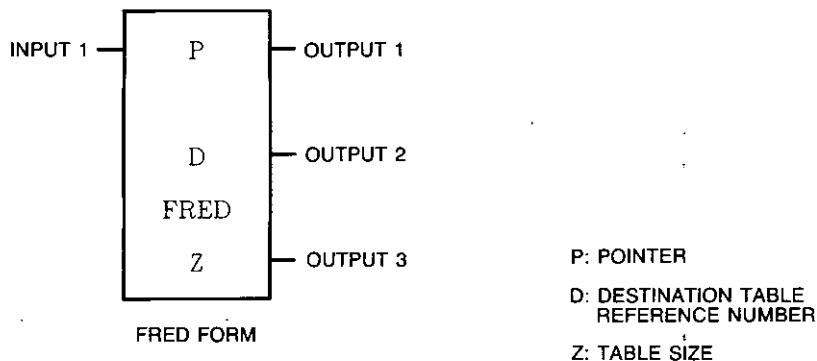
- In the programming panel, it is possible to monitor or change (set) the contents of the extensive registers.
- Reference when the extensive registers are accessed in the programming panel is expressed in hexadecimal of A0000 to A7FFFh.
- Axxxx cannot be read directly by the ladder command.

5.20.1 File Read (FRED)

(1) Function

Data can be read-in by the extensive registers.

(2) Form



- The above figure shows the file read form.
- FRED is a symbol to indicate the file read.
- FRED needs three elements (top, middle, bottom) in the vertical direction to the network.

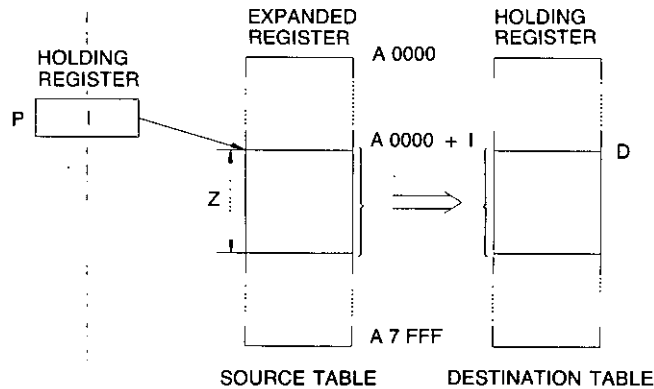
Refer to Table 5.146 for designation.

Table 5.146 FRED Elements

Element	Description	Specified Number
Top	Reference address to which expanded register is transferred by pointer is specified.	Holding register (40001 to 49999) Link register (R0001 to R1024)
Middle	Reference address of destination table is specified.	Holding register (40001 to 49999) Link register (R0001 to R1024)
Bottom	Table size	Constant (1 to 100)

NOTE The expanded register No. specified by the top element pointer is expressed in hexadecimal of 0 to 7FFFh. The pointer 0 indicates the expanded register reference number.

(3) Operation



All registers are transferred by one scan.

① Input

Input 1: When this input is turned on, the first data item of the source table (expanded register) indicated by the pointer content (I) is transferred to the first part of the destination table and the second to the second; that is, all source data are transferred to the destination table by one scan.

Input 2: For future use

Input 3: For future use

② Output

Output 1: Copy of input 1.

Output 2: Turned on when input 1 is turned on and the pointer (P) content (I) exceeds 7FFFh.

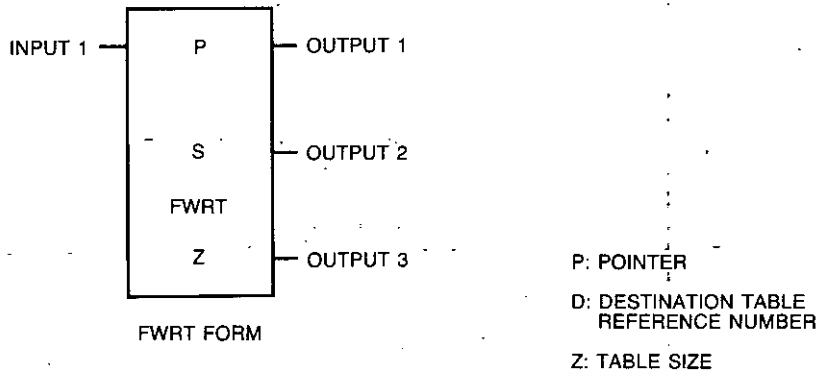
Output 3: Turned on when input 1 is turned on and the sum of the pointer (P) content (I) and table size (Z) exceeds 7FFFh.

5.20.2 File Write (FWRT)

(1) Function

Data can be written-in to the expanded registers.

(2) Form



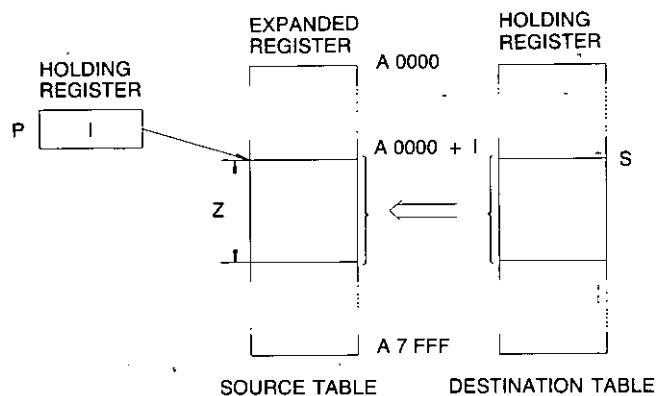
- The above figure shows the file write form.
- FWRT is a symbol to indicate the file write.
- FWRT needs three elements (top, middle, bottom) in the vertical direction to the network. Refer to Table 5.147 for designation.

Table 5.146 FRED Elements

Element	Description	Specified Number
Top	Reference address to which expanded register is transferred by pointer is specified.	Holding register (40001 to 49999) Link register (R0001 to R1024)
Middle	Reference address of source table is specified.	Holding register (40001 to 49999) Link register (R0001 to R1024)
Bottom	Table size	Constant (1 to 100)

NOTE The expanded register No. specified by the top element pointer is expressed in hexadecimal of 0 to 7FFFh. The pointer 0 indicates the expanded register reference number.

(3) Operation



All registers are transferred by one scan.

① Input

Input 1: When this input is turned on, the first data item of the source table (expanded register) indicated by the pointer content (I) is transferred to the first part of the destination table and the second to the second; that is, all source data are transferred to the destination table by one scan.

Input 2: For future use

Input 3: For future use

② Output

Output 1: Copy of input 1

Output 2: Turned on when input 1 is turned on and the pointer (P) content (I) exceeds 1FFFh.

Output 3: Turned on when input 1 is turned on and the sum of the pointer (P) content (I) and table size (Z) exceeds 7FFFh.

SECTION 6

SFC FUNCTIONS

Table 6.1 lists the elements used in the SFC.

Table 6.1 SFC Program Elements

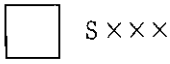
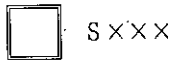
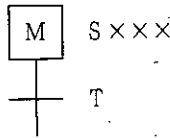
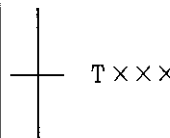
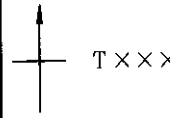


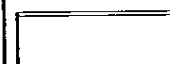
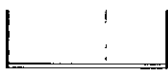

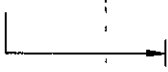

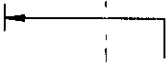
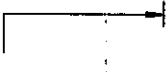

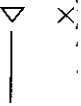
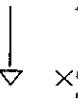


Classification	Name	Symbol	Outline	Refer to Page
Step	Step	 Sxxx	<ul style="list-style-type: none"> ① Shows one unit of execution sequence. ② "xxx" shows the step No. ③ Own action circuit exists. 	307
	Initial Step	 Sxxx	<ul style="list-style-type: none"> ① Step to show program start. ② Becomes active by initialization. ③ "xxx" shows the step No. ④ Action circuit exists. 	307
	Macro Step	 Sxxx T	<ul style="list-style-type: none"> ① Connects master and expanded views. ② Macro step is always provided with dummy transition to be true. ③ "xxx" shows a step No. 	308
Transition	Transition	 Txxx	<ul style="list-style-type: none"> ① Shows execution transition condition between adjoining steps. ② "xxx" shows the transition No. ③ Own transition condition circuit exists. ④ Transition coil inside the transition condition circuit becomes a condition for evolution. 	309
	Counter Transition	 Txxx	<ul style="list-style-type: none"> ① Always used to represent loops with loop links. ② "xxx" shows the transition No. ③ Transition condition circuit that always matches 1 to 1 exists to control number of loop cycles and other items. 	309
Link	Divergence		<ul style="list-style-type: none"> ① Plays to proceed to step with which transition is established among linked steps. ② Diverges from upper part of transition. Divergence is also possible from the left. (From the left transition.) 	310
	Convergence		<ul style="list-style-type: none"> ① Converges divergent processing systems into one. ② Converges to lower part of transition. Convergence is also possible from the left. (From the left transition.) 	310
	Simultaneous Divergence		<ul style="list-style-type: none"> ① Shifts control to simultaneous sequence. ② Diverges from the lower part of transition. 	310

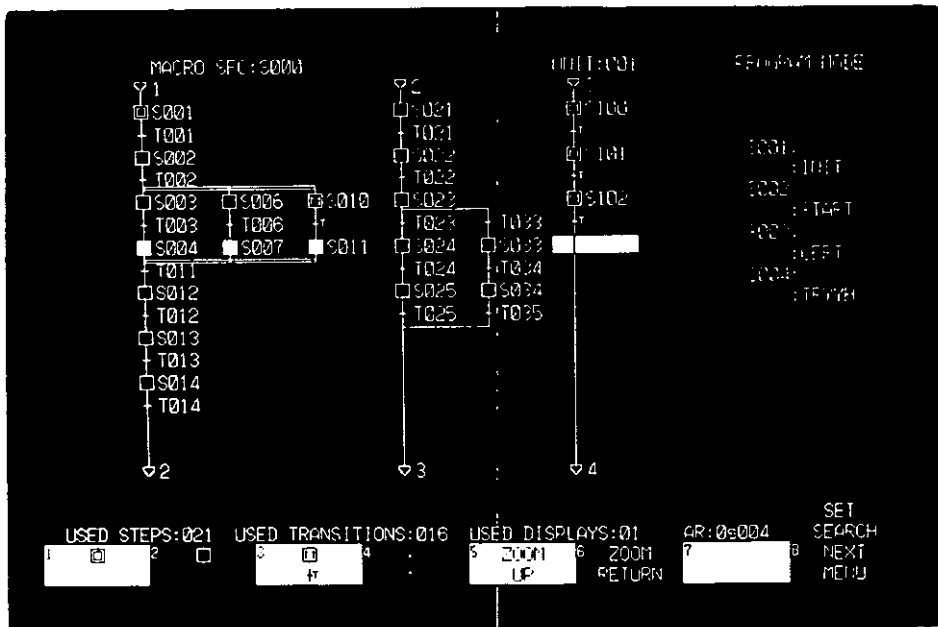
Table 6.1 SFC Program Elements (Cont'd)

Classification	Name	Symbol	Outline	Refer to Page
Link	Simultaneous Convergence		<ul style="list-style-type: none"> ① Converges simultaneous sequences obtained by simultaneous divergence into one process. ② Converges in the upper part of transition. 	310
	Link Line		<ul style="list-style-type: none"> ① Acts as an extension line of link of downward step and transition. 	310
	Loop Output to the Right		<ul style="list-style-type: none"> ① Shows exit to the right of loop when configuring a loop for repetitive control. 	310
	Loop Output to the Left		<ul style="list-style-type: none"> ① Shows exit to the left of loop when configuring a loop for repetitive control. 	310
	Loop Input from the Right		<ul style="list-style-type: none"> ① Shows entrance from the right side of loop when configuring a loop for repetitive control. 	310
	Loop Input from the Left		<ul style="list-style-type: none"> ① Shows entrance from the left side of loop when configuring a loop for repetitive control 	310
	Counter Link Line		<ul style="list-style-type: none"> ① Acts as an extension line of link of step and transition in loop. 	311
	FROM		<ul style="list-style-type: none"> ① Connects to TO connection. ② "X" is the FROM No. 	311
	TO		<ul style="list-style-type: none"> ① Connects with FROM connection. ② "X" is the TO No. A maximum of eight of the same Nos. can be used per screen. 	311
	Macro Entry		<ul style="list-style-type: none"> ① Acts as entrance for macro step expanded views. ② Only one can be used per expanded view. 	311
	Macro Return		<ul style="list-style-type: none"> ① Acts as exit (return to master view) of macro step expanded views. ② A maximum of eight can be used per expanded view. 	311

6.1 OUTLINE OF SFC

"SFC" is an abbreviation for Sequential Function Chart. SFC is a new programmable control language represented by block diagrams which resemble flow charts. Therefore users will be able to see the entire configuration and a series of sequence control systems at a glance. And programming by SFC will be easier than that by only a ladder language. SFC has the following features.

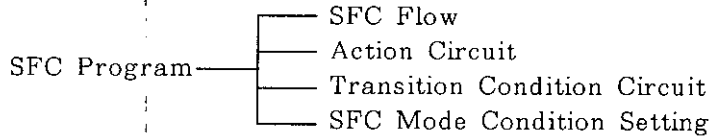
- (1) It is a two-dimensional representation allowing easy visual understanding of the control and processing flow. The conditions needed for sequence progress are clear and present control status can be determined during operation.
- (2) SFC solves only programs that are affected by the present control. It differs from a ladder language in the solving procedures. Therefore the execution time (scan time) can be shortened greatly.
- (3) Program descriptions are very similar to flowcharts, and SFC programs are almost completed by the time the control sequences are built. Therefore, the time needed to create programs can be reduced.



Sample SFC Display

6.2 CONFIGURATION OF SFC PROGRAM

The SFC program has the following configuration.



6.2.1 SFC Flow

The SFC program can contain a maximum of 512 steps and 512 transitions. The flow is stored in units of views and a maximum 64 views can be programmed. The view that will become the foundation is only the first one, and other views are related through macro steps. As shown in Fig. 6.1, a maximum of 64 steps and 64 transitions can be programmed per view. Double use of step and transition Nos. is not possible.

Each view is related through macro steps. Views that contain macro steps are called master views. Views called up by macro steps are called expanded views.

The SFC program always begins with the initial step and is started by converting this initial step to active (corresponding to initialization of the SFC mode). For details, refer to Par. 6.5, "SFC Mode Processing Functions."

NOTE "Active" is the state in which the step is being solved or is being executed. Conversely, "inactive" is the state in which the step is not solved.

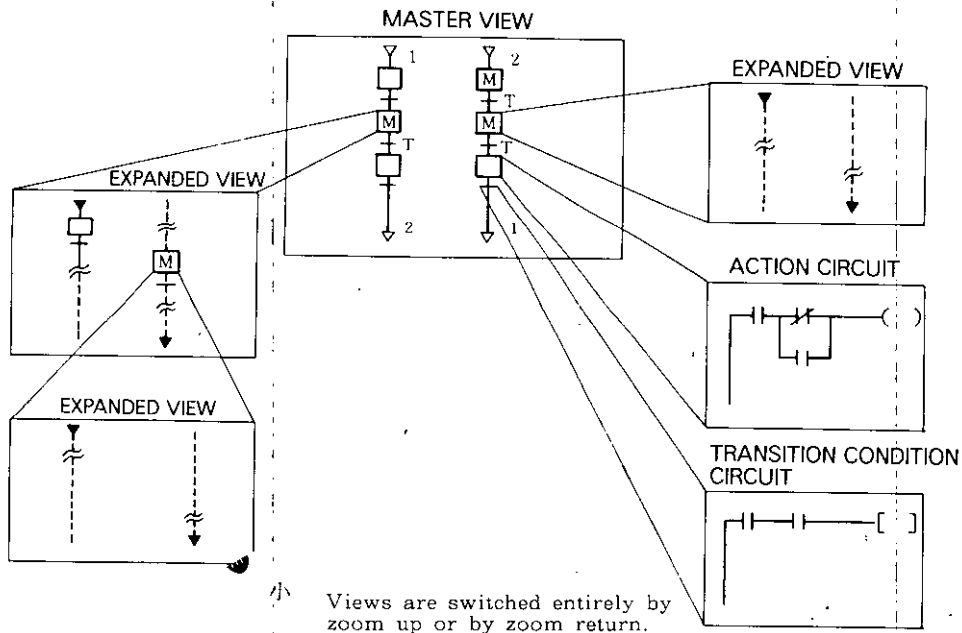


Fig. 6.1 Linking of SFC Views

6.2.2 Action Circuit

An action circuit is a circuit in which special control for step having 1:1 relation with the SFC flow step is described in the ladder language. In one scan, only action circuits of active steps are solved. That is, one unit (one block) of the sequence control is an action circuit.

Therefore, being different from the conventional way using only ladder language, high-speed scanning is possible because an inactive section which does not need solving is not processed in a particular scan. Action circuits are stored in the units of the network, as well as ladder circuits.

For the GL60H or GL70H, it is possible to select whether coils used in an action circuit are forced to be turned off at stepping or if status is held and the next step is advanced (holding mode). This selection is performed by using the dip switch provided in the front of the CPU module. The dip switch setting is performed with the CPU front cover open before the power supply is turned on. After the power supply is turned on, the operation is performed in the set mode. Changing the set mode during the action is ineffective. (Non-holding mode is set at the factory prior to shipping.)

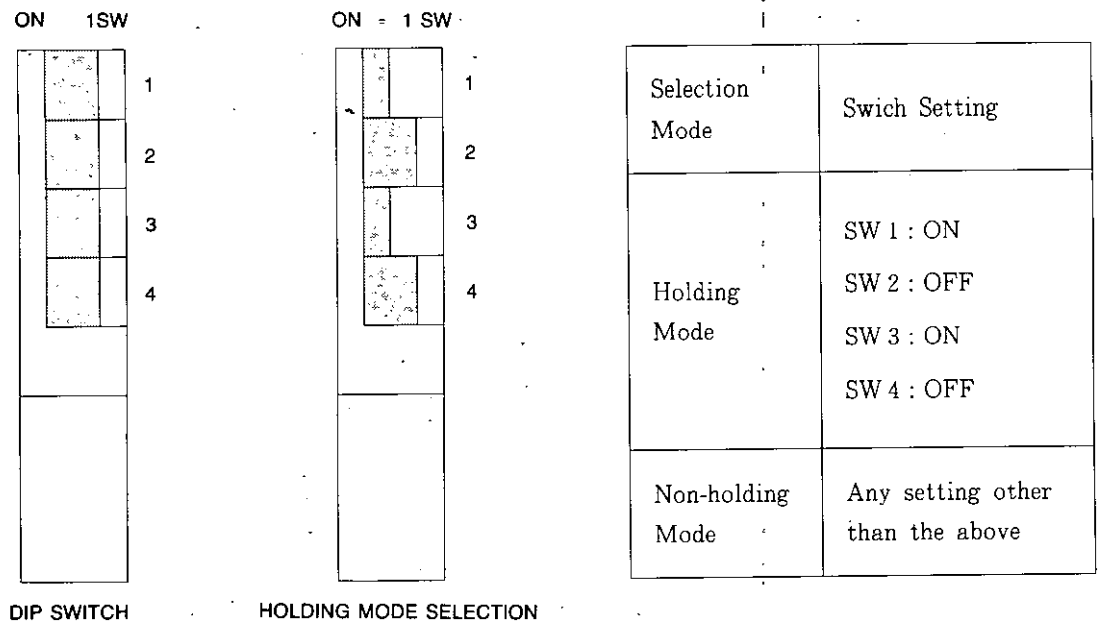


Fig. 6.2 Selection of Non-holding/Holding Modes

In the holding mode, the next step is advanced, holding the coil status immediately before stepping. In the non-holding mode, disregarding the coil status immediately before stepping, the coils are forced to be turned off for the next stepping. The SFC program shown in Fig. 6.3 has a different coil ON/OFF sequence in the holding mode (Fig. 6.4) and the non-holding mode (Fig. 6.5).

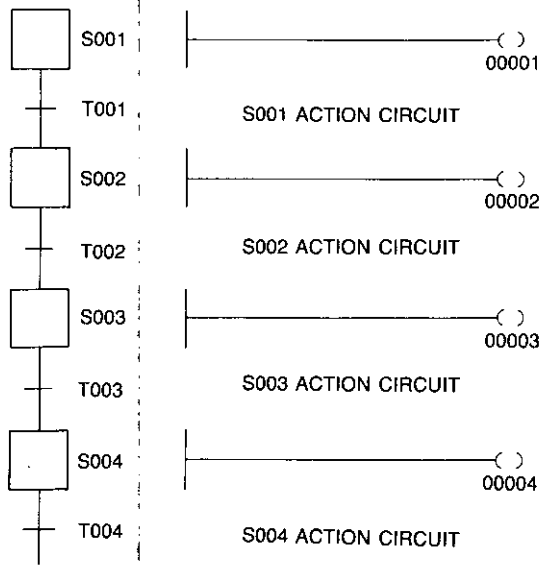


Fig. 6.3 Typical SFC Program

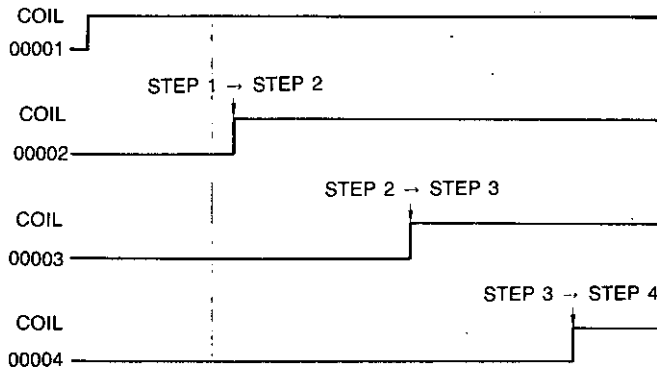


Fig. 6.4 Coil ON/OFF Sequence in Holding Mode

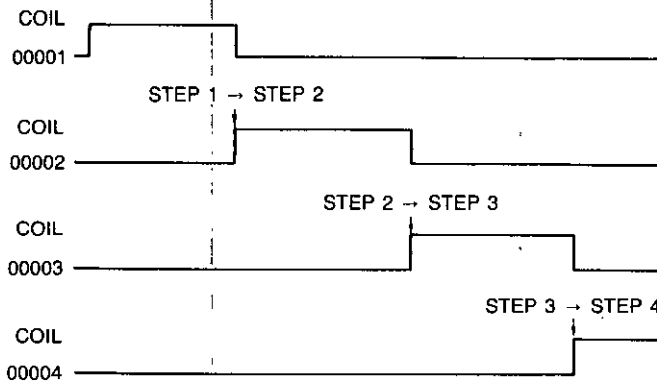


Fig. 6.5 Coil ON/OFF Sequence in Non-holding Mode

6.2.3 Transition Condition Circuit

The transition in SFC flow has its own transition condition circuit. In the circuit, conditions to advance a step to the next step (next control) are represented by a ladder language. The transition condition circuit is composed of only one network and always has one transition coil. Only transition condition circuits that are connected to active steps are solved and are evaluated. Basically, proceeding to the next step occurs during a scan in which the transition coil switches ON.

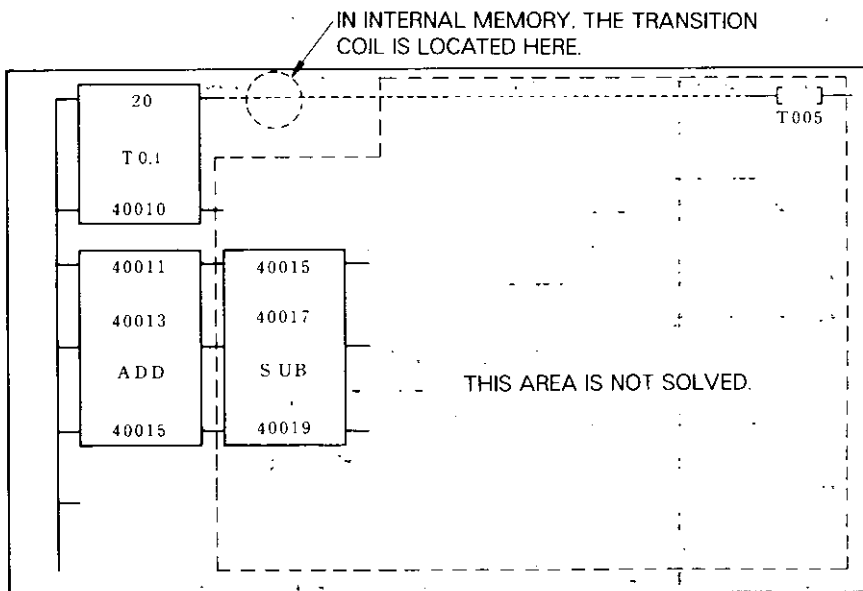
Note that the transition coil has an ON/OFF state only if its transition condition circuit is solved. This is different from ordinary coils. Therefore, the ON/OFF state of the transition coil cannot be referred to by the ladder circuit or by other elements.

NOTE If a counter is used in this circuit, note that its ON/OFF status differs from the ON/OFF status in a general ladder circuit.

The expression "the transition coil is switched ON" in the following means the state in which the transition condition circuit is solved. It does not mean that this state is always maintained.

Solving of the transition condition circuit is to check if the transition coil is switched ON or OFF. Thus, solving of the transition condition circuit is finished when the transition coil is solved. As in ordinary coils, the transition coil is located in No. 11 column on the P150 programming panel display. In the internal memory, it is located in the position where the transition coil is input. Therefore, in the case of the following transition condition circuit, SUB arithmetic operations are not executed.

A horizontal link line must be inserted between the timer output 1 and coil if SUB arithmetic operations are desired to be executed.



6.2.4 SFC Mode Condition Setting

Mode processing forcibly sets steps in an active/inactive state by referring to references such as relays. Mode condition setting is available in the following three types. By switching the input relay or coil to be designated in mode condition setting ON and OFF (OFF to ON and ON to OFF are effective with presetting), the step to be designated becomes active or inactive. For details, refer to Par. 6.5.

Table 6.2 SFC Mode Condition Setting

Name	Mode Function	Mode Controlling Method
Initial Rise	Converts the states of all steps (except initial step) in SFC program to inactive. Only the initial step remains active. (Start and initialization of SFC program)	When condition to be designated by mode condition setting is established.
Reset	Converts active steps into inactive steps.	By the above method and when P150 applies step reset.
Preset	Converts inactive steps into active steps.	Same as in resetting.

NOTE "P150" means the P150 programming panel.

6.2.5 SFC Simulated Operation Functions

In addition to mode condition setting, an SFC program debugging function is available. The following simulated operation functions are offered as an equivalent to the ladder circuit coil disable function. For details, refer to Par. 6.10.

Table 6.3 SFC Simulated Operation Functions

Name	Function of Simulated Operation	Control Method
Disable	Prohibition of evolution	Designated on P150.
Hold	Holding of active state	Designated on P150.

NOTE "P150" means the P150 programming panel.

6.2.6 Set Function, Reset Function

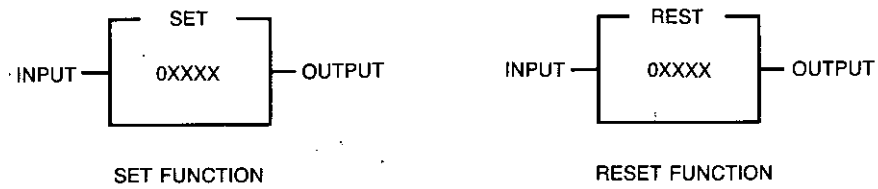
Set and reset functions can be used in the SFC action circuit of the GL60H and GL70H. [Refer to Par. 5.2.1 (11).]

The set function turns on the coil specified by the function if its input is turned off. If the input is turned off, the coil status does not change.

The reset function turns off the coil specified by the function if its input is turned off. Set/reset functions with the same reference number can be used repeatedly. (There is not limitation for overlapping use.)

By using the combination of set/reset functions, coils can be turned on over more than one step, as shown in Fig. 6.6.

Set/reset functions operate disregarding the holding/non-holding mode described in Par. 6.2.2.



INPUT	OUTPUT	"0XXXX" Status
ON	ON	ON
OFF	OFF	No Change

INPUT	OUTPUT	"0XXXX" Status
ON	ON	OFF
OFF	OFF	No Change

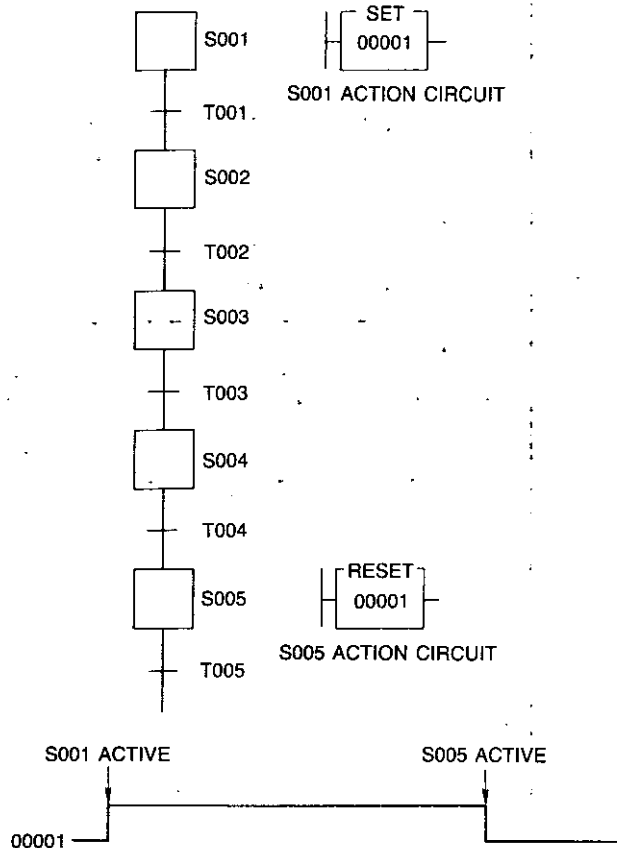


Fig. 6.6 Typical Set/Reset Functions

6.3 SFC PROGRAM ELEMENTS

The program elements that can be used in creating SFC programs are described.

6.3.1 Steps

Step

A step is one sequence unit executed when the given condition is established. As logic states, a step has two states, namely, active (solved state) and inactive (unsolved state). Incidental active circuits are executed and control intrinsic to the steps is performed only in an active state.

If there are no incidental action circuits, processing is performed as a wait step until the next transition condition is established. "XXX" shows the step No., and steps of the same No. cannot be used in the SFC flow more than once. (Prohibition of double use of steps.)

Initial Step

The SFC program always uses an initial step that shows the starting point of the program. The SFC program has only one initial step. All active steps are converted to inactive steps by initialization described in Par. 6.5. The initial step is then forcibly converted into an active step to initialize the SFC program. The initial step is used to raise start of control and in other cases. "XXX" shows the step No.

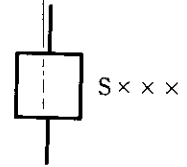


Fig. 6.7 Step Format

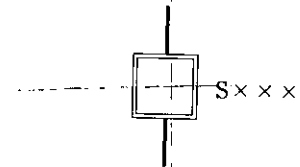


Fig. 6.8 Initial Step Format

Macro Step

In one view, a maximum of only 8 steps and 8 transitions in longitudinal and lateral directions can be programmed. Programming of more steps and transitions requires an expansion of the program area in other views through some elements. Macro steps are used in this function. Views are related through macro steps using the view containing the initial step as an apex. Views that are called by macro steps are called expanded views and views containing macro steps are called master views.

Macro steps are for expansion of views and do not have action circuits. Macro steps are always provided with dummy transitions to be true. "XXX" shows the step No. Figs. 6.10 and 6.11 show the relationship between macro steps and a view and an equivalent SFC flow.

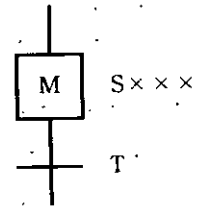


Fig. 6.9 Macro Step Format

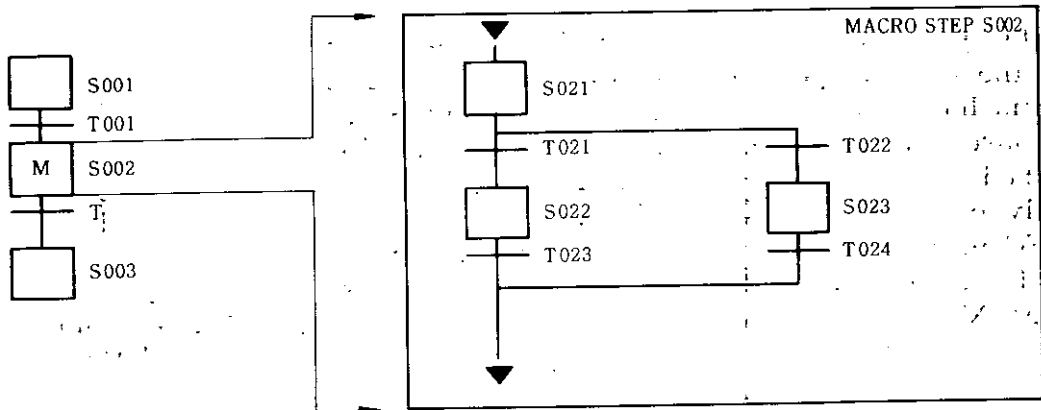


Fig. 6.10 Relationship Between Macro Steps and View

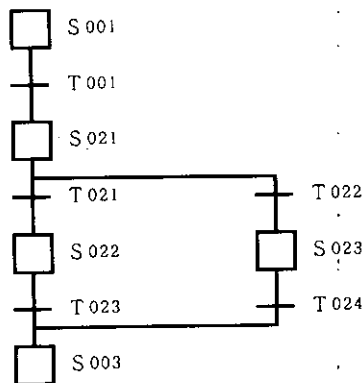


Fig. 6.11 SFC Flow View Equivalent to Fig. 6.10

6.3.2 Transition

Transition

Transitions show execution process transition conditions between adjoining steps. Steps are made to proceed assuming that the conditions have been established if the transition coil in the transition condition circuit is switched on. Only the transition condition circuits of the transitions that are linked to active steps are solved. As in steps, transitions of the same Nos. cannot be used more than once. "XXX" shows the transition No.

Counter Transition

Counter transitions represent loops together with loop links. "XXX" shows the transition No. Counter transitions are always used with loops to control loop cycles and other elements.

Dummy Transition

Dummy transitions are used only directly below macro steps and are stored automatically when macro steps are stored during program creation. A transition is always used with macro steps in the exit of an expanded view, and an additional transition does not have to be provided directly below the macro step. Therefore, dummy transitions do not have transition condition circuits.

6.3.3 Link

The following links are available for connection of steps. Various control programs can be built by suitably combining these links which are roughly divided into three types. One is BRANCH to provide a variation for proceeding to destinations including divergence, simultaneous divergence, convergence, simultaneous convergence and link lines. The second type is LOOP and includes loop input from the right and from the left, loop output to the right and to the left, and counter link lines to build loop connections. The third type is CONNECTOR that includes FROM, TO, macro entry and macro return to connect No. 8 and 1 lines, as well as macro steps and expanded views.

Links	{	BRANCH (divergence, simultaneous divergence, convergence, simultaneous convergence and link line)
		LOOP (loop input from right and from left, loop output to the right and to the left, and counter link lines)
		CONNECTOR (FROM, TO, macro entry, and macro return)

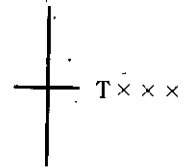


Fig. 6.12 Transition Format

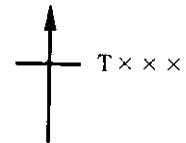


Fig. 6.13 Counter Transition Format

Divergence

Divergence corresponds to decision statements in flowcharts. First, it checks the condition of transition condition circuit. Then, it continues a condition check of the circuit, moving from left to right in turn, beginning with the nearest circuit on the right. The step under transition whose condition is established is processed. Divergence is used when selecting and controlling one of several processes. Transitions must be located directly below divergence link lines.

Convergence

Process systems branched in divergence must converge from one system in order to repetitively execute SFC flow in the SFC program. Otherwise, return to the top of control is sometimes not possible. (Returning from several TOs to one FROM is also possible.) This must be kept in mind when creating programs. It is well to remember that divergence and convergence are used in pairs.

Simultaneous Divergence

This is the link used to simultaneously perform several processes, namely, when wishing to create simultaneous sequence programs.

The number of steps that can be made to be active simultaneously is limited only by the numbers of steps and views that are allowed by using macro steps under simultaneous divergence. The steps must be located directly below links of simultaneous divergence without using link lines.

Simultaneous Convergence

The processing system which diverges in parallel by simultaneous divergence must be merged again by simultaneous convergence as in the relationship between divergence and convergence. Unless this 1-to-1 relationship is observed, the control system will become chaotic. This will be described in more detail in Par. 6.6.

Link Line

Link Lines are used as links of downward steps and transitions.

Loop

Loops are used when wishing to repeat some processes. Functions as loops are accomplished if transitions directly under steps are not established and if counter transitions are established. Loops are divided into a loop (left) and loop (right) and can be selected in accordance with the use condition of the views.

If repeating only one step is required using a loop, the processing will be the same as that of the transition next to the step not established and using no loop. However, there is only one difference, that the timer register of the step is cleared.

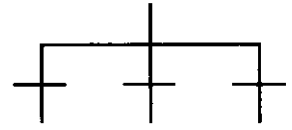


Fig. 6.14 Divergence Format

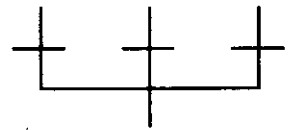


Fig. 6.15 Convergence Format

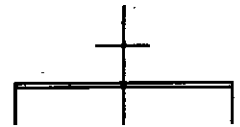


Fig. 6.16 Simultaneous Divergence Format

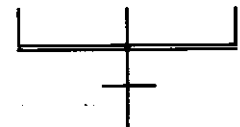


Fig. 6.17 Simultaneous Convergence Format



Fig. 6.18 Link Line Format

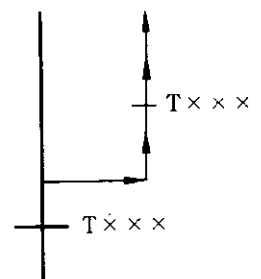


Fig. 6.19 Loop Format

"T X X X" shows the transition No. Loop output to the right must end in loop input from the right and loop output to the left must end in loop input from the left.

Counter Link Line

Counter link lines are used as directed lines of loop steps and counter transitions.

FROM and TO

One view can contain only a maximum of 8 lines of steps, and FROM and TO are connection elements that allow the use of wide views. FROM and TO elements of the same Nos. have logical connection relations. Connection elements are effective only in the same views and cannot be used when spread over two views. A maximum of eight TOs of the same No. can be used in one view. However, only one FROM of the same No. can be used. "X" shows the FROM-TO connection No. and a numeral from 1 to 8 can be used. Fig. 6.22 shows an example of use of FROM-TO.

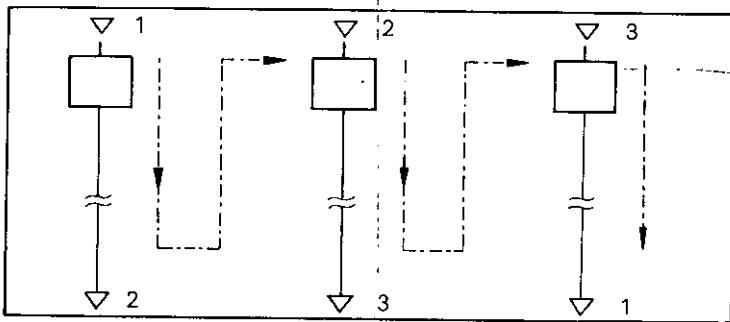


Fig. 6.22 Example of Use of FROM-TO

Macro Entry and Macro Return

Macro entry and macro return are elements that are used inside expanded views defined by macro steps. Macro entry means the program start point of expanded views. Steps connected to macro entry become active when the applicable macro step becomes active. Macro return means returning to the master view if the transition conditions and steps connected to it are active, proceeding to the next step. In other words, it is an element that indicates the evolution of macro steps.

One view has only one macro entry. (One macro entrance.) Macro returns only return to the master view, and a maximum of eight macro returns can be used.



Fig. 6.20 Counter Link Line Format

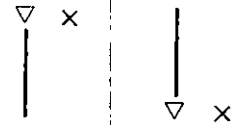


Fig. 6.21 FROM-TO Format

NOTE ----- shows flow of the evolution.

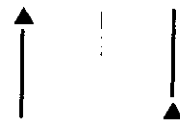


Fig. 6.23 Format of Macro Entry and Macro Return

6.3.3 Link (Cont'd)

Fig. 6.24 shows an example of macro entry and macro return usage.

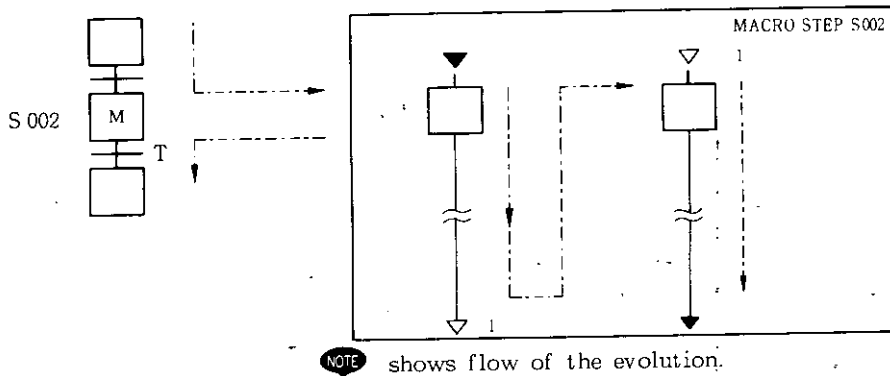


Fig. 6.24 Example of Usage of Macro Entry and Macro Return

6.4 STEP EVOLUTION RULE

This section describes the basic step evolution methods in executing the SFC programs executed with the individual links. Refer to Par. 6.14 for the method of evolution if programs are incorrect in syntax or if operation is incorrect.

6.4.1 Evolution Condition

The conditions which cause evolution are listed below.

- (1) Steps that are presently active are not in hold state of simulated operation function.
- (2) The preceding step is connected to the corresponding transition and its transition condition circuit is created.
- (3) The transition coil of the transition condition circuit connected below is switched on.
- (4) Steps at the destination exist.
- (5) All the steps at the destination are inactive.
- (6) None of the steps at the destination are in a disable state of the simulated operation function.

Evolution occurs only if all these conditions are established.

Evolution of the same step occurs only once in one scan.

In solving evolution, the transition connected next to the step, which was active in the previous scan, is solved in one scan. If all the above evolution conditions are established, action circuits incidental to the steps at the destination are solved. Conversely, if the evolution conditions are not established, the action circuits of the step which is presently active are solved. Therefore, action circuits of steps before and after evolution are not solved in one scan.

Macro steps can be used in expanded views of macro steps (nesting) freely. The maximum number of macro steps used as an intervention to proceed from one step to another is 32.

6.4.2 Most Simple Connection

The most simple connection shown in Fig. 6.25 which has only a transition between steps is described below.

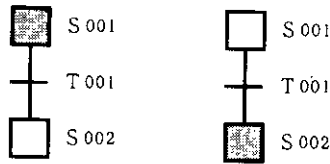


Fig. 6.25 Evolution in the Most Simple Connection

If the transition coil in the transition condition circuit of T001 switches ON after the scan next to the scan in which Step S001 became active, S001 becomes inactive and S002 becomes active. (Evolution takes place.) The coil of the action circuit of the step before evolution is OFF when evolution takes place, and the action circuit of the proceeded step is solved. The timer register of the step before evolution is cleared once during evolution (when becoming active) and is counted up until the next evolution (when becoming active). The timer register retains the present value until it becomes active the next time, if the step changes from active to inactive.

A simple circuit as shown in Fig. 6.26 is taken into consideration as a transition condition circuit of T001. In this example, the step which is active during a scan in 1 sec after S001 became active changes from S001 to S002. The value of the timer register of S001 at this time will be 10. (The timer unit of the timer register is 100 msec.) The timer register can be used for monitoring step dwell time and for other purposes.



NOTE The evolution does not take place even if the transition coil switches ON if S001 is held or if S002 is disabled by the SFC simulated operation function.

Fig. 6.26 Example of Transition Condition Circuit T001

6.4.3 Divergence Connection

The method of evolution steps if steps are connected by divergence is described.

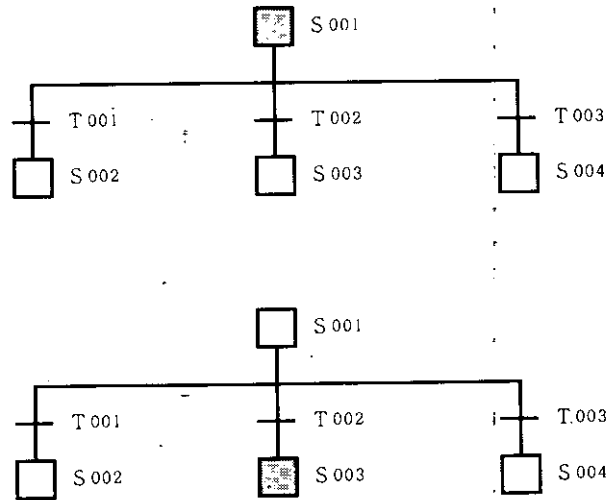


Fig. 6.27 Evolution in Divergence Connection

In Fig. 6.27, the system checks the transition condition circuit in the order of T001, T002 and T003 for each scan after the scan next to the scan when Step S001 became active. (Principle of priority to the left.) The sequence proceeds to the step below the transition during the scan if there is a transition in which the transition coil of the transition condition circuit is ON. The above example shows the transition state if the T001 transition coil is OFF and if the T002 transition coil is on in one scan. The sequence does not proceed to S004 even if the T003 transition coil is ON at this time. In program creation, high priorities in divergence must be placed sequentially from left to right.

NOTE The sequence does not proceed even if the T001 transition coil switches ON if S002 is disabled in the SFC simulated operation function. S001 remains held until the transition coil of T002 or T003 switches ON.

6.4.4 Simultaneous Divergence Connection

The method of evolution steps if steps are connected by simultaneous divergence is described.

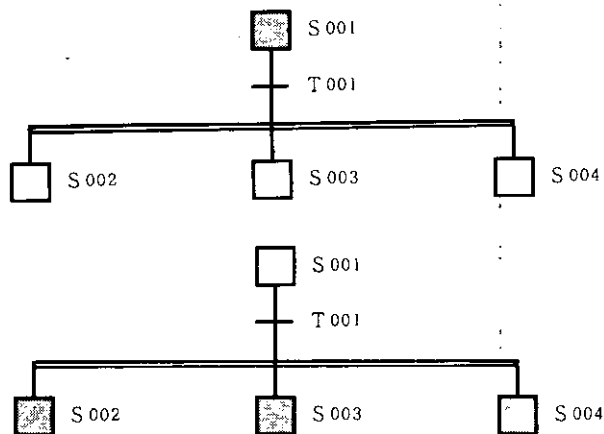


Fig. 6.28 Evolution in Simultaneous Divergence Connection

Active steps change from S001 to three steps, namely, to S002, S003 and S004 in Fig. 6.23, if the T001 transition coil switches ON after the scan next to the scan in which Step S001 became active. Simultaneous divergence means starting a simultaneous sequence and Steps S002, S003 and S004 are sequences which are independent of each other. The steps separated by simultaneous divergence must be collected ultimately to one sequence by simultaneous convergence. Refer to Par. 6.4.5.

The number of steps that can be diverged simultaneously in one scan from one step is 128 (excluding the number of macro steps; the maximum number of macro steps that can be used as an intervention is 32).

6.4.5 Simultaneous Convergence Connection

The method of evolution steps if steps are connected by simultaneous convergence is described below.

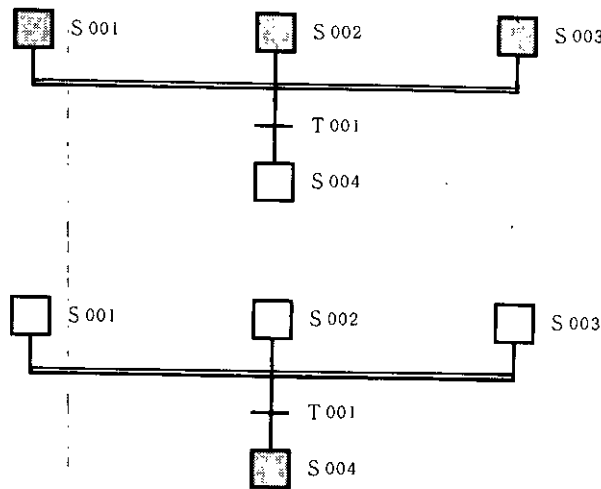


Fig. 6.29 Evolution in Simultaneous Convergence Connection

In Fig. 6.29, the active steps change from S001, S002 and S003 to S004 if the T001 transition coil switches ON after the scan next to the scan during which Steps S001, S002 and S003 all became active. Simultaneous convergence means an end of the simultaneous sequence started by simultaneous divergence.

Let us assume that only S001 is active in Fig.6.29. At this time, the condition for simultaneous convergence evolution "All steps connected in conjunction with simultaneous convergence must be active" is not met. Therefore, the T001 transition condition circuit is not solved. Step S001 continues to be held until Steps S002 and S003 become active. Therefore, care must be exercised not to create a program that will cause a mismatch in the numbers of steps which are simultaneously diverged and are finally simultaneously converged when creating simultaneous divergence and simultaneous convergence programs. Refer to Par. 6.6 for the details.

6.4.6 Loop Connection

The method of evolution steps if steps are connected by loops (left and right) is described.

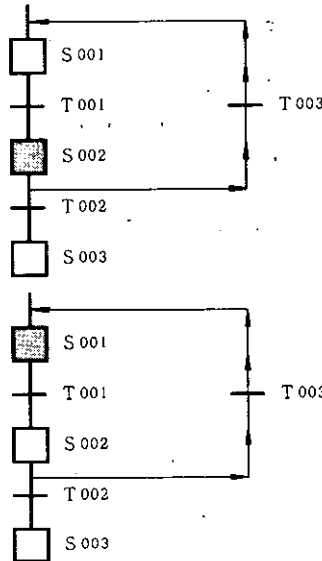


Fig. 6.30 Evolution in Loop Connection

The active step changes from S002 to S001 if the transition coil of T002 switches OFF and if the transition coil of Counter Transition T003 switches ON in Fig. 6.30 after the scan next to the scan in which Step S002 became active. Loop connection acts only as a loop if the transition coil of the transition below switches OFF and if the transition coil of the counter transition is ON. Evolution becomes as shown in Table 6.4 in other states.

Table 6.4 Evolution by Transition State

Transition		T003	
		ON	OFF
T002	ON	S003	S003
	OFF	S001	S002

6.4.7 Macro Step Evolution

The method of macro step evolution is described below. Views can be used effectively by skillfully using macro steps.

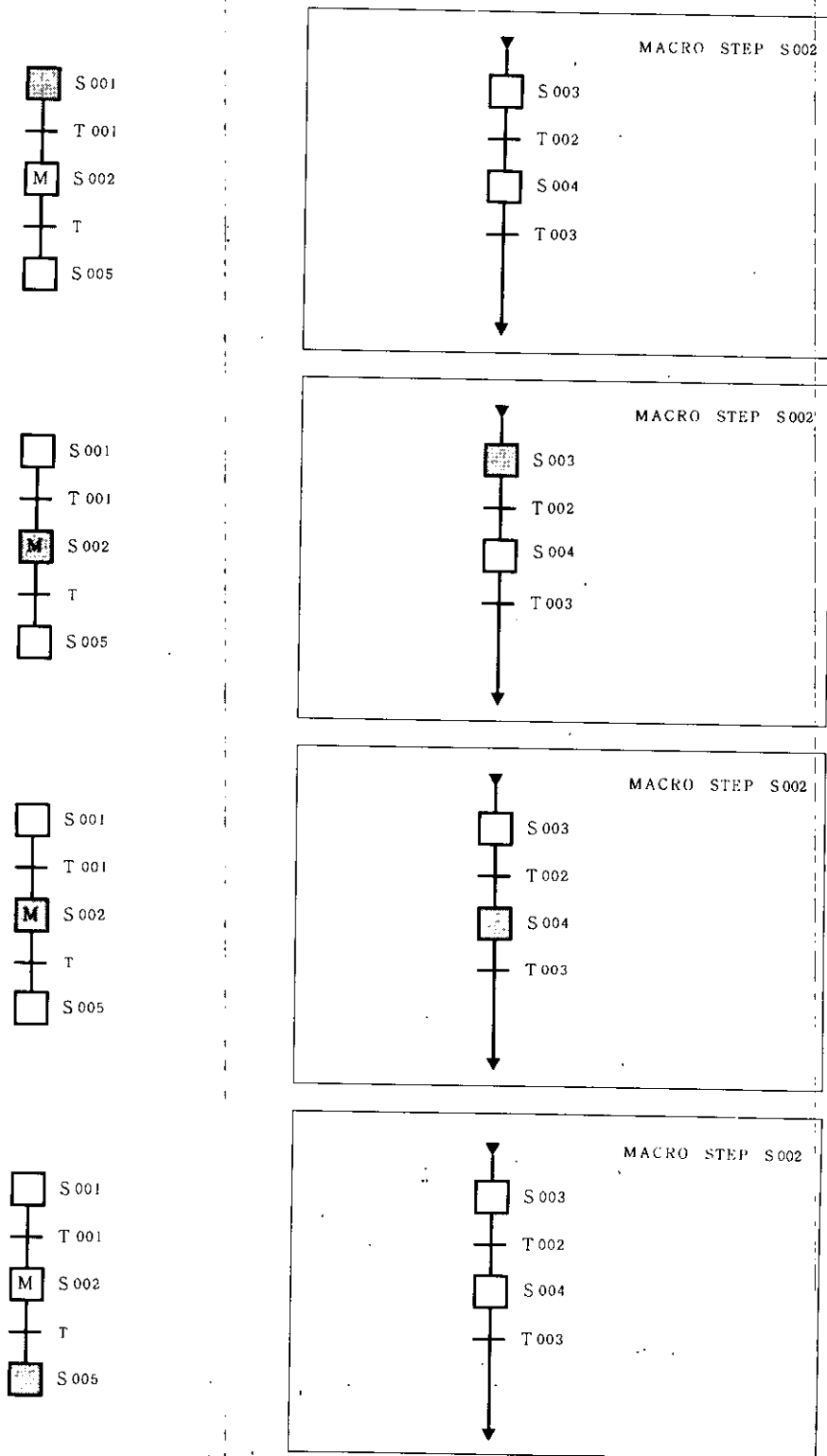


Fig. 6.31 Macro Step Evolution

The active step changes from S001 to Macro Step S002 in Fig. 6.31 if the T001 transition coil switches ON after the scan next to the scan during which Step S001 became active. Macro steps have no action circuits and merely connect master and expanded views. Thus, only Step S003 in expanded view actually becomes active.

If S004 is active and the T003 transition coil is ON, the active step shifts from S004 to Master View S005 after passing through a macro return. Active steps disappear from expanded views and macro steps change from active to inactive.

If active steps do not exist because macro step expanded views are not yet created or for other reasons, the active step shifts from S001 to S002 in case S001 is active and the T001 transition coil is ON. In the next scan, the active step proceeds from S002 to S005 as the dummy transition does not have a transition condition circuit. A macro step without expanded views acts as a wait step for one scan.

6.5 SFC MODE PROCESSING FUNCTION

Three functions are available as SFC mode processing functions, namely, initialization, resetting and presetting. Initialization must be performed before executing an SFC program. Resetting and presetting are mainly used for debugging. Fig. 6.32 shows the flow of mode processing which represents the priority order of mode processing.

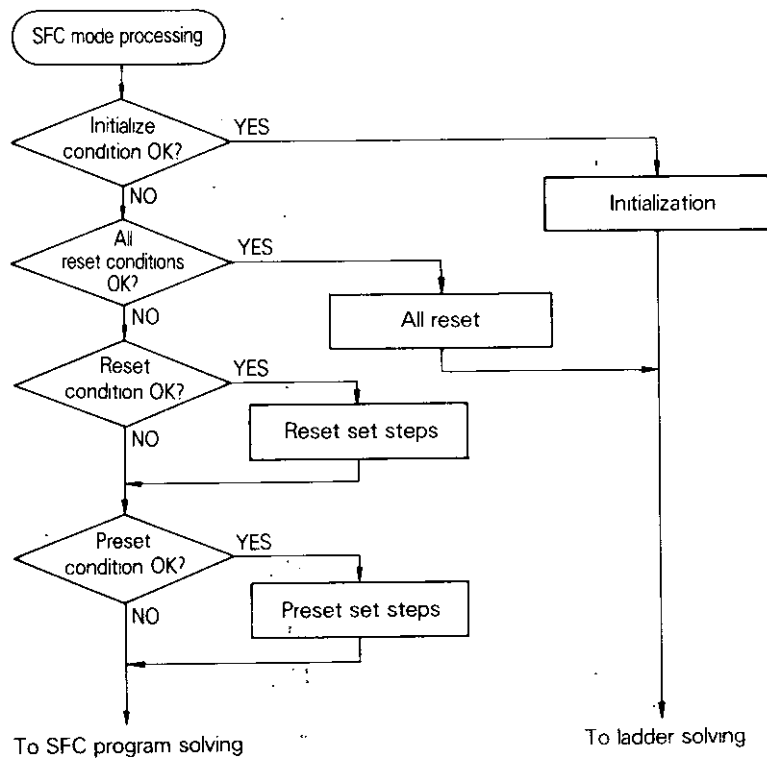


Fig. 6.32 SFC Mode Processing Flow

As Fig. 6.32 shows, presetting is performed after resetting is performed if both resetting and presetting are set for the same step and if establishment conditions for both of them are met in one scan. An active state is therefore established.

6.5.1 Initialization Function

After the conditions designated by mode condition setting of the initialization function is established, all the active steps inside the SFC program are inactivated, leaving only the initial step to be active. The initialization function initializes and starts the SFC program.

Mode condition setting of the initialization function consists of a reference No. and initialize establishment condition. Input relay 1×××× or coil 0×××× can be designated as the reference No. ON or OFF can be designated for the initialize establishment condition.

Fig. 6.33 shows an example of mode condition setting.

INITIALIZE		UNIT:XXX PROGRAM MODE
NO.	REF	STATUS
1	10001	ON

Fig. 6.33 Example of Mode Condition Setting of Initialization Function

In the case of the example in Fig. 6.33, after all the active steps within the SFC program become inactive during the scan in which Input Relay 10001 switches ON for the first time following mode condition setting, only the initial step becomes active, and its action circuit is solved. If the steps become inactive, all the coils in the action circuit switch OFF and the timer registers of all the steps are cleared. The holding register retains the value immediately before initialization.

The initialization not only is established by the rise and fall of reference, but also by each scan while an ON or OFF state is retained. Unless the state is reversed, the condition is initialized during each scan.

The transition below the initial step is not checked during a scan in which initialization is processed. Transition does not occur during this scan.

Refer to the P150 Programming Panel User's Manual SFC Information (SIE-C815-14.3) for the condition setting method in detail.

NOTE Be sure to initialize or perform all resetting if a program is loaded. The program may not operate correctly if the state remains as it is.

6.5.2 Presetting Function

The following two presetting function methods are available.

- (1) Steps can be forcibly made to be active on the edit view of the P150 programming panel or by listing the ON/OFF states of the steps. This function mainly becomes effective when executing the SFC program during debugging.
- (2) As in the initialization function, steps designated based on mode condition setting of the presetting function can be made to be active as explained in details below.

If the conditions set during presetting function mode condition setting are established, a maximum of eight steps set to the conditions become active simultaneously and their action circuit is solved. A maximum of eight steps can be set to a set of reference Nos. and preset establishment conditions in mode condition setting of the presetting function. A maximum of 64 sets of this combination can be set. Input Relay 1×××× of Coil 0×××× can be designated as a reference No. On or OFF can be designated as the preset establishment condition. Step No. to be preset can be designated not only directly as step No., but also indirectly by designating Input Register 3×××× and by using the data of this input register as the step No. Direct and indirect designation can be selected in accordance with the circumstances.

Fig. 6.34 shows an example of mode condition setting.
(The diagram is slightly different from the actual view.)

PRESET			UNIT. XXX	PROGRAM	MODE
NO.	REF #	STATUS	STEP	NO/REGISTER	NO
1	10001	ON	S002	S004	
2	00003	OFF	S005	30001	
8 STEPS MAX					

64 TYPES MAX

Fig. 6.34 Example of Mode Condition Setting in Presetting Function

In the case of the example shown in Fig. 6.34, the two steps, S002 and S004, become active during the scan in which Input Relay 10001 changes from OFF to ON for the first time after mode condition setting. Their action circuits are solved. During the scan in which Coil 00003 changes from ON to OFF for the first time, S005 and the step shown by the data of Input Register 30001 become active, and their action circuits are solved.

For example, if the data of Input Register 30001 is 15, Step S015 becomes active.

In presetting, reference rise and fall are provided to prevent an increase in the number of active steps, and presetting is performed only once without reversing the reference condition.

The active steps remain active even if they are preset.

NOTE

1. By presetting a macro step itself, all the steps of the corresponding expanded view do not become active and proceed to the steps connected below during the next scan.
2. By presetting steps inside expanded views of a macro step, the macro step does not become an active display, but operates as usual.

6.5.3 Resetting Function

The following three methods are available for the resetting function.

- (1) Steps can be forcibly made inactive on the panel by operating the edit view of the P150 programming panel or by listing the ON/OFF states of the steps. This function becomes effective mainly when wishing to change an active step in the SFC program into an inactive state during debugging.
- (2) Unlike the presetting function, resetting can change all active steps into inactive steps as explained below.
- (3) As in the initialization function, the steps designated based on the mode condition setting of the resetting function can be changed from active to inactive as explained below.

All Resetting

If the mode conditions set by the resetting function are established, all the programmed steps are changed to inactive. All the coils in the action circuit are switched OFF if the steps are inactivated. The data in the timer register are cleared. However, the data of the holding register and other registers used in the action circuit and transition condition circuit remain. Fig. 6.35 shows an example of all resetting being performed during the scan in which Input Relay 10002 switches ON. All the steps are inactivated and the steps are not designated.

Resetting Function

If the mode condition set by the resetting function is established, a maximum of eight active steps set to this condition inactive become simultaneously and their action circuits are cleared. Mode allocation of the resetting function can set a maximum of eight steps per set of reference Nos. and reset establishment condition, and a maximum of 64 sets of this combination can be set. Input Relay 1×××× or Coil 0×××× can be designated as reference No. ON or OFF can be designated as the reset establishment condition. Step No. to be reset can be designated not only directly as step No., but also indirectly by designating Input Register 3×××× and by using the data of this input register as the step No. Direct and indirect designation can be selected in accordance with the circumstances.

Fig. 6.35 shows an example of mode condition setting. (The diagram is slightly different from the actual view.)

RESET			UNIT	XXX	PROGRAM	MODE
NO.	REF#	STATUS	STEP NO./REGISTER		NO	
ALL	10002	ON				
1	10001	ON	S002	S004		
2	00003	OFF	S005	30001		
8 STEPS MAX						

64 TYPES MAX

Fig. 6.35 Example of Mode Condition Setting in Resetting Function

In the case of the example shown in Fig. 6.35, the two steps, S002 and S004, change from active to inactive during the next scan after Input Relay 10001 changes from OFF to ON for the first time following mode condition setting. Their action circuits are cleared. S005 and the step shown by the data of Input Register 30001 change from active to inactive and the action circuits are cleared during the scan after Coil 00003 changes from ON to OFF for the first time.

For example, Step S015 changes from active to inactive if the data value of Input Register 30001 is 15.

In resetting, only the reference state is scanned, and resetting is performed every scan if the reference state is not reversed.

If resetting is performed with an inactive step, no processing is performed.

NOTE

1. Active steps inside expanded views cannot be reset even if a macro step is reset (except for all resetting).
2. Macro step display remains that for active even if steps inside the macro step expanded view are reset (except for all resetting), but normal operation is performed.
3. Steps become an active state if preset and reset is established for the same step during one scan.

6.5.4 Precautions for Presetting and Resetting

Precautions for presetting and resetting processing are described below.

Presetting and Resetting inside Simultaneous Divergence and Simultaneous Convergence

The number of active steps contradicts (as mentioned in Par. 6.9) unless all the steps that become active by simultaneous divergence are reset and all the steps that are supposed to become active by simultaneous divergence are preset when presetting and resetting inside simultaneous divergence and simultaneous convergence.

Solving Sequence of Active Steps Created by Presetting

Active steps created by presetting are solved after solving of active steps created by normal processing, instead of by presetting, that is, at the end of SFC solving. If several steps are preset, steps that are preset first are solved first. Refer to Par. 6.9 for the details.

6.6 SFC PROGRAMMING PROHIBITED ITEMS

This section describes expressions that are not allowed in SFC programming. These expressions are not allowed either because they have a logic contradiction that may disable evolution or because of limitations in SFC program processing.

The prohibited items are divided into two groups. The first group does not allow input on the P150 programming panel. The second group allows input, but performs abnormal evolution during execution. Therefore, the prohibited items, particularly of the second group, must be checked before creating the SFC programs.

6.6.1 SFC Storage Prohibited Item

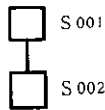
The following SFC flow examples are the connections that cannot be stored and cannot actually be stored.

Direct Connection

(1) Direct Connection between Steps

Conditions for shifting are always needed to enable shifting from one control to another. Therefore, direct connection between steps is not allowed.

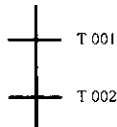
Example:



(2) Direct Connection between Transitions

If two conditions are needed for shifting from one control to another, ANDing inside one transition condition circuit is needed, and direct connection between transitions becomes unnecessary.

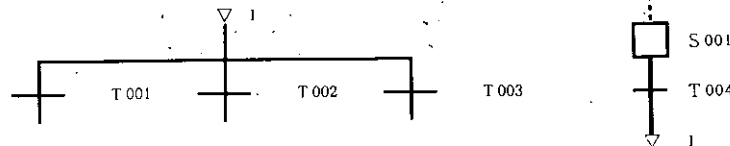
Example:



(3) Direct Connection between FROM and Divergence - Simultaneous Convergence - Loop

Links for divergence, simultaneous convergence and loop always require transitions below them. FROM always requires transition above it (above TO). Thus, direct connection of them is included in the prohibition items in (2) and is prohibited.

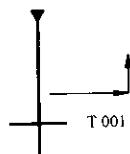
Example:



(4) Direct Connection between Macro Entry and Divergence - Simultaneous Convergence - Loop

Macro entries too require transitions above them (above macro steps). Therefore, direct connection of them is included in the prohibition items of (2) and is prohibited.

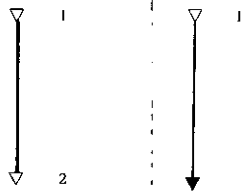
Example:



(5) Direct Connection between FROM and TO-Macro Return

This is a meaningless connection and is prohibited.

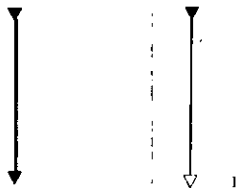
Example:



(6) Direct Connection between Macro Entry and TO-Macro Return

This is a meaningless connection and is prohibited.

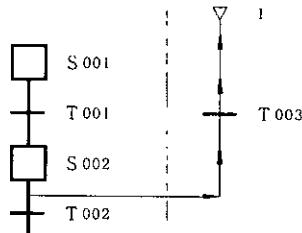
Example:



(7) Direct Connection between FROM-Macro Entry and Counter Link Line-Counter Transition

The counter link line-counter transition condition is used only in loop connections. Therefore, it cannot be connected directly to FROM-macro entry.

Example:

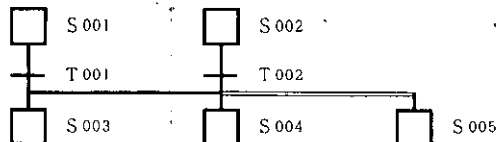


Divergence, Simultaneous Divergence, Convergence and Simultaneous Convergence

(1) Parallel Use of Convergence and Simultaneous Divergence

Evolution is bound to have two meanings and is prohibited.

Example:

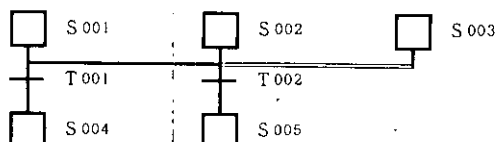


If S002 becomes active, identification whether the destination is S003 of convergence or S004 and S005 of simultaneous divergence.

(2) Parallel Use of Divergence and Simultaneous Convergence

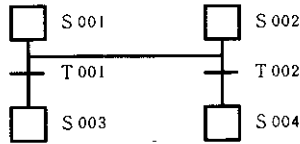
Evolution is bound to have two meanings and is prohibited.

Example:



(3) Connecting more than Two Steps above Divergence.

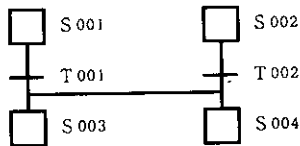
Example:



If S002 also becomes active when the sequence proceeds from S001 to S003, the destination of S002 is limited.

(4) Connection of more than Two Steps below Convergence

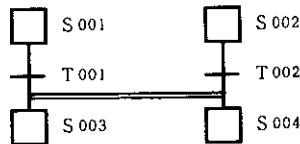
Example:



If S001 becomes active, two steps, S003 and S004, can proceed and identification will no longer be possible.

(5) Connection of more than Two Steps above Simultaneous Divergence

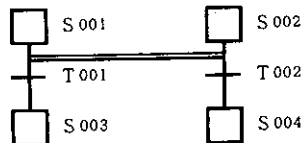
Example:



If S002 also becomes active while the sequence proceeds from S001 to S003 and to S004, the destination of S002 becomes active, disabling evolution.

(6) Connection of more than Two Steps below Simultaneous Convergence

Example:

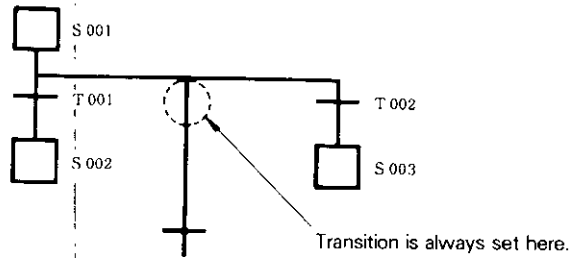


If S001 and S002 are active, the destination is affected by the states of T001 and T002. In the CPU, the destination of simultaneous convergence is always one unless there is connection for simultaneous divergence below.

(7) Connection of Link Line between Divergence and Transition

This connection is prohibited because of GL60H, GL70H processing. This must not cause any limitation to program creation.

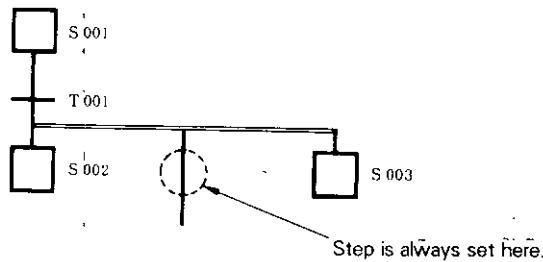
Example:



(8) Connection of Link Line between Simultaneous Divergence and Step

This connection is prohibited because of GL60H, GL70H processing. This must not cause any limitation to program creation.

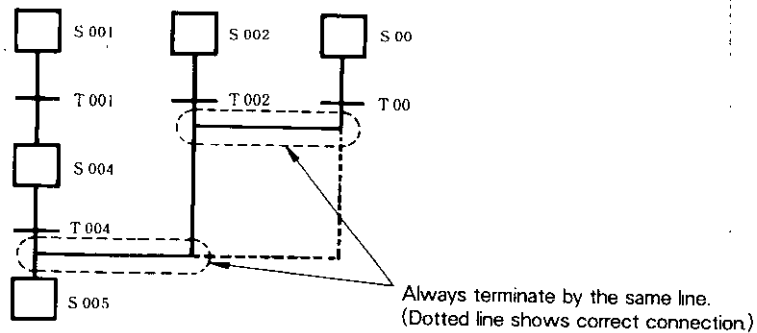
Example:



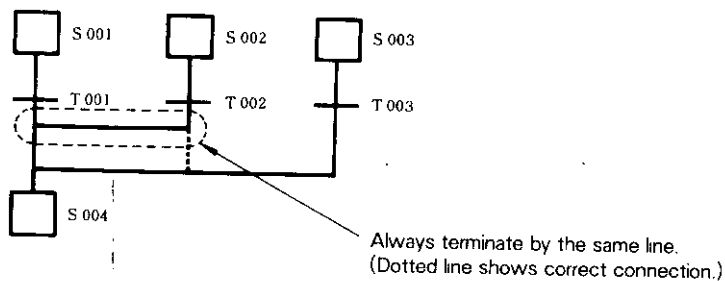
(9) Double Convergence

Convergence cannot be connected using a double line. Always converge using the same line.

Example 1:



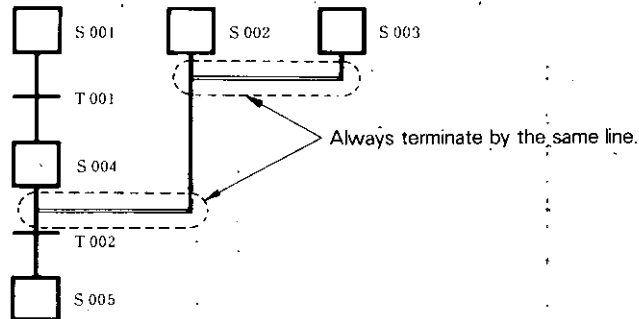
Example 2:



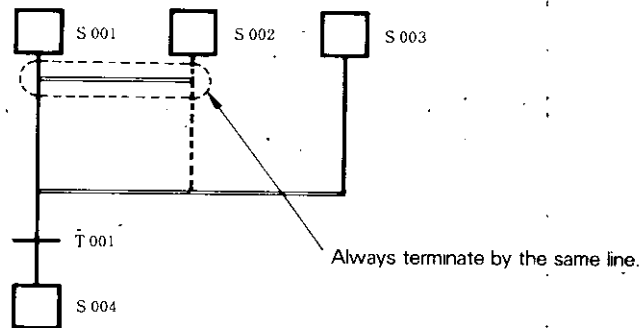
(10) Double Simultaneous Convergence

Simultaneous convergence cannot be connected by a double line. Always perform simultaneous convergence by the same line.

Example 1:



Example 2:

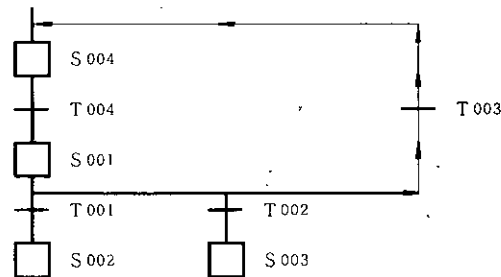


Loop

(1) Connecting Loop to Divergence, Simultaneous Divergence, Convergence and Simultaneous Convergence

Connection of a loop to divergence, simultaneous divergence, convergence and simultaneous divergence, convergence and simultaneous convergence is prohibited as the destination cannot be identified.

Example:

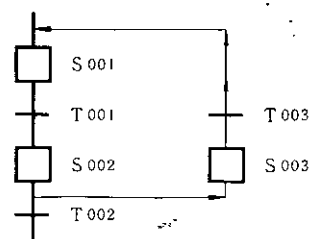


If S001 becomes active, the destination can no longer be identified, S002 or S003 below divergence, or S004 at the loop destination.

(2) Use of Elements other than Loop Elements inside a Loop

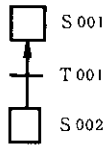
Loop elements: Loop output to the right and to the left, loop input from the right and from the left, counter link line, and counter transition.

Example:



(3) Use of Loop Elements Outside a Loop

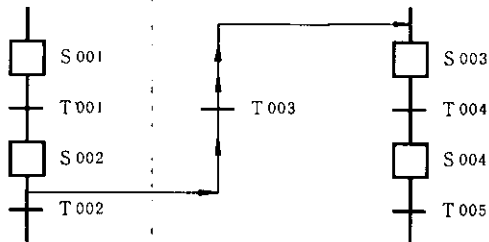
Example:



(4) Mismatch between the Left and the Right of a Loop

Loop output to the right must end with loop input from the right and loop output to the left must end with loop input from the left.

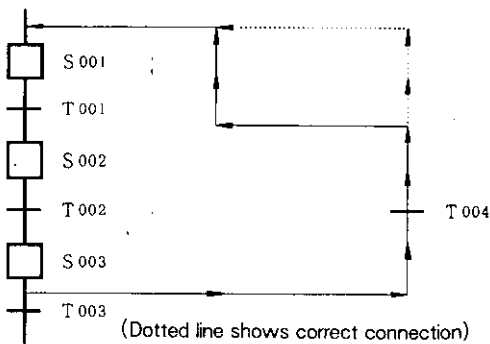
Example:



(5) Loop Parallel Input and Output

A loop must have parallel input and output.

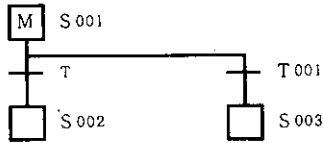
Example:



Macros

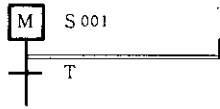
A link cannot be inserted between a macro step and dummy transition.

Example:



The sequence unconditionally proceeds to S002 even if the T001 transition coil is ON when a step proceeds from an expanded view of Micro Step S001.

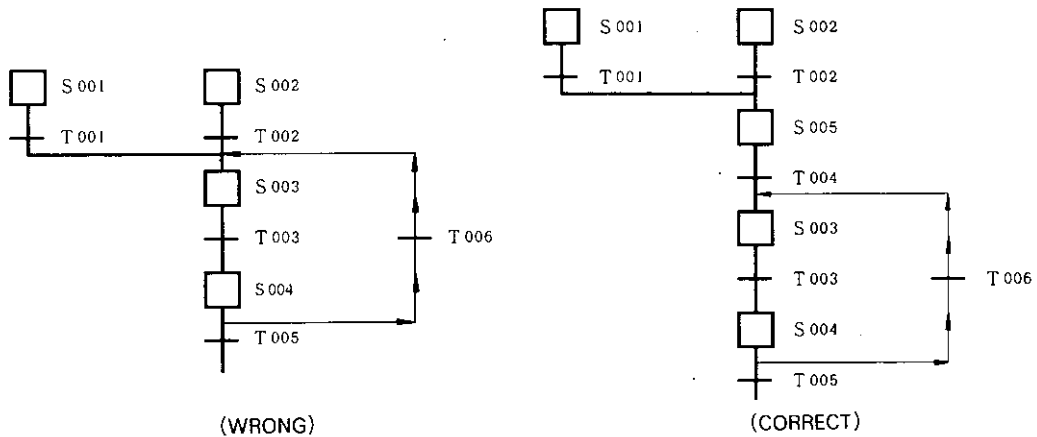
Example:



Other

As mentioned in Par. 6.7. links are divided into two groups: one is placed above transitions, the other one below. links of different groups cannot be used as a link elements.

Example: Convergence and loop input



Therefore, as shown in the example at the right, a dummy step (S005) has to be inserted between S002 and S003. A transition condition circuit whose transition coil always switches ON if solved should be created as the dummy transition T004. Evolution from S002 to S003 delays by one scan in this case.

6.6.2 SFC Evolution Prohibited Items

The two prohibited items introduced below allow storage from the P150 programming panel but malfunctions evolution during execution. Sufficient care must be exercised so that the programs described below are not created during SFC program creation.

Convergence inside Simultaneous Sequence

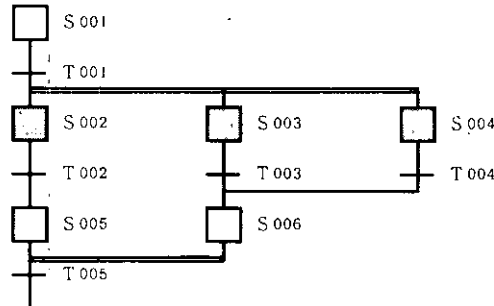


Fig. 6.36 Convergence in Simultaneous Sequence (1)

Assume an SFC flow as shown in Fig. 6.36 with active S002, S003 and S004 steps.

If only the T004 transition coil switches ON in one scan, S002, S003 and S006 become active steps. This is illustrated in Fig. 6.37.

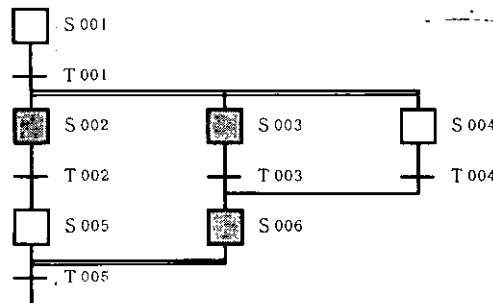


Fig. 6.37 Convergence in Simultaneous Sequence (2)

In Fig. 6.37, top and bottom steps, S003 and S006 are active. Based on the principle that steps do not proceed if steps at destinations are active, S003 cannot proceed until S006 proceeds even if the T003 transition coil is switched ON.

Assume that S002 proceeds to S005 and further to other destinations after the T005 transition coil switches ON. S003 can proceed to S006, but cannot proceed indefinitely as S005 is not active. This is illustrated in Fig. 6.33.

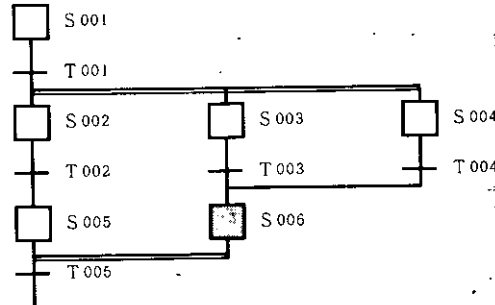


Fig. 6.38 Convergence in Simultaneous Sequence (3)

Divergence in Simultaneous Convergence

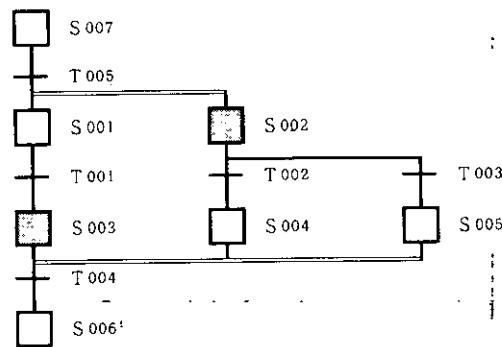


Fig. 6.39 Divergence in Simultaneous Sequence (1)

As an example, assume that we have an SFC flow as shown in Fig. 6.34 and that S002 and S003 are active. If the transition coil of T003 transition condition circuit switches ON, S002 proceeds to S005. This is illustrated in Fig. 6.35.

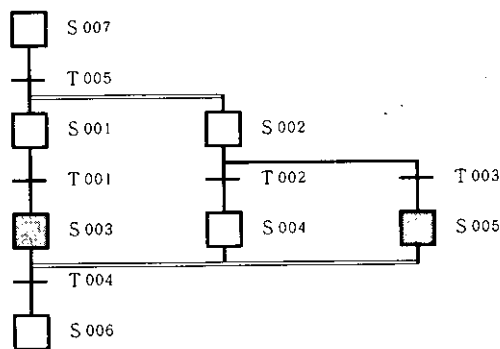


Fig. 6.40 Divergence in Simultaneous Sequence (2)

Simultaneous convergence does not proceed unless all the steps connected to it become active, and S003 and S005 cannot proceed indefinitely.

Precautions

- Two examples on SFC evolution prohibited items were described above. However, this problem can be solved if divergence and convergence, as well as simultaneous divergence and simultaneous convergence, are used in pair in a simultaneous sequence as shown in the following examples.

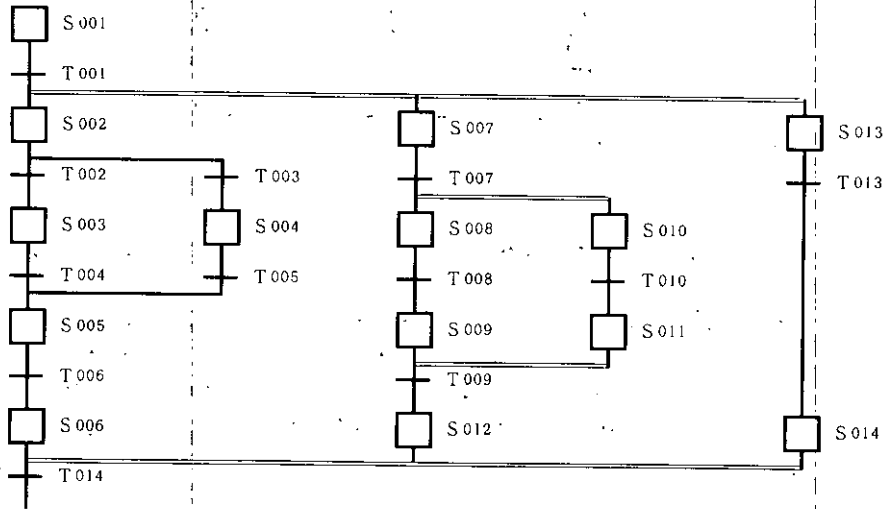


Fig. 6.41 Correct Divergence and Convergence in Simultaneous Sequence

- Correct evolution is possible if simultaneous divergence, simultaneous convergence, divergence and convergence are ultimately in pair as shown in the following examples.

① Simultaneous divergence and simultaneous convergence

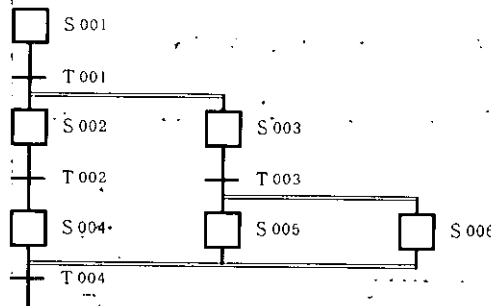


Fig. 6.42 Correct Connection in Simultaneous Divergence and Simultaneous Convergence

The example in Fig. 6.42 shows two simultaneous divergence connections and one simultaneous convergence connection. This example is correct because three independent simultaneous sequences that ultimately become independent from one sequence merged into one sequence by simultaneous convergence.

② Divergence and convergence

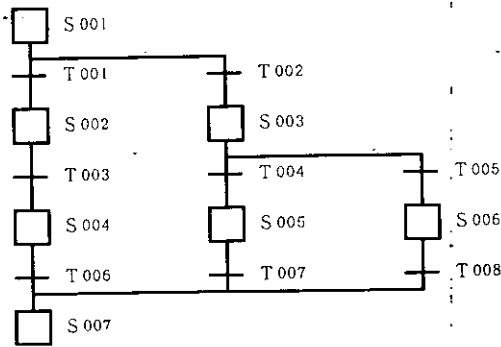


Fig. 6.43 Correct Connection of Divergence and Convergence

The example in Fig. 6.43 has two divergence connections and one convergence connection and does not seem to be in pair. However, the active step proceeding from S001 always reaches S007 at the end regardless of which path it follows. Therefore, this connection is not wrong.

TO can also be used instead of convergence as shown in Fig. 6.44.

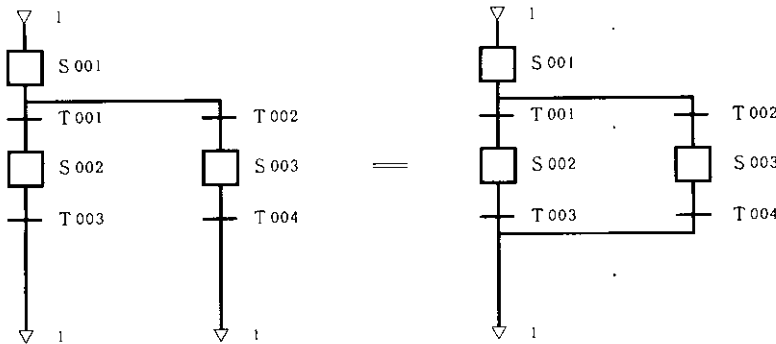


Fig. 6.44 Substitution by TO of Convergence

Macro returns can also substitute convergence.

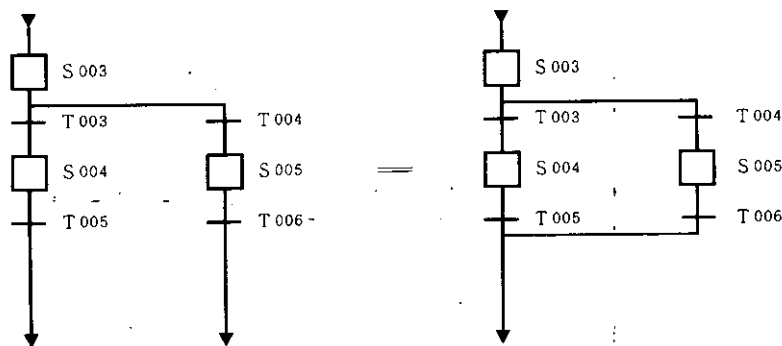


Fig. 6.45 Substitution of Convergence by Macro Return

• Special Usage

The following control sequence is correct usage even if simultaneous divergence and simultaneous convergence are not paired.

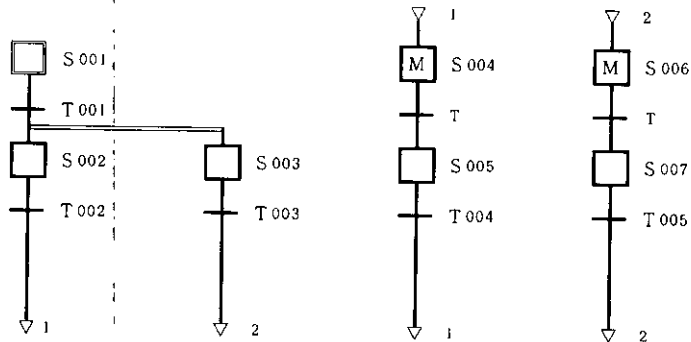


Fig. 6.46 Special Usage of Simultaneous Divergence

This SFC flow shifts to two simultaneous sequences by simultaneous divergence after initialization and then executes two perfectly independent controls in parallel.

SFC of GL60H or GL70H has only one initial step. Mutually independent active steps must be created by such an SFC flow to continuously execute more than two independent controls. This can also be executed by presetting mode processing. Nevertheless, by creating such a program, preset timing and other elements do not have to be considered.

One step is provided below each macro step to prevent inability of evolution without it after the second cycle. Refer to Par. 6.9 for details.

6

6.7 SFC VIEW FORMAT

The SFC view format is illustrated below. Sixty-four of these views can be used. Each view comprises eight elements horizontally, as well as eight lines each of step and transition columns and one each FROM line and TO line (0 and 9) vertically. Transition elements and step elements cannot be programmed in step and transition lines, respectively.

		COLUMN							
		1	2	3	4	5	6	7	8
LINE	0	FROM Line							
	1S	First Step Line							
	1T	First Transition Line							
	2S	Second Step Line							
	2T	Second Transition Line							
	⋮	⋮							
	8S	Eighth Step Line							
	8T	Eighth Transition Line							
	9	TO Line							

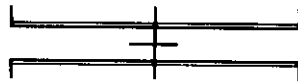
Fig. 6.47 SFC View Format

(1) The SFC program elements that are allowed for these lines are as follows.

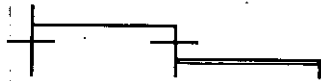
- Program elements allowed for FROM line
 - FROM connector
 - Macro entry
 - Simultaneous divergence
 - Loop input from the left and from the right
- Program elements allowed for step lines 1S to 8S
 - Initial step
 - Step
 - Macro step
 - Link line
 - Counter link line
- Program elements allowed for transition lines 1T to 8T
 - Transition
 - Counter transition
 - Link line
 - Counter link line
 - Links (simultaneous divergence, simultaneous convergence, divergence, convergence, loop input from the left and from the right, loop output to the left and to the right)
- Program elements allowed for TO line
 - Macro return
 - TO connector
 - Simultaneous convergence
 - Loop output to the left and to the right

(2) Basically, one link is stored in one transition line. However, the following combinations can be stored in one line.

(a) Simultaneous convergence and simultaneous divergence



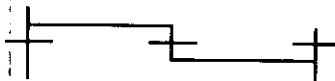
(b) Divergence and simultaneous divergence



(c) Simultaneous convergence and convergence



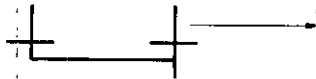
(d) Divergence and convergence



(e) Loop output and simultaneous divergence



(f) Loop output and convergence



(3) As the P150 programming panel screens show, links have two groups: one group is put above transitions and the second group is put below.

Table 6.5 Transition and Link

Put above	Divergence, simultaneous convergence, loop output to the left and to the right
Put below	Simultaneous divergence, convergence, loop input from the left and from the right

The links are put above and below for some reason. Links that are put above or below cannot be put between a step and a transition or between a transition and a step.

6.8 EXAMPLES OF SFC FLOW SEQUENCE

This section creates several SFC programs using the SFC program elements described and reviews their evolution states.

6.8.1 Program Example (1)

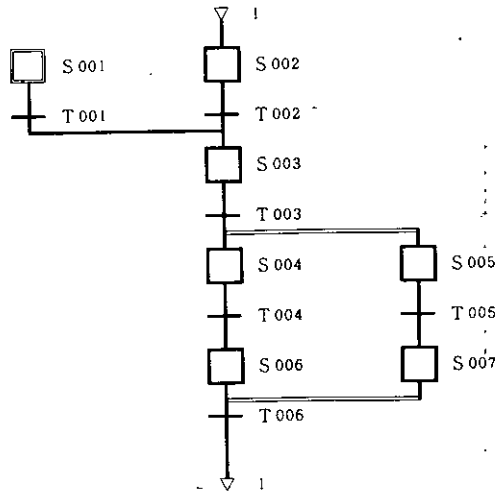


Fig. 6.48 SFC Program Example (1)

This SFC flowchart shows a program for the sequence to proceed to S003 if the T001 transition coil switches ON (condition is established) or to S004 and S005 if the T003 transition coil switches ON using S001 as the starting point (initial step). The initial step S001 deviate from the overall flow because basically it is for starting and does not have to be processed after rising in many cases. For this reason, the initial step may be put outside the overall flow.

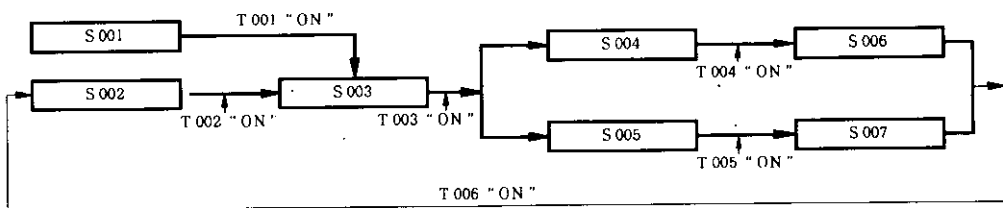
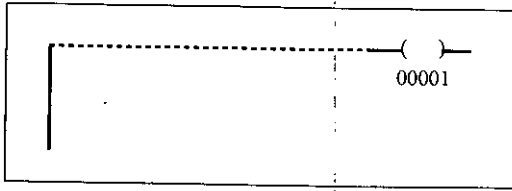


Fig. 6.49 State Transition

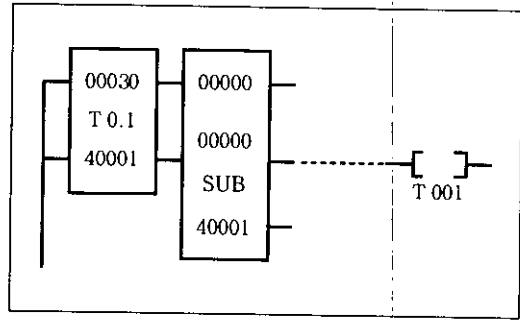
The S001 action circuit has a ladder which is solved when S001 becomes active. The ladder which becomes the decision condition for evolution from S001 to S003 after the scan next to the one during which S001 became active is the transition condition circuit paired with the T001 Transition. The transition condition circuit always has a transition coil. Switching on of this coil means that the evolution condition is established. The transition condition circuit solves only transitions that are connected to active steps. The action circuit also solves only active steps.

Assume that action circuits and transition condition circuits of the steps and transitions are simple ladders as shown below.

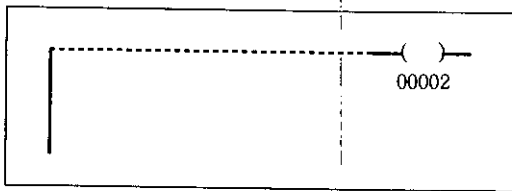
① S 001 Action Circuit



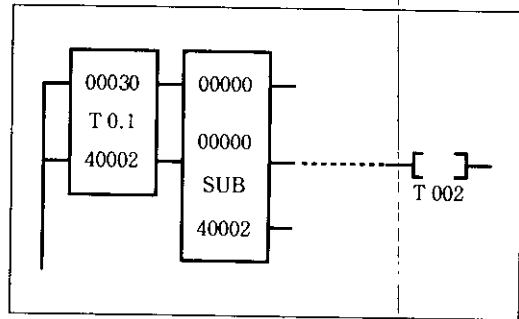
② T 001 Transition Condition Circuit



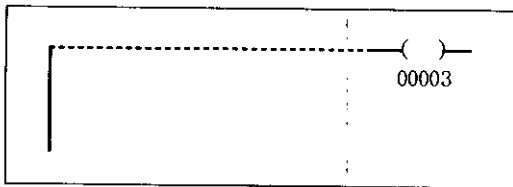
③ S 002 Action Circuit



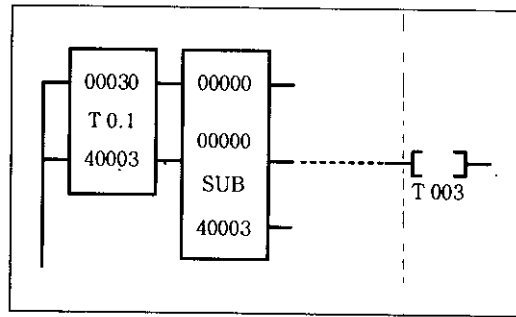
④ T 002 Transition Condition Circuit



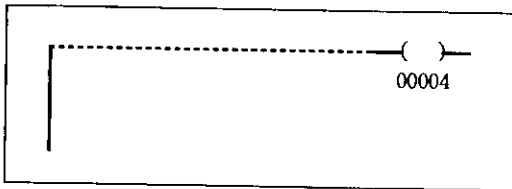
⑤ S 003 Action Circuit



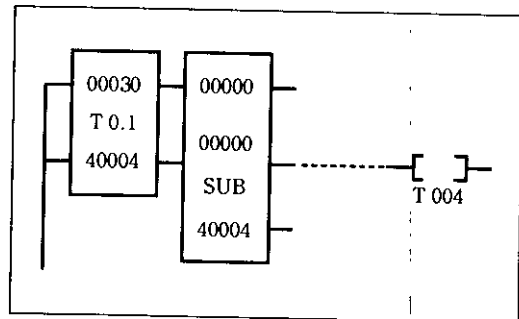
⑥ T 003 Transition Condition Circuit



⑦ S 004 Action Circuit

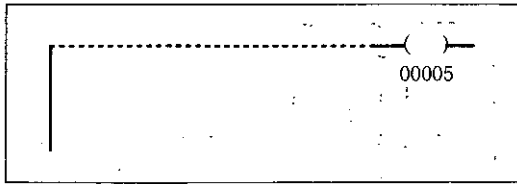


⑧ T 004 Transition Condition Circuit

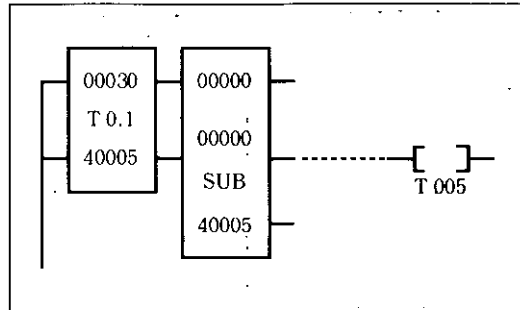


6.8.1 Program Example (1) (Cont'd)

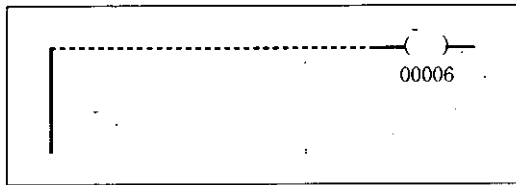
⑨ S 005 Action Circuit



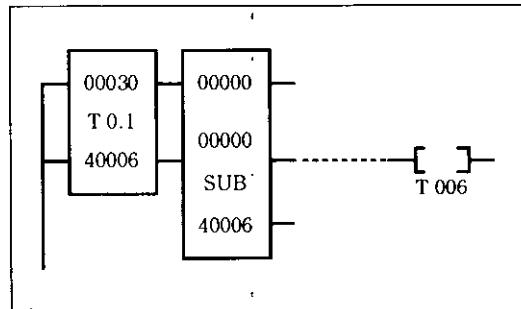
⑩ T 005 Transition Condition Circuit



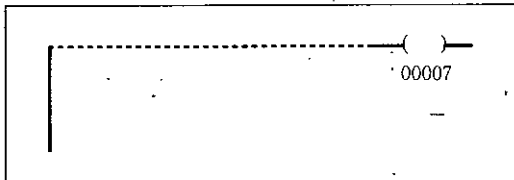
⑪ S 006 Action Circuit



⑫ T 006 Transition Condition Circuit



⑬ S 007 Action Circuit



In the transition condition circuit, the transition coil switches ON in 3 sec after the related step becomes active. "SUB" clears the 3-sec timer register and operates again beginning with "present value = 0" in the next cycle. Unless this circuit is incorporated, the next cycle operates beginning with "present value = 30" in the next cycle and proceeds with one scan. Note that the values of the registers in the action and transition condition circuits are not cleared during evolution.

The coils of the action circuits switch ON when the steps become active. In practice, these coils become external output coils. The coils automatically switch OFF during evolution.

Mode allocation is allocated with initialization of Input Relay 10001 ON. Fig. 6.50 shows the timing chart after Input Relay 10001 is switched ON and then OFF again (initialization is executed). (Action sequence in non-holding mode.)

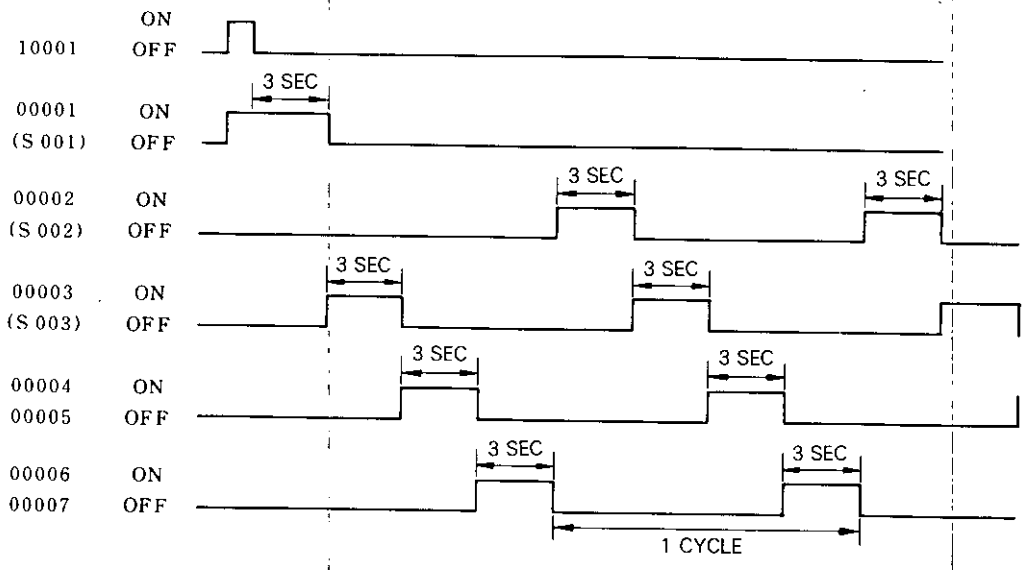


Fig. 6.50 SFC Program Timing Chart (1)

This timing chart can be viewed on the P150 programming panel screen by using the trace back function described in Par. 4.8. Switching ON and OFF again in the same timing sequence as that of the coil as shown above is also possible by monitoring the S001, 002, 003, 004, 005, 006 and 007 by the normally open contact in the ladder program (program solved every scan).

The timer registers 50001, 50002, 50003, 50004, 50005, 50006 and 50007 measure active time of Steps S001, 002, 003, 004, 005, 006 and 007 in units of 100 msec. (This function can be used to monitor step delay time.)

6.8.2 Program Example (2)

Program Example (2) is an example that includes a macro step.

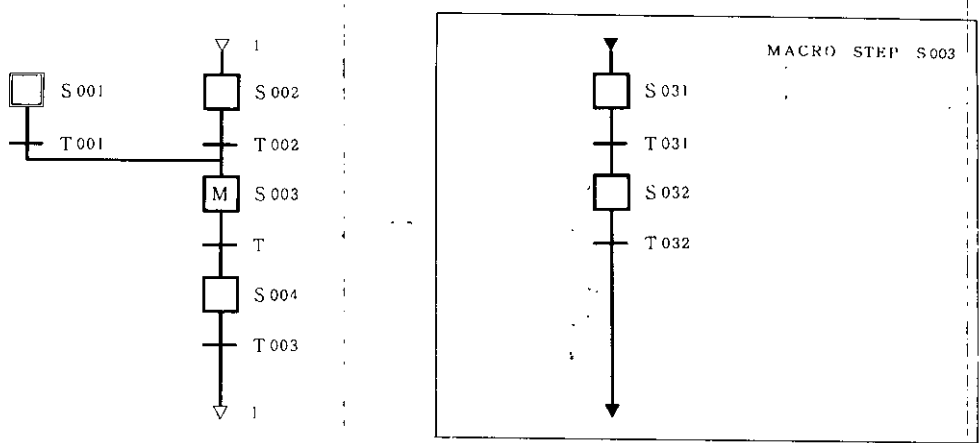


Fig. 6.51 SFC Program Example (2)

As mentioned in Par. 6.8.1, the transition condition circuit of each transition clears the value of the present-value storage register of the timer in 3 sec after the related step becomes active to switch ON the transition coil. The action circuit of each step switches ON a coil.

6.8.2 Program Example (2) (Cont'd)

Fig. 6.52 shows a timing-chart.
 (Action sequence in the non-holding mode.)

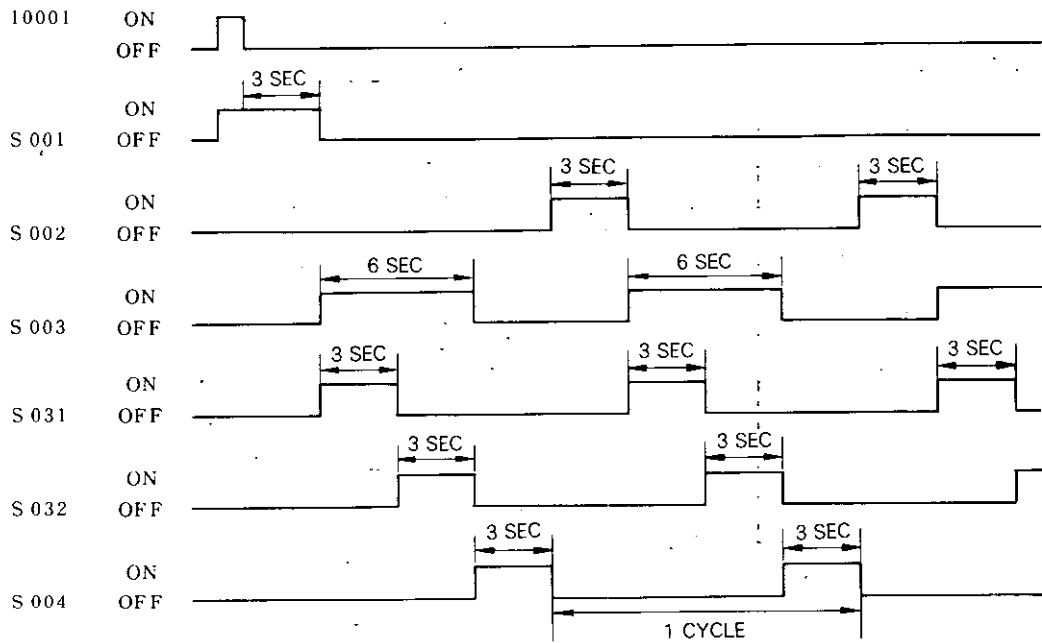


Fig. 6.52 SFC Program Timing Chart (2)

Macro Step S 0 0 3 shows active while an expanded view has an active step. For this reason, the timing chart becomes active for 6 sec.

6.8.3 Program Example (3)

Program example (3) shows an example of simultaneous convergence.

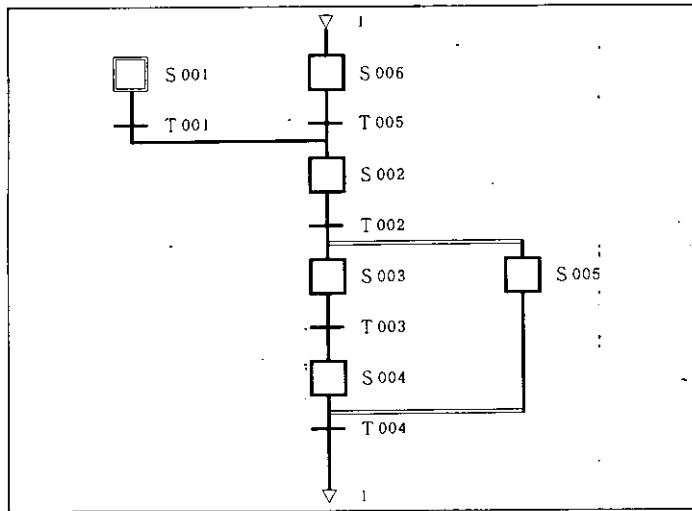


Fig. 6.53 SFC Program Example (3)

As mentioned in Par. 6.8.1, the transition condition circuit of each transition clears the value of the present-value storage register of the timer in 3 sec after the related step becomes active to switch ON the transition coil. The action circuit of each step switches ON a coil.

Fig. 6.54 shows the timing chart. (Action sequence in the non-holding mode.)

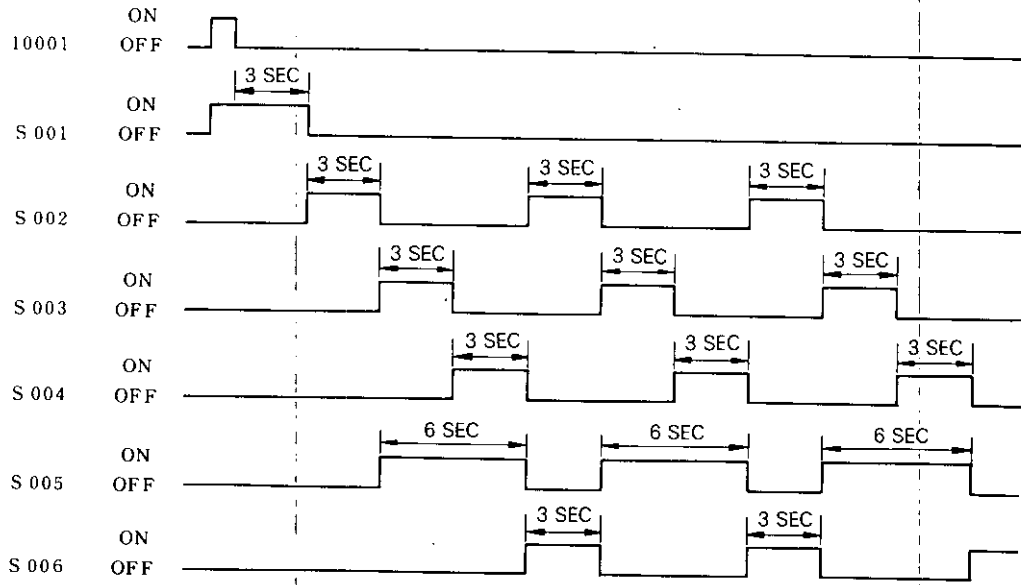


Fig. 6.54 SFC Program Timing Chart (3)

In simultaneous convergence connection, only after all the steps connected to the simultaneous convergence circuit become active, the transition condition circuit next to the steps is solved. Therefore, the transition condition circuit T004 is not solved even if Step S005 becomes active and is started to be solved only after S004 becomes active. For this reason, S005 becomes active for 6 sec.

6.8.4 Program Example (4)

Program example (4) is an example of divergence connection.

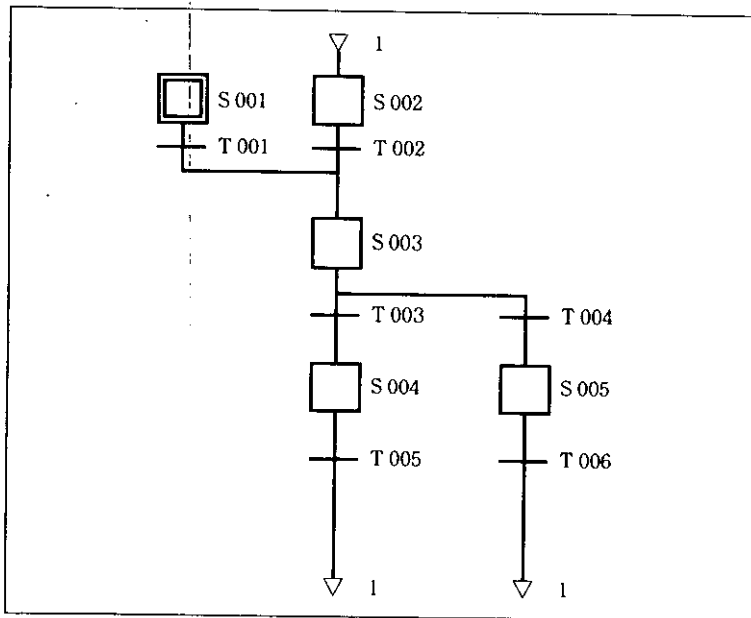


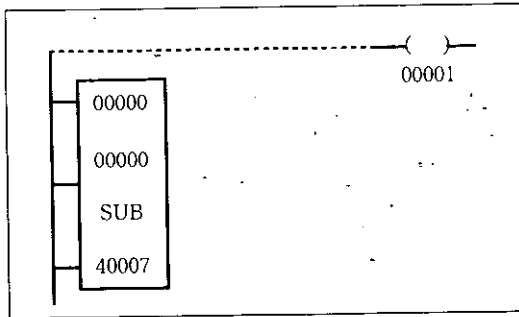
Fig. 6.55 Example of Divergence Connection

Assume action and transition condition circuits that solve S002 and S003 alternately every cycle. The simplest method is to decide whether the value of a holding register is "0" or "1."

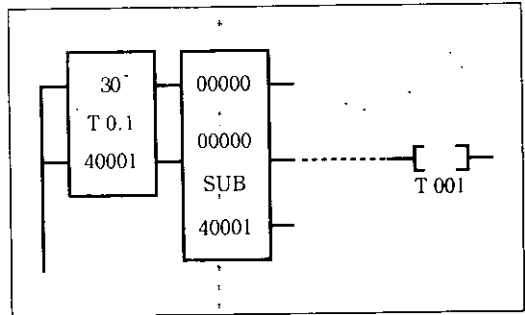
The action and transition condition circuits for this case are shown below.

6.8.4 Program Example (4) (Cont'd)

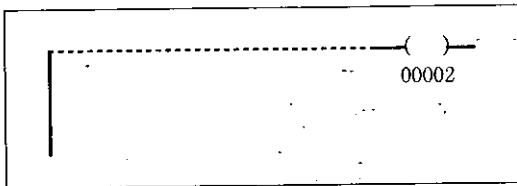
① S001 Action Circuit



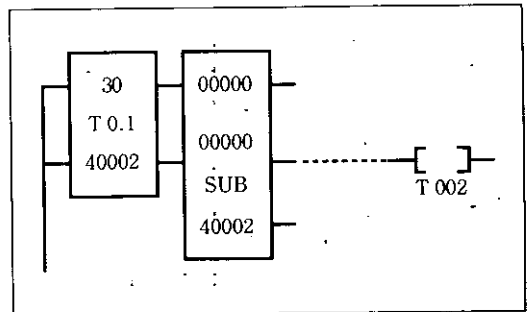
② T001 Transition Condition Circuit



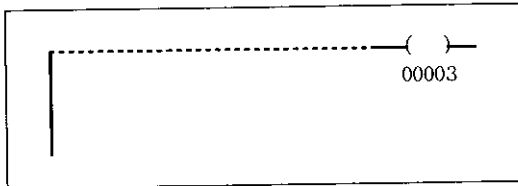
③ S002 Action Circuit



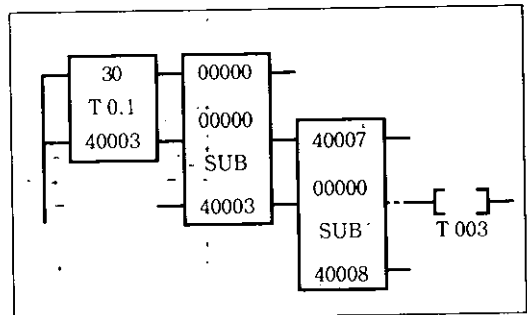
④ T002 Transition Condition Circuit



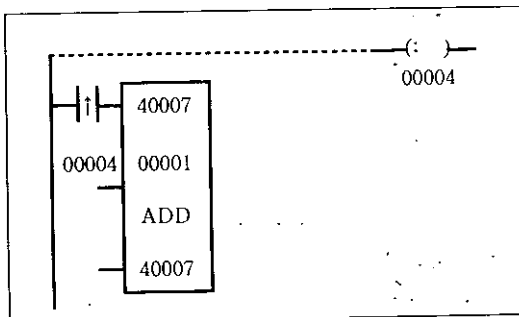
⑤ S003 Action Circuit



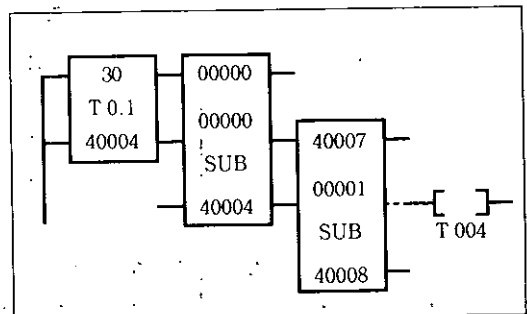
⑥ T003 Transition Condition Circuit



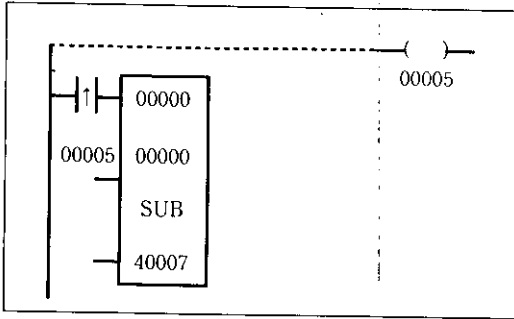
⑦ S004 Action Circuit



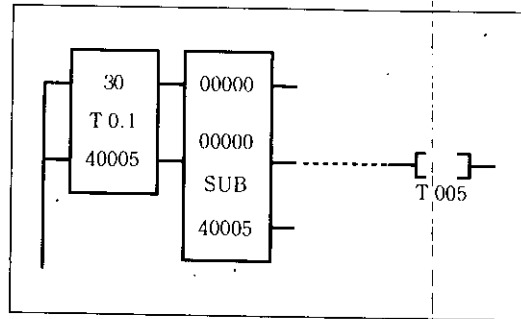
⑧ T004 Transition Condition Circuit



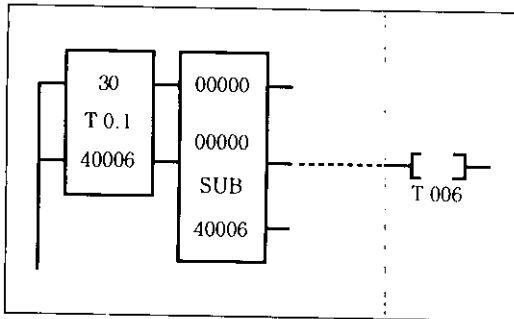
⑨ S005 Action Circuit



⑩ T005 Transition Condition Circuit



⑪ T006 Transition Condition Circuit



Initial Step S001 becomes active and the divergence control holding register 40007 is cleared if the initialize condition is established. S003 becomes active afterward. In 3 sec, the sequence proceeds to S004 as the value of holding register 40007 is 0. The value of Holding Register 40007 becomes "1" in the S004 action circuit. Therefore, in the next cycle, the sequence proceeds from S003 to S005. This sequence is repeated afterward and the sequence proceeds to S004 and to S005 alternately. Fig. 6.51 shows the timing chart of this action. (Action sequence in the non-holding mode.)

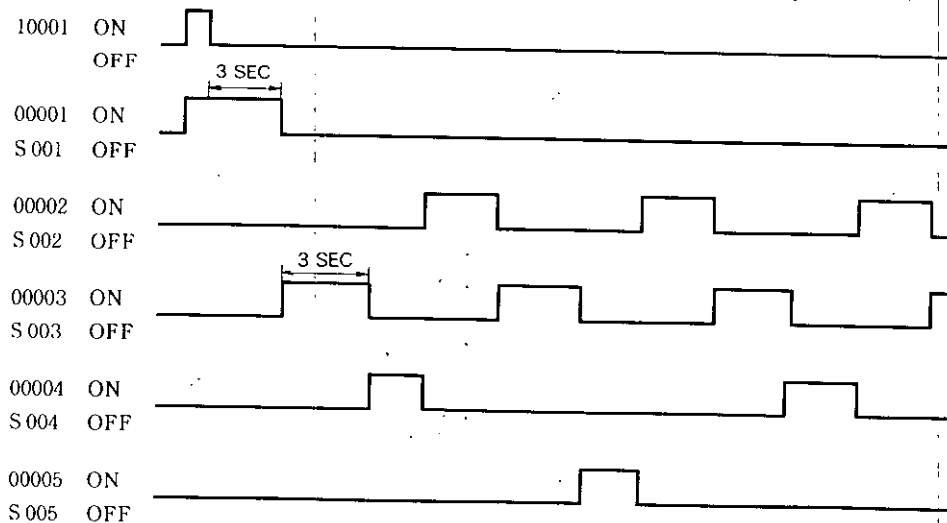


Fig. 6.56 SFC Program Timing Chart (4)

Fig. 6.56 shows an example of simple divergence. Divergence conditions can be controlled by the SFC transition condition circuit and also based on the ladder circuit state.

6.8.5 Program Example (5)

Fig. 6.57 shows an example of loop connection

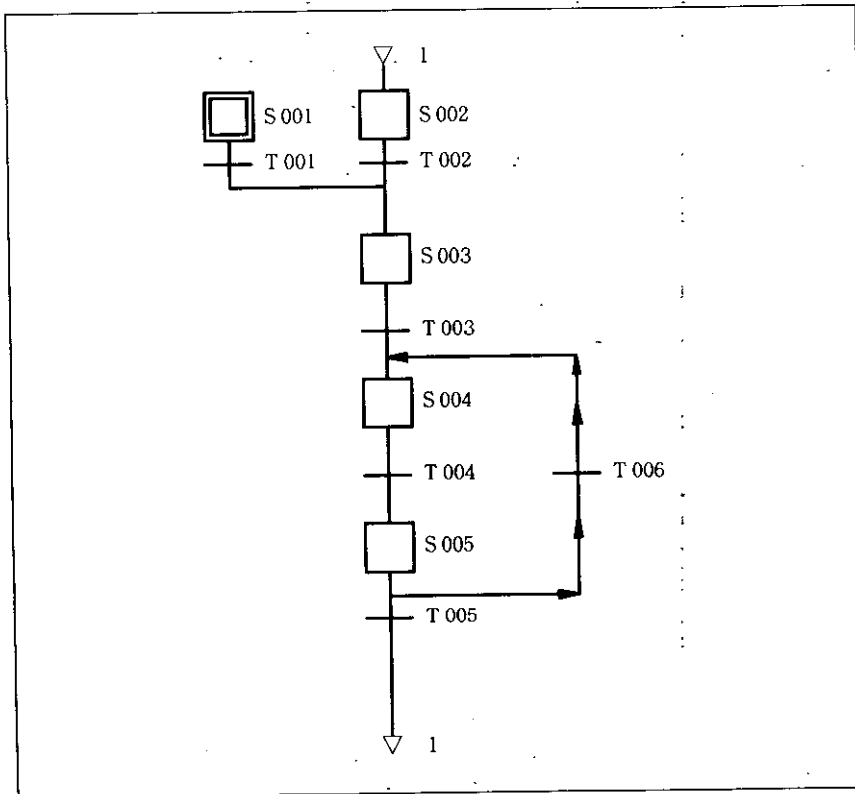
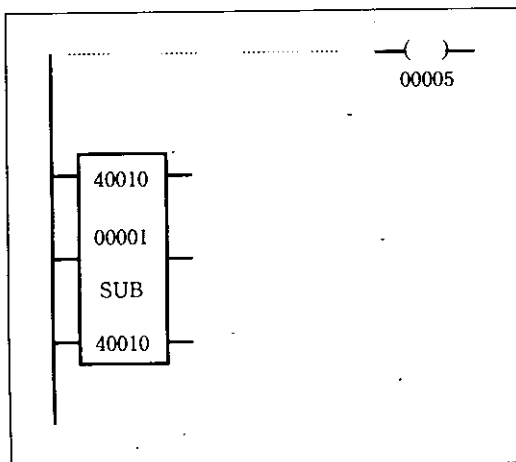


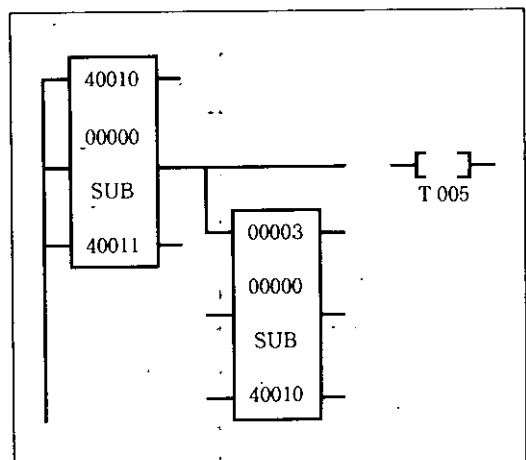
Fig. 6.57: SFC Program Example (5)

Assume that the S005 action and T005 transition condition circuits are programmed as follows and that the other transition condition circuits contain a circuit which always switches ON the transition coil if solved. It is also assumed that the value of Holding Register 40010 that controls number of loop cycles is set to 3 by Initial Step S001 and Step S002.

① S005 Action Circuit



② T005 Transition Control Circuit



The data of Holding Register 40010 decreases by 1 and becomes 2 when S005 becomes active for the first time. In the next scan, the transition condition circuit of Transition T005 is solved. However, the data value of Holding Register 40010 is 2 and the transition coil does not switch ON. Thus, the T006 transition condition circuit is solved. The transition coil switches ON and the loop proceeds to Step S004. The value of the holding register is 1 even if S005 becomes active, and the loop proceeds to Step S004 again. If S005 becomes active again, Output 2 of SUB in the T005 transition condition circuit switches ON and the transition coil switches ON since the value of the holding register is 0. The sequence proceeds to Step S002 through TO and FROM. The S002 action circuit sets the value of Holding Register 40010 to 3 and loop action is repeated.

In this example, evolution occurs in one scan and presents no problem. However, if several scans are needed for evolution, S005 Action Circuit must be designed carefully so that the value of the holding register to control the number of loop cycles is not updated more than twice in one cycle of loop.

Numbers of loop cycles must not be controlled using a counter.

Fig. 6.58 shows the timing chart.
(Action sequence in the non-holding mode.)

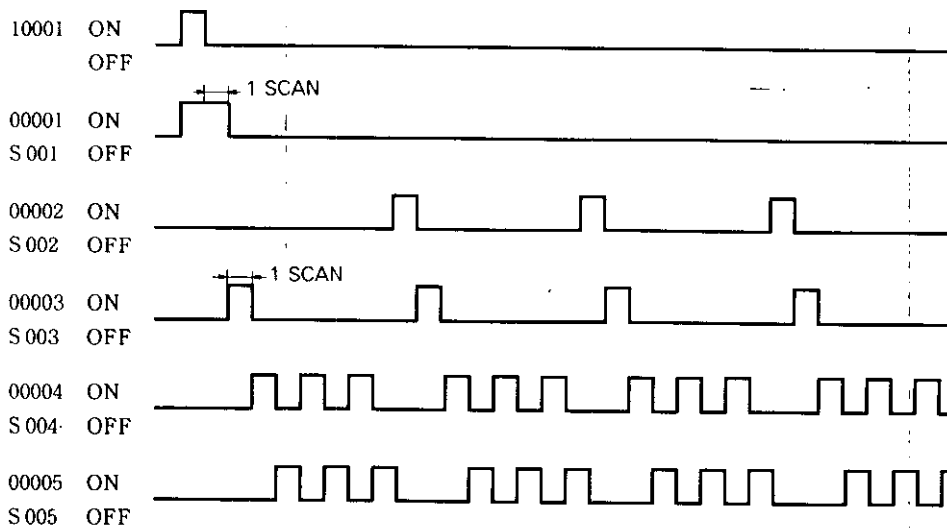


Fig. 6.58 SFC Program Timing Chart (5)

6.9 METHOD OF EVOLUTION IN VARIOUS CIRCUMSTANCES.

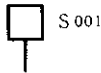
Par. 6.4 described various methods of evolution when perfect programs were operated normally. This section describes methods of evolution if programs are unfinished or if erroneous operation is made in mode processing or on the P150 programming panel.

This section sometimes mentions that the transition coil is switched ON or OFF. This shows the state when solving is actually executed. Note that the transition coil does not have an ON or OFF state if it is not solved.

6.9.1 Evolution of Program Unfinished

(1) No Transition to be Connected to Step

Example:



Step S001 has no destination in this case. S001 continues to be held until the destination is stored.

(2) Transition with No Transition Condition Circuit

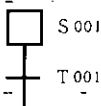
Example:



Step evolution takes place when the transition coil of the transition condition circuit of the transition to be connected to the step is switched ON. In this case, the transition condition circuit is not provided and, naturally, a transition coil is not provided either. Therefore, S001 continues to be held.

(3) No Step to be Connected to Transition

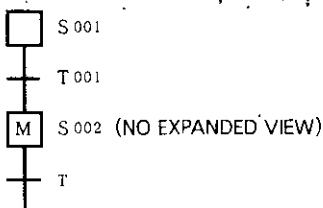
Example:



S001 is kept held even if the T001 Transition Coil switches ON after S001 becomes active since there is no step at the destination.

(4) No Expanded View in Macro Step

Example:



If S001 becomes active and the T001 transition coil switches ON, only S002 becomes active as in ordinary steps since Macro Step S002 has no expanded views. Dummy transition is provided below, giving the same effect to the circuit as if the transition coil is switched ON in every scan. The sequence proceeds downward in the next scan after S002 becomes active if there is a step below and disable, etc. are not set. Macro Step S002 has no action circuit and does not perform control.

Macro Step S002 is indefinitely active if there are no steps that are connected to the macro step as illustrated above. Steps of the expanded view do not become active even if steps are stored in the expanded view.

6.9.2 Evolution with Active Steps Arranged Vertically

Basically, active steps are not arranged vertically. However, active steps are arranged vertically due to presetting or to connections mentioned in Par. 6.5. This paragraph describes how evolution occurs in these cases.

(1) Sequence of GL60H, GL70H Step Solving

First, the sequence of solving the GL60H, GL70H step is described. This is because the evolution method changes in accordance with the solving sequence.

SFC solving by initializing starts with the initial step. There is one active step. This is then followed by:

- More than two active steps created by simultaneous divergence which are solved beginning with the left step.
- Active steps created by presetting which are solved at the end. Steps preset first are solved if several steps are preset.

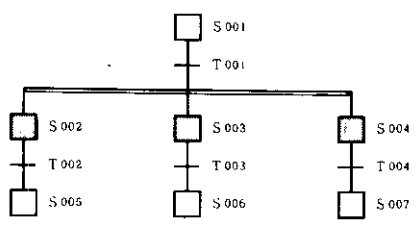


Fig. 6.59 Sequence of SFC Active Step Solving (Simultaneous Divergence)

Fig. 6.59 shows Active Step S001 proceeding to S002, S003 and S004 after T001 Transition Coil switches ON. The solving sequence after the scan that caused this state are solved in the sequence of S002, S003 and S004 from the left to the right.

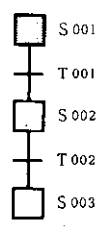


Fig. 6.60 Sequence of SFC Active Step Solving (Presetting)

The solving sequence after the scan in which Step S002 is preset in the state shown in Fig. 6.60 is S001 and S002.

6.9.2 Evolution with Active Steps Arranged Vertically (Cont'd)

(2) Evolution with Simple Connection

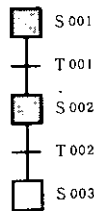


Fig. 6.61 Evolution with Simple Connection

In scanning with the situation as shown in Fig. 6.61:

- ① Both T001 and T002 transition coils are switched OFF.
No evolution occurs.
- ② T001 transition coil is OFF, T002 Transition Coil is ON.
S002 proceeds to S003 and S001 is in a held state.
S001 and S003 are active steps.
- ③ T001 transition coil is ON, T002 Transition Coil is OFF.
No evolution occurs.
- ④ Both T001 and T002 Transition Coils are ON.

Evolution method changes depending on in which sequence GL60S solves two steps.

• Solving S001 and then S002

If S001 is solved first, no evolution occurs since S002 at the destination is active, and S001 remains active. S002 is then solved, and the sequence proceeds since S003 is inactive. Therefore, Steps S001 and S003 will be active after one scan. In the next scan, S001 proceeds to S002.



Fig. 6.62 Evolution Solved in Sequence of S001 to S002

• Solving in sequence of S002 to S001

The sequence proceeds to S003 if S002 is solved first since S003 is inactive. S001 is solved next. The sequence proceeds to S002 since S002 is already inactive. Therefore, unlike the sequence of S001 and S002, active steps shift to S002 and S003 in one scan.

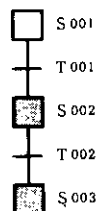


Fig. 6.63 Evolution Solved in Sequence of S002 to S001

Table 6.6 summarizes these cases.

Table 6.6 Evolution in Simple Connection

Transition Coil State		Active Step		Solving Sequence
T001	T002	After 1 Scan	After 2 Scans	
OFF	OFF	S001, S002		
	ON	S001, S003		
ON	OFF	S001, S002		
	ON	S001, S003	S002, S003	
		S002, S003		S002→S001

(3) Evolution in Divergence Connection

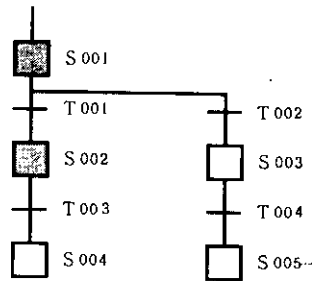


Fig. 6.64 Evolution in Divergence Connection (1)

In a scan in the situation shown in Fig. 6.64:

① All T001, T002 and T003 transition coils are OFF.

No evolution occurs.

② T001 and T002 Transition Coils are OFF, T003 Transition Coil is ON.

Only S002 proceeds to S004.

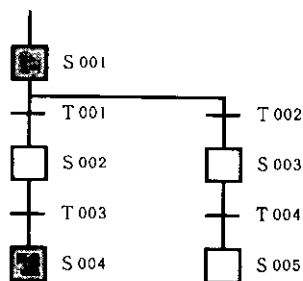


Fig. 6.65 Evolution in Divergence Connection (2)

6.9.2 Evolution with Active Steps Arranged Vertically (Cont'd)

③ T001 and T003 Transition Coils are OFF, T002 Transition Coil is ON. Only S001 proceeds to S003.

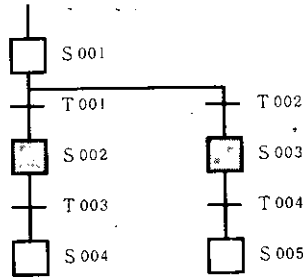


Fig. 6.66 Evolution in Divergence Connection (3)

④ T001 Transition Coil is OFF, T002 and T003 Transition Coils are ON. S001 and S002 proceed to S003 and S004.

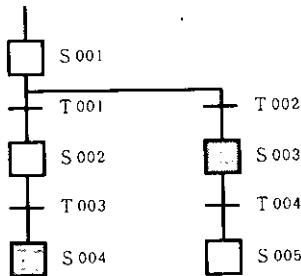


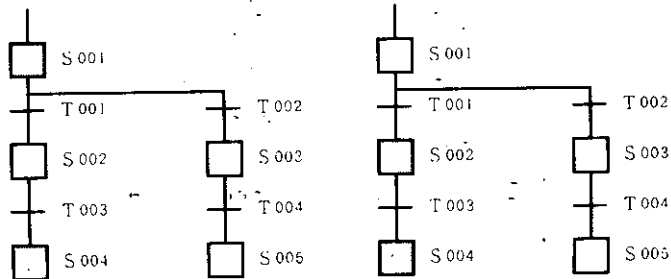
Fig. 6.67 Evolution in Divergence Connection (4)

⑤ T001 Transition Coil is ON, T002 and T003 Transition Coils are OFF. No evolution occurs.

⑥ T001 and T003 Transition Coils are ON, T002 Transition Coil is OFF. The sequence proceeds by the following two methods depending on the sequence of active step solving.

- Solving S001 and then S002

In the first scan, S001 cannot proceed since S002 at the destination of the other side is active. S003 on the other side cannot proceed since the T002 Transition Coil is OFF. S002 proceeds to S004. S001 and S004 will be the active steps after one scan. In the next scan, S001 proceeds to S002 at this time since the destination is inactive. Thus, S002 and S004 will be the active steps after two scans.



(a) After One Scan

(b) After Two Scans

Fig. 6.68 Evolution in Divergence Connection (5)

- Solving S002 and then S001

First, S002 proceeds to S004 and S002 becomes inactive. S001 is then solved and proceeds to S002. Unlike solving in the sequence of S001 and then S002, proceeding to S002 and to S004 is completed in one scan.

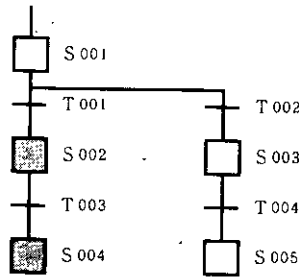


Fig. 6.69 Evolution in Divergence Connection (6)

- ⑦ T001 and T002 transition coils are ON, T003 transition coil is OFF.

S002 does not proceed since the T002 Transition Coil is switched OFF. S001 searches a destination sequentially from the left. T001 Transition Coil is switched ON, but the sequence cannot proceed since S002 is active, searching to the right. T002 Transition Coil is switched ON, and S003 is inactive. S001 proceeds to S003.

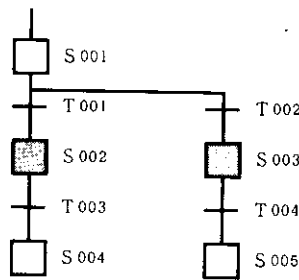


Fig. 6.70 Evolution in Divergence Connection (7)

- ⑧ All T001, T002 and T003 are ON.

Two types of evolution are available depending on the solving sequence of active steps.

- Solving S001 and then S002

S001 searches a destination sequentially from the left. The sequence cannot proceed since S002 is active even though T001 Transition Coil is switched ON, searching to the right. S001 proceeds to S003 since T002 Transition Coil is switched ON and S003 is inactive. S002 proceeds to S004 since T002 Transition Coil is switched ON.

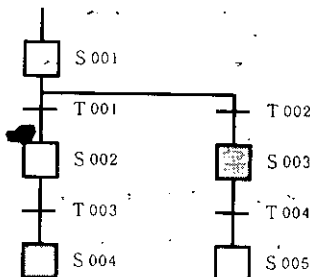


Fig. 6.71 Evolution of Divergence Connection (8)

6.9.2 Evolution with Active Steps Arranged Vertically! (Cont'd)

- Solving S002 and then S001

First, S002 proceeds to S004 and becomes inactive. S001 is then solved and proceeds to S002. Proceeding to S002 and S004 is completed in one scan unlike solving in the sequence of S001 and S002.

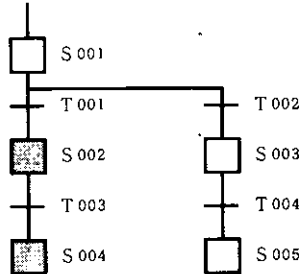


Fig. 6.72 Evolution in Divergence Connection (9)

Table 6.7 summarizes the above situations.

Table 6.7 Evolution in Divergence Connection

Transition Coil State			Active Step		Solving Sequence		
T001	T002	T003	After 1 Scan	After 2 Scans			
OFF	OFF	OFF	S001, S002		—		
		ON	S001, S004				
	ON	OFF	S002, S003				
		ON	S003, S004				
ON	OFF	OFF	S001, S002		—		
		ON	S001, S004	S002, S004		S001→S002	
		S002, S004		S002→S001			
	ON	OFF	S002, S003			—	
		ON		S003, S004		S001→S002	
				S002, S004		S002→S001	

(4) Evolution in Simultaneous Divergence Connection

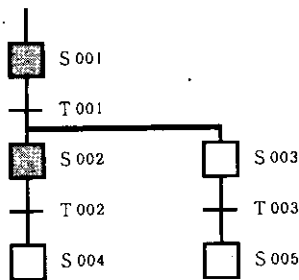


Fig. 6.73 Evolution in Simultaneous Divergence (1)

In a scan in the situation shown in Fig. 6.73:

- ① Both T001 and T002 transition coils are OFF.
No evolution occurs.

② T001 Transition Coil is OFF, T002 Transition Coil is ON.
Only S002 proceeds to S004.

③ T001 Transition Coil is ON, T002 Transition Coil is OFF.
S002, which is one of the destinations of S001 is active and S001 does not proceed. S002 does not proceed since T002 Transition Coil is switched OFF.

④ Both T001 and T002 Transition Coils are OFF.

The following two evolution types are available depending on the solving sequence of active steps.

• Solving S001 and then S002

In the first scan, S001 does not proceed since one of the destinations, S002, is active. S002 proceeds to S004 since T002 Transition Coil is switched ON. In the next scan, S001 proceeds to S002 and to S003 since T001 Transition Coil is switched ON and S002 and S003 at destinations are both inactive.

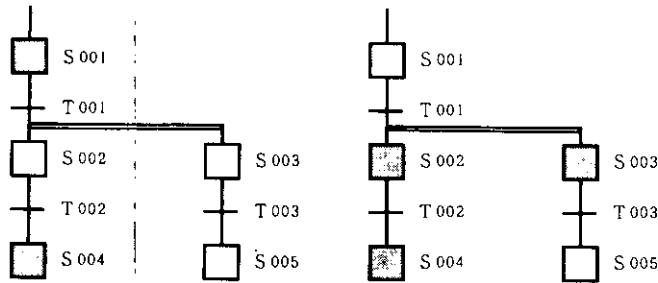


Fig. 6.74 Evolution in Simultaneous Divergence (2)

• Solving S002 and then S001

S002 proceeds to S004 since T002 Transition Coil is switched ON and S004 at the destination is inactive. S001 proceeds to S002 and S003 since T001 Transition Coil is ON and S002 and S003 at destinations are already inactive. Unlike the sequence of S001 to S002, proceeding to S002, S003 and S004 in one scan is possible.

Table 6.8 summarizes the above situations.

Table 6.8 Evolution in Simultaneous Divergence Connection

Transition Coil State		Active Step		Solving Sequence
T001	T002	After 1 Scan	After 2 Scans	
OFF	OFF	S001, S002		—
	ON	S001, S004		
ON	OFF	S001, S002		
	ON	S001, S004	S002, S003, S004	
		S002, S003, S004		S002→S001

6.9.2 Evolution with Active Steps Arranged Vertically (Cont'd)

(5) Evolution in Convergence Connection (1)

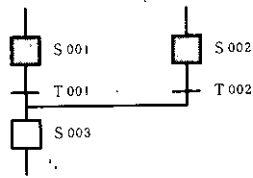


Fig. 6.75 Evolution in Convergence Connection (1)

In a scan in the situation shown in Fig. 6.70:

① Both T001 and T002 Transition Coils are switched OFF.

No evolution occurs.

② T001 Transition Coil is OFF, T002 Transition Coil is ON.

S001 does not proceed since T001 Transition Coil is switched OFF. S002 proceeds to S003 since T002 Transition Coil is ON and the step at the destination is inactive.

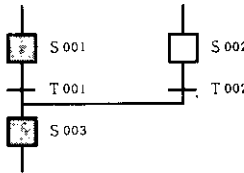


Fig. 6.76 Evolution of Convergence Connection (2)

③ T001 Transition Coil is ON, T002 Transition Coil is OFF.

S001 proceeds to S003 since T001 Transition Coil is switched ON and the step at the destination is inactive. S002 does not proceed since T002 Transition Coil is switched OFF.

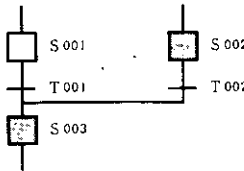


Fig. 6.77 Evolution of Convergence Connection (3)

④ Both T001 and T002 Transition Coils are ON.

The sequence proceeds by the following two methods depending on the sequence of active step solving.

- Solving S001 and then S002

S001 proceeds to S003 since T001 Transition Coil is ON and S003 at the destination is inactive. S002 does not proceed since S003 at the destination is already active even though T002 Transition Coil is ON.

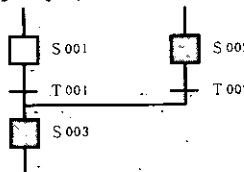


Fig. 6.78 Evolution of Convergence Connection (4)

- Solving S002 and then S001

S002 proceeds to S003 since T002 Transition Coil is ON and S003 at the destination is inactive. S001 does not proceed since S003 at the destination is already active even though T001 Transition Coil is ON.

- Solving S002 and then S001

S002 proceeds to S003 since T002 Transition Coil is ON and S003 at the proceeding destination is inactive. S001 does not proceed since S003 at the destination is already active even though T001 Transition Coil is ON.

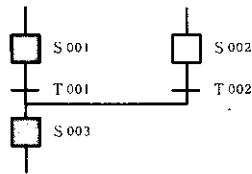


Fig. 6.79 Evolution of Convergence Connection (5)

Table 6.9 summarizes the above situations.

Table 6.9 Evolution in Divergence Connection

T001	T002	Active Step	Solving Sequence
OFF	OFF	S001, S002	
	ON	S001, S003	
ON	OFF	S002, S003	
	ON	S002, S003	
		S001, S003	S002→S001

(6) Evolution in Convergence Connection (2)

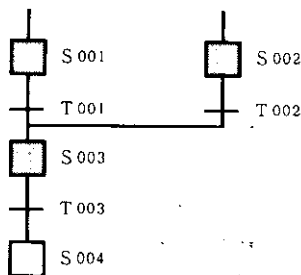


Fig. 6.80 Evolution of Convergence Connection (6)

In a scan in the situation shown in Fig. 6.80:

- ① All T001, T002 and T003 Transition Coils are OFF.

No evolution occurs.

6.9.2 Evolution with Active Steps Arranged Vertically (Cont'd)

② T001 and T002 Transition Coils are OFF, T003 Transition Coil is ON.
Only S003 proceeds to S004.

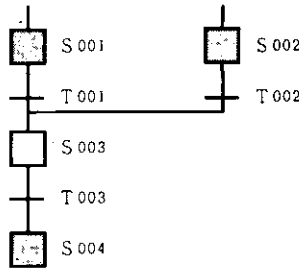


Fig. 6.81 Evolution in Convergence Connection (7)

③ T001 and T003 Transition Coils are OFF, T002 Transition Coil is ON.
No evolution occurs.

④ T001 Transition Coil is OFF, T002 and T003 Transition Coils are ON.
The following two evolution types are available depending on the solving sequence of active steps.

• Solving S002 and then S003

In the first scan, S002 cannot proceed since S003 at the destination is active even though T002 Transition Coil is ON. S003 proceeds since S004 at the destination is inactive and T003 Transition Coil is ON. S001 does not proceed since T001 Transition Coil is OFF.

In the next scan, S002 proceeds to S003 since T002 Transition Coil is switched ON and S003 at the destination is inactive this time.

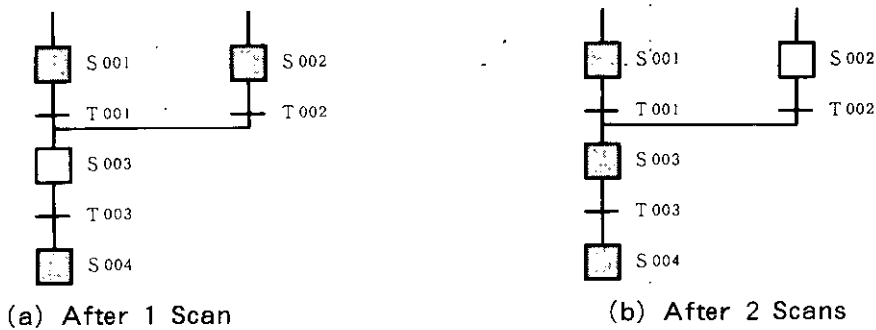


Fig. 6.82 Evolution in Convergence Connection (8)

• Solving S003 and then S002

S003 proceeds to S004 since T003 Transition Coil is ON and S004 at the destination is inactive. S002 proceeds to S003 since T002 Transition Coil is ON and S003 at the destination is already inactive.

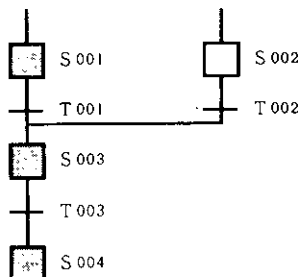


Fig. 6.83 Evolution in Convergence Connection (9)

⑤ T001 Transition Coil is ON, T002 and T003 Transition Coils are OFF. S001 does not proceed since S003 at the destination is active even though T001 Transition Coil is ON.

⑥ T001 and T003 Transition Coils are ON, T002 Transition Coil is OFF. The following two evolution types are available depending on the solving sequence of active steps.

- Solving S001 and then S003

In the first scan, S001 cannot proceed since S003 at the destination is active even though T001 Transition Coil is ON. S003 proceeds since S004 at the destination is inactive even though T003 Transition Coil is ON. S002 does not proceed since T002 Transition Coil is OFF.

In the next scan, S001 proceeds to S003 since T001 Transition Coil is switched ON and S003 at the destination is inactive this time.

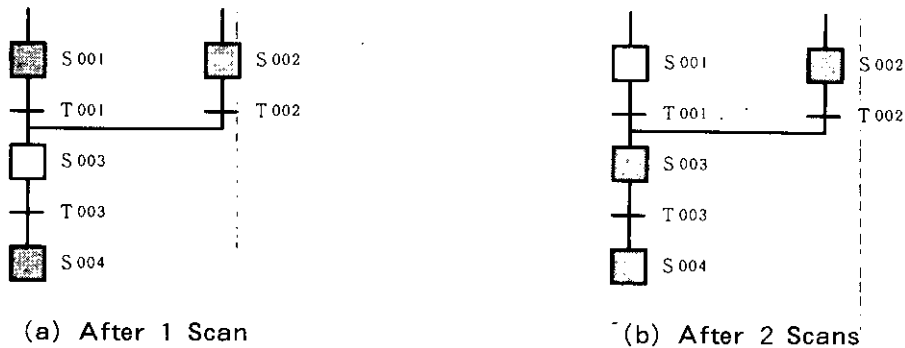


Fig. 6.84 Evolution in Convergence Connection (10)

- Solving S003 and then S001

S003 proceeds since T003 Transition Coil is switched ON and S004 at the destination is inactive. S001 proceeds to S003 since T001 Transition Coil is ON and S003 at the destination is already inactive. Therefore, unlike solving in the sequence of S001 to S003, solving ends in one scan.

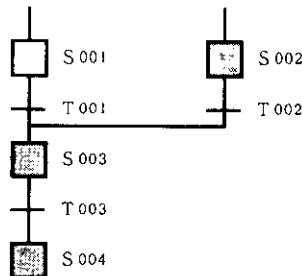


Fig. 6.85 Evolution in Convergence Connection (11)

⑦ T001 and T002 Transition Coils are ON, T003 Transition Coil is OFF. No evolution occurs.

6.9.2 Evolution with Active Steps Arranged Vertically (Cont'd)

⑧ All T001, T002 and T003 Transition Coils are ON.

Evolution methods differ depending on the sequence of active step solving.

- Solving S001, S002 and then S003

In the first scan, S001 cannot proceed since S003 at the destination is active even though T001 Transition Coil is ON. S002 cannot proceed either. S003 proceeds to S004 since T003 is active and S004 at the destination is inactive.

In the next scan, S001 proceeds to S003 since T001 Transition Coil is switched ON and S003 at the destination became inactive during the previous scan. S002 does not proceed since S003 at the destination is already active even though T002 Transition Coil is switched ON.

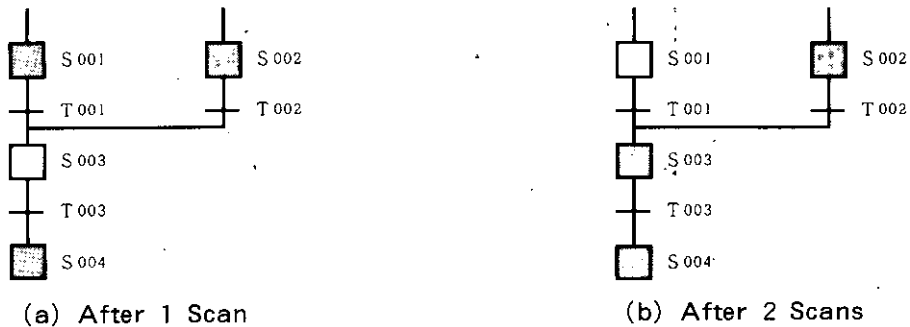


Fig. 6.86 Evolution in Convergence Connection (12)

- Solving S002, S001 and then S003

In the first scan, S002 does not proceed since S003 at the destination is active even though T002 Transition Coil is switched ON. S001 does not proceed either. S003 proceeds to S004 since T003 is active and S004 at the destination is inactive.

In the next scan, S002 proceeds to S003 since T002 Transition Coil is ON and S003 at the destination became inactive during the previous scan. S001 does not proceed since S003 at the destination has already become active even though T001 Transition Coil is ON.

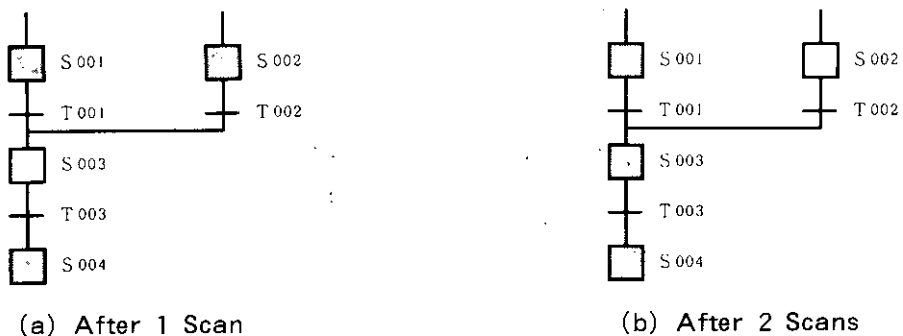


Fig. 6.87 Evolution in Convergence Connection (13)

- Solving S001, S003 and then S002

S001 does not proceed since S003 at the destination is active even though T001 Transition Coil is ON. S003 proceeds to S004 since T003 Transition Coil is active and S004 at the destination is inactive. S002 proceeds to S003 since T002 Transition Coil is switched ON and S003 at the destination is already inactive.

- Solving S003, S002 and then S001

S003 proceeds to S004 since T003 Transition Coil is ON and S004 at the destination is inactive. S002 proceeds to S003 since T002 Transition Coil is ON and S003 at the destination is already inactive. S001 does not proceed since S003 at the destination has become active again even though T001 Transition Coil is switched On.

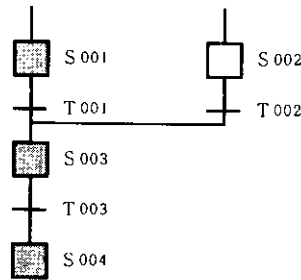


Fig. 6.88 Evolution of Cnvergence Connection (14)

- Solving S002, S003 and then S001

S002 does not proceed since S003 at the destination is active even though T002 Transition Coil is switched ON. S003 proceeds since T003 Transition Coil is ON and S004 at the destination is inactive. S001 proceeds to S003 since T001 Transition Coil is switched ON and S003 at the destination is already inactive.

- Solving S003, S001 and then S002

S003 proceeds since T003 Transition Coil is switched ON and S004 at the destination is inactive. S001 proceeds to S003 since T001 Transition Coil is switched ON and S003 at the destination is already inactive.

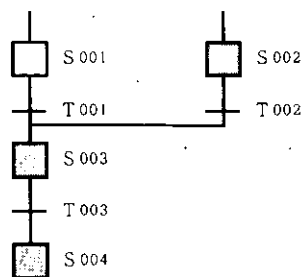


Fig. 6.89 Evolution in Convergence Connection (15)

Table 6.10 summarizes the above situations.

6.9.2 Evolution with Active Steps Arranged Vertically (Cont'd)

Table 6.10 Evolution in Convergence Connection

Transition Coil State			Active Step		Solving Sequence	
T001	T002	T003	After 1 Scan	After 2 Scans		
OFF	OFF	OFF	S001, S002, S003		—————	
		ON	S001, S002, S004			
	ON	OFF	S001, S002, S003			
		ON	S001, S002, S004	S002, S003, S004		T002→T003
			S002, S003, S004			T003→T002
ON	OFF	OFF	S001, S002, S003		—————	
		ON	S001, S002, S004	S002, S003, S004		T001→T003
			S002, S003, S004			T003→T001
	ON	OFF	S001, S002, S003			
		ON	S001, S002, S004	S002, S003, S004		T001→T002→T003
			S001, S002, S004	S001, S003, S004		T002→T001→T003
		ON	S001, S003, S004			T001→T003→T002 T003→T002→T001
	S002, S003, S004		S002, S003, S004			T002→T003→T001 T003→T001→T002

(7) Evolution in Simultaneous Convergence Connection

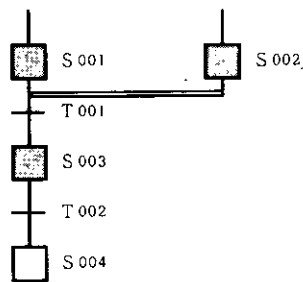


Fig. 6.90 Evolution in Simultaneous Convergence Connection (1)

In a scan in the situation shown in Fig. 6.85:

① Both T001 and T002 Transition Coils are OFF.

No evolution occurs.

② T001 Transition Coil is OFF, T002 Transition Coil is ON.

S003 proceeds to S004 since T002 Transition Coil is ON and S004 at the destination is inactive.

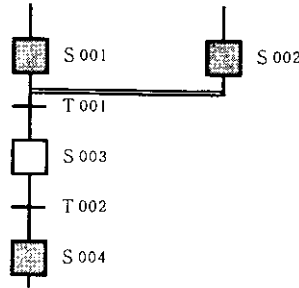


Fig. 6.91 Evolution in Simultaneous Convergence Connection (2)

③ T001 Transition Coil is ON, T002 Transition Coil is OFF.

No evolution occurs.

④ Both T001 and T002 Transition Coils are ON.

The following two evolution types are available depending on the solving sequence of active steps.

• Solving S001, S002 and then S003

In the first scan, S001 and S002 do not proceed since S003 at the destination is active even though T001 Transition Coil is ON. S003 proceeds since T002 Transition Coil is switched ON and S004 at the destination is inactive.

In the next scan, S001 proceeds since T001 Transition Coil is switched ON and S003 at the destination is already inactive.



(a) After 1 Scan

(b) After 2 Scans

Fig. 6.92 Evolution in Simultaneous Convergence (3)

• Solving S003 and then S001 (S002)

S003 proceeds to S004 since T002 Transition Coil is ON and S004 at the destination is inactive. S001 and S002 proceed to S003 since T001 Transition Coil is ON and S003 at the destination is already inactive.

6.9.2 Evolution with Active Steps Arranged Vertically (Cont'd)

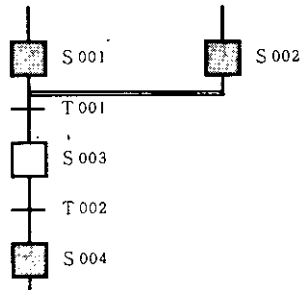


Fig. 6.93 Evolution in Simultaneous Convergence Connection(4)

Table 6.11 summarizes the above situations.

Table 6.11 Evolution in Simultaneous Convergence Connection

Transition Coil State		Active Step		Solving Sequence
T001	T002	After 1 Scan	After 2 Scans	
OFF	OFF	S001, S002, S003		—
	ON	S001, S002, S004		
ON	OFF	S001, S002, S003		
	ON	S001, S002, S004	S003, S004	
		S003, S004		S003→S001, S002

(8) Evolution in Loop Connection (1)

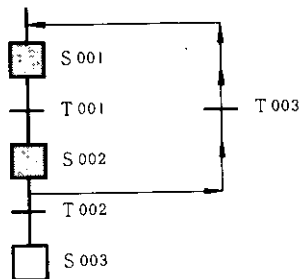


Fig. 6.94 Evolution in Loop Connection (1)

In a scan in the situation shown in Fig. 6.94.

① All T001, T002 and T003 Transition Coils are OFF.

No evolution occurs.

② T001 and T002 Transition Coils are OFF, T003 Transition Coil is ON. S002 checks Counter Transition T003 since T002 Transition Coil is OFF. T003 Transition Coil is ON, but S001 at the destination is active. S002, therefore, does not proceed.

③ T001 and T003 Transition Coils are OFF, T002 Transition Coil is ON. S002 proceeds to S004 since T002 Transition Coil is ON and S004 at the destination is inactive. S001 does not proceed since T001 Transition Coil is OFF.

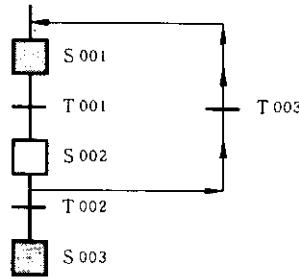


Fig. 6.95 Evolution in Loop Connection (2)

④ T001 Transition Coil is OFF, T002 and T003 Transition Coils are ON. S002 proceeds to S003 since T002 Transition Coil is ON and S003 at the destination is inactive. T003 Transition Condition Circuit is not solved at this time.

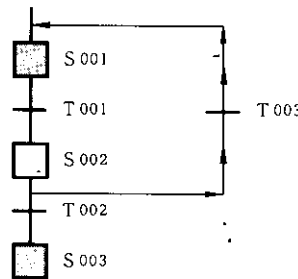


Fig. 6.96 Evolution in Loop Connection (3)

⑤ T001 Transition Coil is ON, T002 and T003 Transition coils are OFF. S002 does not proceed since both T002 and T003 Transition Coils are OFF. S001 does not proceed since S002 at the destination is active even though T001 Transition Coil is ON.

⑥ T001 and T003 Transition Coils are ON, T002 Transition Coil is OFF. Neither S001 nor S002 proceed since the steps at the destinations are active.

⑦ T001 and T002 Transition Coils are ON, T003 Transition Coil is OFF. The evolution method differs depending on the solving sequence of active steps.

- Solving S001 and then S002

In the first scan, S001 does not proceed since S002 at the destination is active even though T001 Transition Coil is ON. S002 proceeds to S003 since T002 Transition Coil is ON and S003 at the destination is inactive.

In the next scan, S001 proceeds to S002 since T001 Transition Coil is ON and S002 at the destination is already inactive.

6.9.2 Evolution with Active Steps Arranged Vertically (Cont'd)

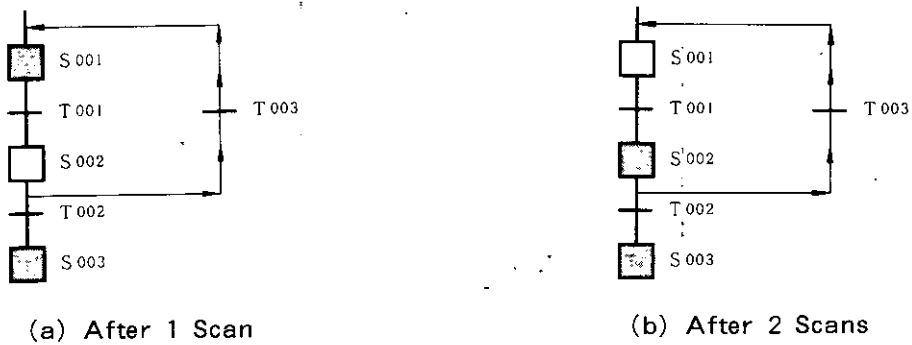


Fig. 6.97 Evolution in Loop Connection (4)

- Solving S002 and then S001

S002 proceeds to S003 since S003 at the destination is inactive and T002 Transition Coil is ON, S001 proceeds to S002 since T001 Transition Coil is switched ON and S002 at the destination is already inactive.

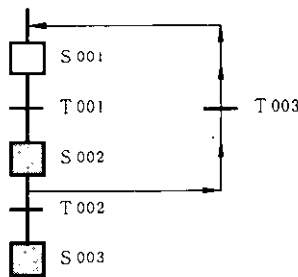


Fig. 6.98 Evolution in Loop Connection (5)

- ⑧ All T001, T002 and T003 Transition Coils are ON. Same as in ⑦.

Table 6.12 summarizes the above situations.

Table 6.12 Evolution in Loop Connection

Transition Coil State			Active Step		Solving Sequence
T001	T002	T003	After 1 Scan	After 2 Scans	
OFF	OFF	OFF	S001, S002		
		ON	S001, S002		
	ON	OFF	S001, S003		
		ON	S001, S003		
ON	OFF	OFF	S001, S002		
		ON	S001, S002		
	ON	---	S001, S003	S002, S003	S001→S002
			S002, S003		S002→S001

(9) Evolution in Loop Connection (2)

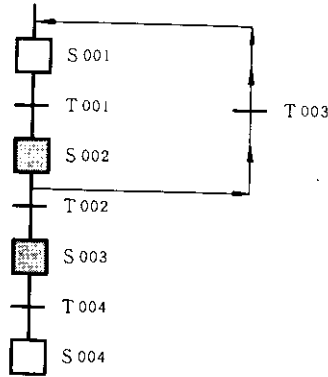


Fig. 6.99 Evolution in Loop Connection (6)

In a scan in the situation shown in Fig. 6.99:

- ① All T002, T003 and T004 Transition Coils are OFF. No evolution occurs.
- ② T002 and T003 Transition Coils are OFF, T004 Transition Coil is ON. Only S003 proceeds to S004.

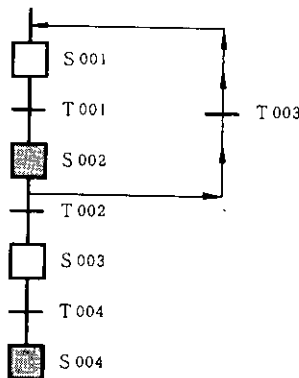


Fig. 6.100 Evolution in Loop Connection (7)

- ③ T002 and T004 Transition Coils are OFF, T003 Transition Coil is ON. S002 solves Counter Transition T003 since T002 Transition Coil is OFF and proceeds to loop destination S001 since T003 Transition Coil is ON. S003 does not proceed since T004 Transition Coil is OFF.

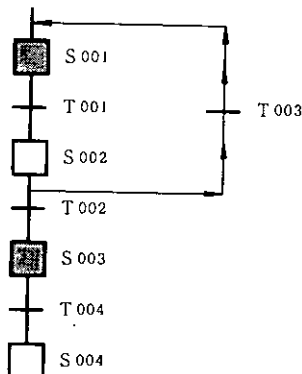


Fig. 6.101 Evolution in Loop Connection (8)

6.9.2 Evolution with Active Steps Arranged Vertically (Cont'd)

④ T002 Transition Coil is OFF, T003 and T004 Transition Coils are ON. S002 solves Counter Transition T003 since T002 Transition Coil is OFF and proceeds to loop destination S001 since T003 Transition Coil is ON. S003 proceeds to S004 since T004 Transition Coil is ON.

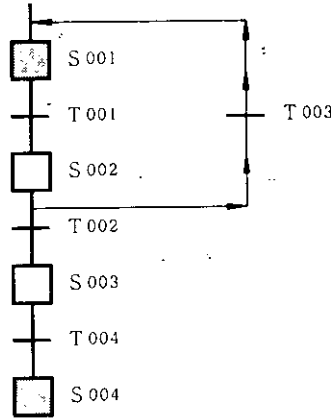


Fig. 6.102 Evolution in Loop Connection (9)

⑤ T002 Transition Coil is ON, T003 and T004 Transition Coils are OFF. S002 does not proceed since S003 at the destination is active even though T002 Transition Coil is ON, S003 does not proceed since T004 Transition Coil is OFF.

⑥ T002 and T004 Transition Coils are ON, T003 Transition Coil is OFF. The following two evolution types are available depending on the solving sequence of active steps

- Solving S002 and then S003

In the first scan, S002 solves Counter Transition T003 since S003 at the destination is active even though T002 Transition Coil is ON, but does not proceed since T003 Transition Coil is inactive. S003 proceeds to S004 since T004 Transition Coil is switched ON.

During the next scan, S002 proceeds to S003 since T002 Transition Coil is ON and S003 at the destination is already inactive.

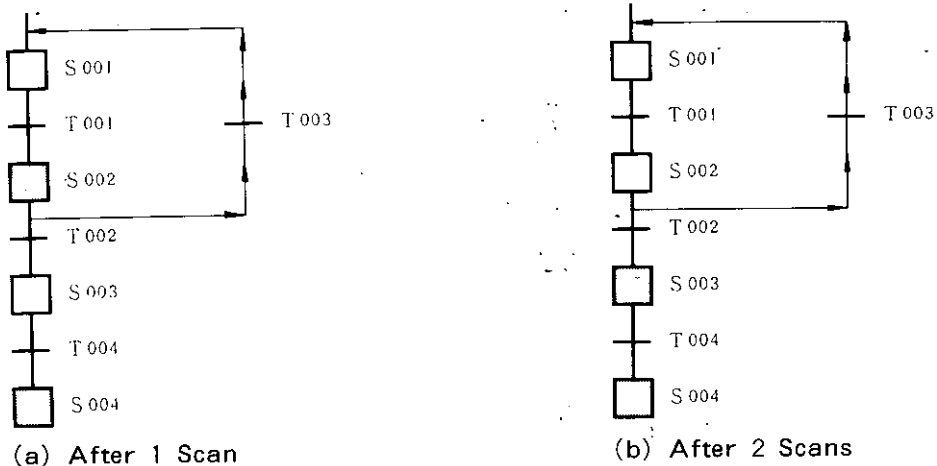


Fig. 6.103 Evolution in Loop Connection (10)

- Solving S003 and then S002

S003 proceeds to S004 since T004 Transition Coil is switched ON. S002 proceeds to S003 since T002 Transition Coil is ON and S003 at the destination is already inactive.

Unlike solving S002 and then S003, this solving sequence completes evolution in one scan.

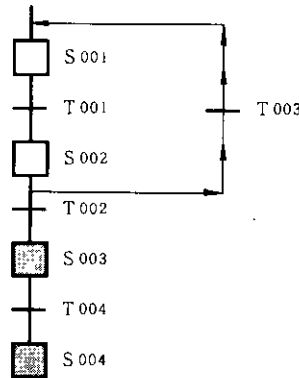


Fig. 6.104 Evolution in Loop Connection (11)

- ⑦ T002 and T003 Transition Coils are ON, T004 Transition Coil is OFF.

S002 solves Counter Transition T003 since S003 at the destination is active even though T002 Transition Coil is ON, but proceeds to S001 since T003 Transition Coil is switched ON,

S003 does not proceed since T004 Transition Coil is OFF.

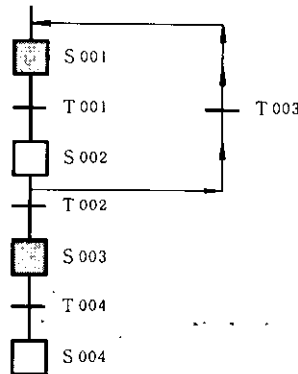


Fig. 6.105 Evolution in Loop Connection (12)

- ⑧ All T002, T003 and T004 Transition Coils are ON.

The following two evolution types are available depending on the solving sequence of active steps.

- Solving S002 and then S003

S002 solves Counter Transition T003 since S003 at the destination is active even though T002 Transition Coil is switched ON, but proceeds to S001 since T003 Transition Coil is switched ON. S003 proceeds to S004 since T004 Transition Coil is switched ON.

6.9.2 Evolution with Active Steps Arranged Vertically (Cont'd)

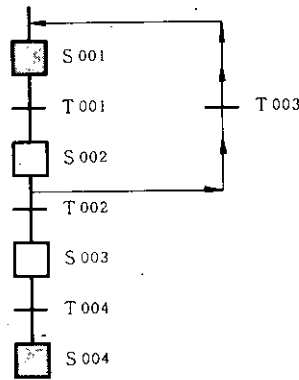


Fig. 6.106 Evolution in Loop Connection (13)

- Solving S003 and then S002

S003 proceeds to S004 since T004 Transition Coil is switched ON, S002 proceeds to S003 since T002 Transition Coil is switched ON and S003 at the destination is already inactive.

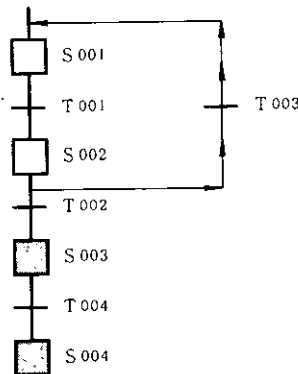


Fig. 6.107 Evolution in Loop Connection (14)

Table 6.13 summarizes the above situations.

Table 6.13 Evolution in Loop Connection

Transition Coil State			Active Step		Solving Sequence	
T002	T003	T004	After 1 Scan	After 2 Scans		
OFF	OFF	OFF	S002, S003		---	
		ON	S002, S004			
	ON	OFF	S001, S003			
		ON	S001, S004			
ON	OFF	OFF	S002, S003			
		ON	S002, S004	S003, S004	S002→S003	
	ON	OFF	OFF	S003, S004		S003→S002
			ON	S001, S003		---
		ON	OFF	S001, S004		S002→S003
			ON	S003, S004		S003→S002

(10) Macro Step Evolution

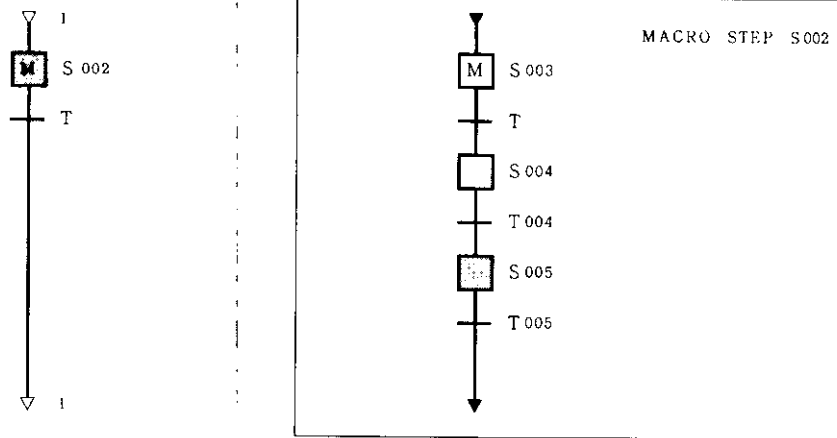


Fig. 6.108 Macro Step Evolution (1)

In an SFC flow shown in Fig. 6.108, active S005 does not proceed since Macro Step S002 at the destination is active even though T005 Transition Coil is switched ON

One precaution about macro steps is that macro steps only connect master and expanded views, but prohibit any active steps in expanded views to proceed to them. Therefore, when creating an SFC flow as shown in Fig. 6.108, the system will not have evolution unless a transition (which has a transition condition circuit whose transition coil is always switched ON) and a dummy step (wait step without an action circuit) are put below Macro Step S002, or Step S005 and Transition T005 are shifted to the master view. Fig. 6.109 shows the SFC flow which has S005 and T005 shifted to the master view.

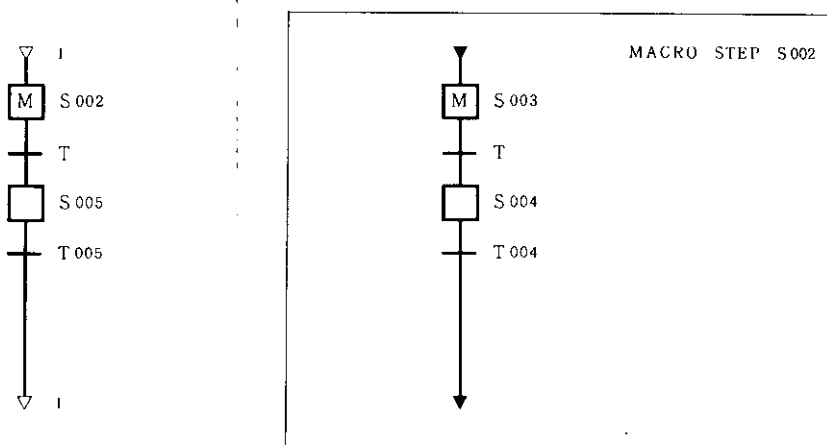


Fig. 6.109 Macro Step Evolution (2)

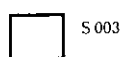
Control delays by one scan if a dummy step and a transition are put in the master view. The SFC flow shown in Fig. 6.109 performs exactly the same control as that of the SFC flow shown in Fig. 6.108.

6.10 SFC SIMULATED OPERATION FUNCTIONS

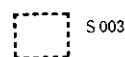
The following two simulation operation functions are provided for use during SFC program creation. SFC programs can be easily debugged by using them skillfully.

6.10.1 Disable Function

If a step at the destination is disabled during program solving, even though conditions for evolution are established, evolution does not occur and the top step is held until disable is released. This function is used if a step is not desired to be converted active during program creation, or in other cases. Disable can be designated through the P150 programming panel. Disabled steps are represented in views as shown below.



(a) Normal State



(b) Disabled State

A disabled step below a divergence connection does not proceed even if the transition coil of the transition above the step is switched ON. It proceeds to a step below which is switched ON by another transition coil.

A disabled step below a simultaneous divergence connection does not proceed even if the transition coil of the transition above is switched ON. Simultaneous divergence proceeds only if the transition coil is switched ON and none of the steps below are disabled.

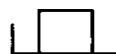
A disabled step below a loop connection does not proceed downward even if the transition coil of the transition above is switched ON and remains held until the transition coil of a counter transition is switched ON. It proceeds around the loop when the transition coil of the counter transition switches ON. In loop connection, a step continues to go around the loop until the disable state is released. (It is sometimes held depending on the transition condition circuit of the counter transition.)

6.10.2 Hold Function

Steps in a hold state are put in a perfect hold state, and their action circuits are solved. However, transition condition circuits of transitions connected below are not solved. Therefore, counting up is not performed until the hold state is released even if timers and counters are used in the transition condition circuits of the transitions in a hold state below. Steps in a hold state are represented on the P150 programming panel as shown below.



(a) Normal State



(b) Hold State

When a macro step is put in a hold state, the hold state is applied to the step immediately before a macro return, and the transition is solved in this case. Proceeding to the next step is prevented if a macro step is put in a hold state.

SECTION 7

I/O ALLOCATION

Since I/O module can be located at any module slot corresponding to I/O allocation, a variety of combination of I/O modules is available. Before operating GL60H or GL70H Controller, be sure to set I/O allocation table to the CPU module memory using the P150 programming panel.

I/O allocation is made independently to each location. A change of I/O allocation made to a location does not affect those for the other locations.

7.1 I/O CONFIGURATION

As shown in Fig. 7.1, the I/O section of CPU is composed of three channels. The maximum number of I/O modules that can be mounted is 256 each for discrete input, discrete output, register input and register output, for a total of 1024 modules maximum.

A total of 4096 discrete I/O points and a total of 512 sets of register I/O can be mounted in any location of the I/O modules.

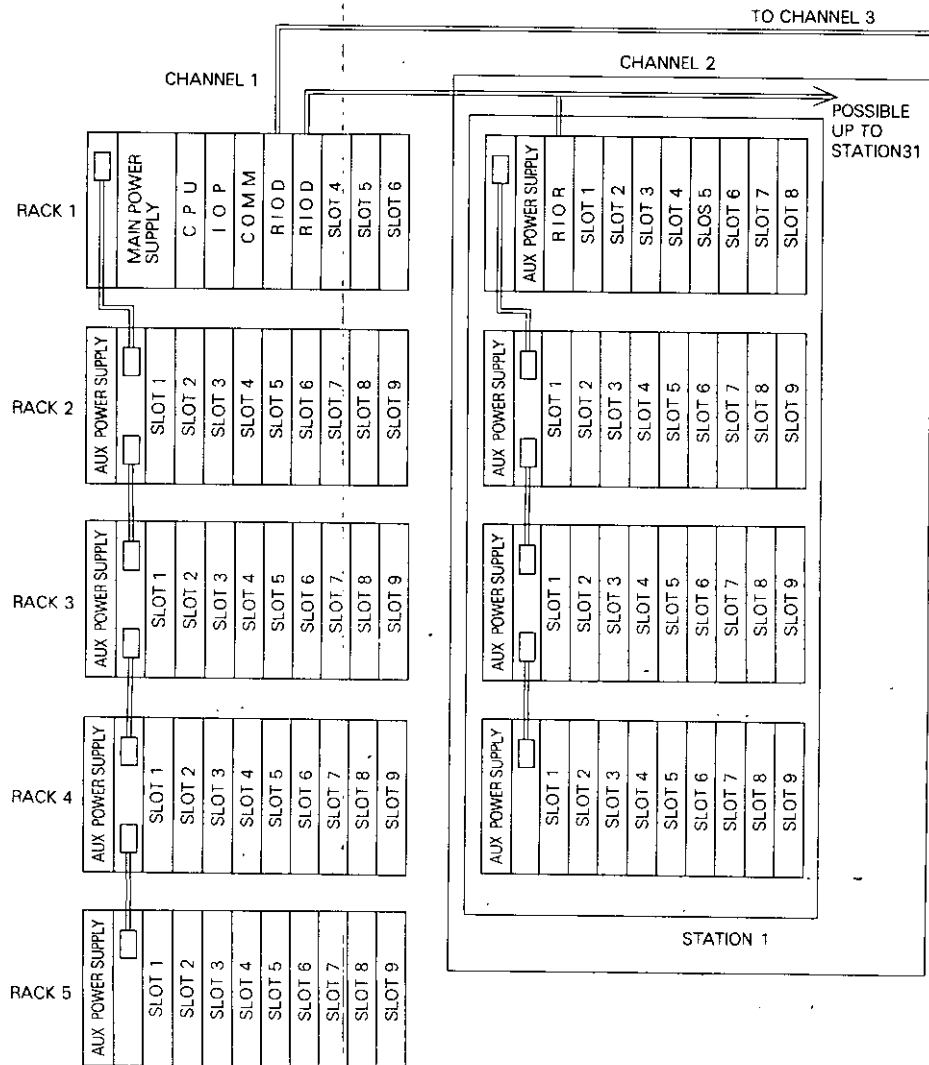


Fig. 7.1 I/O Section Configuration

7.2 I/O MODULE LAYOUT

I/O modules may be installed at any location of channels 1,2 and 3 only in a range of 4096 discrete I/O points and 512 register I/O points. They need not be installed in contiguous locations. However, it is recommended that the I/O modules be installed in groups (by input and output, voltage level, application, etc.). Fig. 7.2 shows a sample layout of I/O modules.

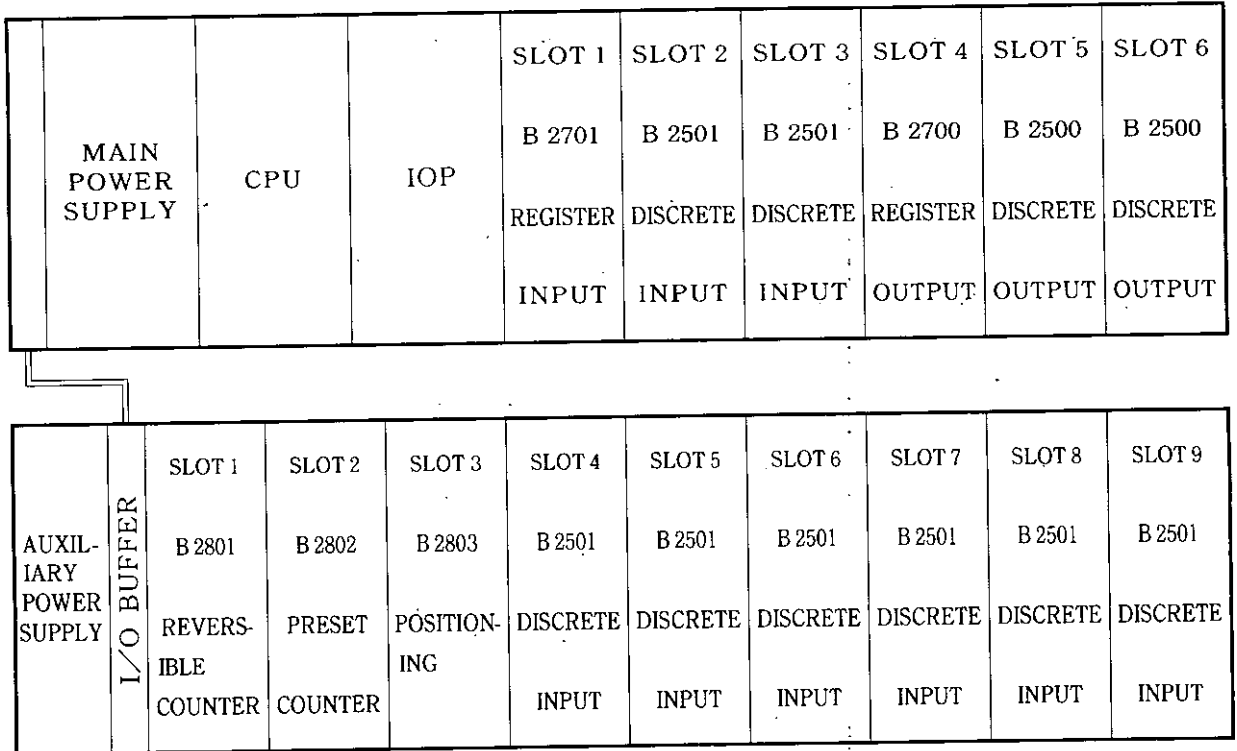


Fig. 7.2 Sample Layout of I/O Modules

7.3 I/O NUMBERS

I/O signals include discrete inputs/outputs and register inputs/outputs (numerical value). The reference number is used as the I/O number.

Table 7.1 I/O Number List

Input/Output Type	I/O Number (Reference Number)
Discrete Input (Input Relay)	10001 - 14096
Register Input (Input Register)	30001 - 30512
Discrete Output (Output Coil)	00001 - 04096
Register Output (Output Register)	40001 - 40512

NOTE The table above shows the range of I/O numbers as signable to each group of I/O signals. Note the following limitations:

- Discrete inputs + discrete outputs \leq 4096
- Register inputs + discrete outputs \leq 512

7.4 I/O MODULE LOCATION

The location of an I/O module is defined by a channel number, a rack number, and a slot number. Number of slots differs depending on channels, stations and racks as shown in Table 7.2.

Table 7.2 Number of Slots at Each Channel, Station and Rack

CH No.	RACK No.	1	2	3	4	5	Name
		1	6	9	9	9	
2	Station 1	8	9	9	9	—	Remote 1
	Station 2	8	9	9	9	—	
	⋮	⋮	⋮	⋮	⋮	⋮	
	Station 31	8	9	9	9	—	
3	Station 1	8	9	9	9	—	Remote 2
	Station 2	8	9	9	9	—	
	⋮	⋮	⋮	⋮	⋮	⋮	
	Station 31	8	9	9	9	—	

NOTE Number of slots to be allocated is a maximum of 256 slots for each of discrete I/O and register I/O.

It is permissible to assign a pair of discrete input and output or a pair of register input and output to a slot. This is possible owing to the modules, such as the counter, PID, and positioning module, each of which deals with discrete inputs/outputs and register inputs/outputs, (called a modular module). The number of I/O points allocated to a slot is given in Table 7.3.

Table 7.3 Number of Input/Output Points Allocated to a Slot

Input/Output Type	Allowable I/O Points in Allocation
Discrete Input	8, 16, 24, 32, 64
Discrete Output	8, 16, 24, 32, 64
Register Input	1 - 8
Register Output	1 - 8

- NOTE**
- Any I/O allocation is available in the range of I/O points given above. To a slot where a 16-point discrete output module is installed, for example, 8 or 16 discrete outputs must be allocated.
If 24 or 32 points are allocated to the slot (it is possible through the P190 programming panel), the outputs may be degraded.
 - Up to 128 discrete I/O points can be allocated in units of 8 points for future expansion. At present, however, only B2804 and B2805 can deal with more than 64 discrete I/Os.
 - External data is stored as is in input registers as register input. If input as BCD, internal processing does not perform correct arithmetic operations unless binary conversion is conducted using the arithmetic function BIN. Similarly, register output must also be output by BCD after converting it into BCD using the arithmetic function BCD.

7

The following conditions are established with the number of points allocated to each station, assuming the numbers of discrete I/O points and register I/O points to be DI, DO, RI and RO.

$$1 \text{ Input: } \frac{DI}{8} + 2RI \leq 512 \quad 2 \text{ Output: } \frac{DO}{8} + 2RO \leq 512$$

7.5 TYPES OF I/O MODULES AND I/O ALLOCATION

The maximum number of I/O points for allocation is specified in accordance with the type of I/O module as shown in Table 7.4. Refer to Table 7.4 and the followings.

- Registers can be allocated, in 16-bit binary forms, to a discrete module.
- I/O allocation for the register module can not be performed in discrete form. Register module I/O data is given in 16-bit binary form.
- Analog modules can not be allocated in discrete form.
- Both discrete and register points must be allocated to counters and positioning modules.
- Allocation is not necessary for the power supply module because it does not deal with I/O signals.

Table 7.4 Number of I/O Points for Allocation by Module Type

	Modules	Type
Input	16-point Discrete	B2501, B2503, B2601
	32-point Discrete	B2505, B2507, B2603, B2607
	Register	B2701
	Analog	B2703
Output	16-point Discrete	B2500, B2600, B2900, B2904
	32-point Discrete	B2504, B2602, B2606, B2902
	Register	B2700
	Analog	B2702
	Reversible Counter	B2801
	Preset Counter	B2802
	Positioning	B2803, B2813

7.6 I/O ALLOCATION REFERENCE NO.

The reference number should be the first number allocated to the slot. For discrete I/O allocation, the reference numbers must begin with fixed numbers as follows.

- First number of discrete output = $00001 + 8n$ ($n=0, 1, 2, \dots, 511$)
- First number of discrete input = $10001 + 8n$ ($n=0, 1, 2, \dots, 511$)

The following limitations exist in relation to the range of the GL60H or GL70H reference numbers.

- First number of discrete output + number of points ≤ 4097
- First number of discrete input + number of points ≤ 14097
- First number of register input + number of points ≤ 30513
- First number of register output + number of points ≤ 40513

7.7 HIGH-SPEED STATION

When two-level scan is specified, the solving of programs and the I/O processing are as follows;

- (1) SFC & high-speed segment programs are solved every high-speed scan.
- (2) Local I/O processing is done every high-speed scan.
- (3) Remote I/O processing is done every low-speed scan.

Thus, local I/O data are updated every high-speed scan in SFC & high speed segment programs.

However, remote I/O data are not updated every high-speed scan in these programs. They are updated every low-speed scan.

If some remote I/O data must be updated every high-speed scan, the remote I/O station must be specified as a high-speed station. By specifying as a high-speed station, high-speed I/O processing is executed with the station every high-speed scan. The reference data allocated to the station will be updated every high-speed scan.

A maximum of eight stations can be specified per remote channel (channels 2 and 3) as high-speed stations. The reference Nos. allocated to the high-speed stations must be sequential in the station.

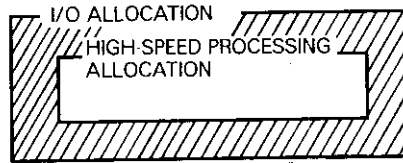
If one-level scanning is selected as the scan level, all I/O processing is performed on each scan. High-speed station allocation is not needed, and only normal I/O allocation must be set.

7.8 I/O ALLOCATION AND HIGH-SPEED STATION

No problem is caused if normal I/O allocation reference and high-speed processing allocation reference match perfectly in each station. If mismatched, the following will result.

7.8 I/O ALLOCATION AND HIGH-SPEED STATION (Cont'd)

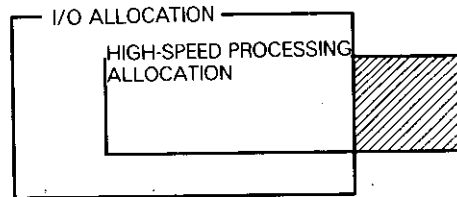
- (1) If I/O allocation reference inside each station is larger than high-speed processing allocation reference



In this case, I/O processing is performed during each high-speed scan, and the I/O side sends I/O data to the CPU on each high-speed scan. The CPU updates only data which is allocated for high-speed processing during high-speed scans. Thus, the input data in the shaded part is neither reflected on the ladder circuit, etc. of the high-speed segment nor output. Therefore, this allocation results in execution of wasteful I/O processing.

No problem will be caused in I/O processing or in solving if I/O allocation and high-speed station data are mismatched once this is recognized.

- (2) If allocation of high-speed station is spreading to I/O allocation range in each station



In this case, the shaded part is not allocated for I/O, and the I/O section does not send I/O data of this part to the CPU. The CPU updates it as 0 (OFF for coils), and solving is performed with incorrect data.

High-speed station allocation by this method must be avoided.

SECTION 8 GL60H, GL70H APPLICATIONS

8.1 NOTES ON APPLICATIONS

Memocon-SC GL60H or GL70H should be used to meet your system specifications paying attention to the following points.

8.1.1 Backup Circuit

Since GL60H or GL70H is more reliable than relays, and also their eventual faults can be repaired quickly, it requires no backup circuit in ordinary systems. However, when it does require a backup circuit because of the special nature of the system, the selection of a proper backup method is an important consideration. An external manual circuit and standby GL60H or GL70H are sound methods.

8.1.2 Interlock

The GL60H or GL70H CPU is provided with a self-diagnosis function of stopping operation and turning output OFF when the stored ladder circuits (program) are destroyed, or when module cards develop faults. However, some faults and misoperations are not detected, and may destroy machines and devices. Where there is a possibility of such destruction, start and stop the system under redundant control such as external interlock and electric-mechanical redundant control.

8.1.3 Control Panel Layout

Lay out the control panel locations in consideration of the electrical environment conditions. For details, refer to Par. 8.3 "CONSTRUCTION, INSTALLATION AND WIRING OF CONTROL PANEL."

8.1.4 Local Station Network Processing

The network processing method of GL60H or GL70H has been explained in this manual. However, for reading the ON and OFF states of all input signals into the CPU module of GL60H or GL70H correctly, the ON and OFF states of input signals must continue longer than the total delay time in the input module and one scan time. Therefore, when dealing with signals of short duration, special devices such as external memory circuits must be used, and for limit switch signals, the dog length must be sufficiently long. Refer to Fig. 8.1.

Output signals are also delayed up to the total of one scan time and the delay time within the output module. Therefore, for applications requiring a high degree of accuracy, some external devices are required. Refer to Fig. 8.2.

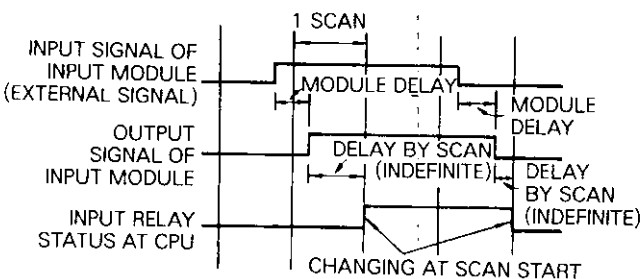


Fig. 8.1 Input Signal Delay

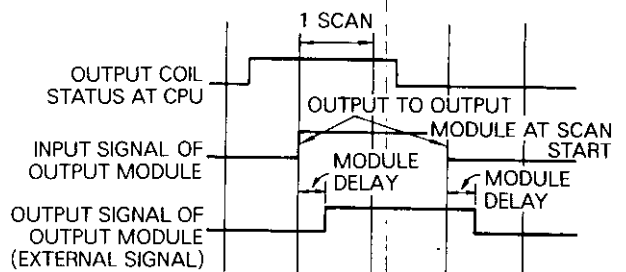


Fig. 8.2 Output Signal Delay

8:1.5 Remote Station Network Processing

I/O processing of a remote station is performed as follows.

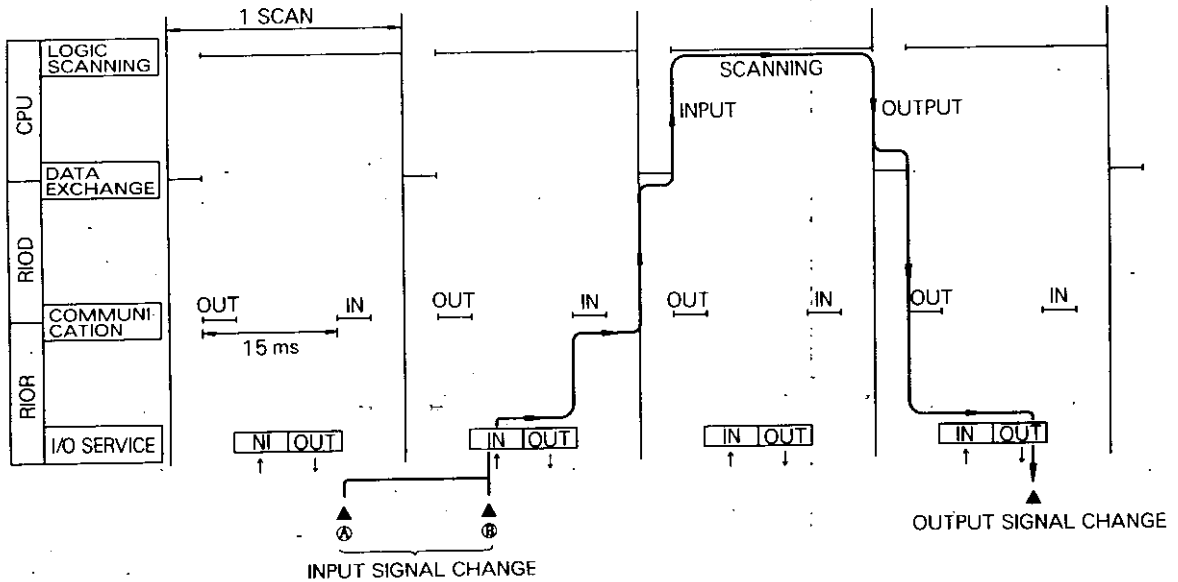


Fig. 8.3 Remote I/O Processing

If the input signal changes at Point (B), the output signal to this data is output after $(2 + \alpha)$ scans (α changes in accordance with I/O allocation.) If the input signal changes at Point (A), approximately three scans are needed before the output signal is output.

8.2 CALCULATION OF MEMORY CAPACITY

To find the required number of memories is a difficult task in composing any system. Exact numbers can only be determined from the intended ladder circuits, but here, the memory capacity of GL60H or GL70H may be roughly found, as follows. Although more memories are required when the number of I/O signals increases, and when the sequence becomes more complex (more complex control),

$$40 \times \text{Number of Output Coils}$$

is taken as a rough guideline regarding the number of memories (words).

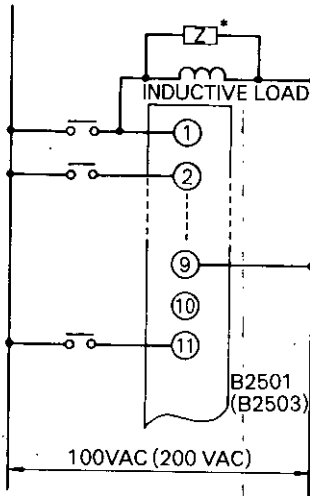
Additional sequences may become required during trial and adjustment operations, and for this reason, some reserve memories should be prepared from the beginning.

8.3 PRECAUTIONS FOR USING I/O MODULES

8.3.1 Input Module

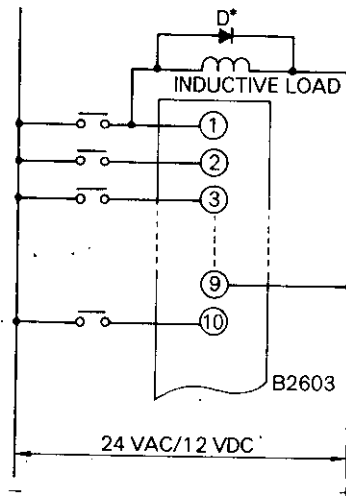
(1) Inductive Load

Where an inductive load is connected in parallel with the input module as shown in Figs. 8.4 and 8.5, connect a surge absorber or flywheel diode in parallel with the inductive load, respectively for the AC input module and DC input module.



*The surge absorber capacity should be selected corresponding to the load. It is recommended that type CR 50500 (made by Okaya Electric Industries Co.) or equivalent be used.

Fig. 8.4 AC Input Module



*The flywheel diode should be selected corresponding to the load. It is recommended that type F14 series (made by NEC) or equivalent be used.

Fig. 8.5 DC Input Module

(2) Input Dummy Resistor

Where the external wiring is long or where there is an induction source in the vicinity, connect a dummy resistor in parallel to the input module, as shown in Fig. 8.6.

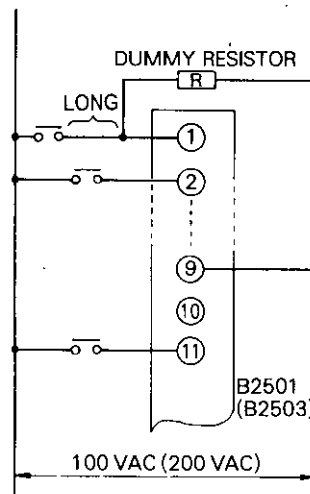


Fig. 8.6 AC Input Module

R : INPUT DUMMY RESISTOR
 B 2501 : 5 k Ω (W \geq 10 W)
 B 2503 : 10 k Ω (W \geq 20 W)

(3) Leakage Current in Input Equipment

Some input equipment (e.g. noncontact switches and limit switches with LED) has leakage current during the OFF state. If this equipment is connected to AC input modules, it may fail to maintain the voltage for an OFF condition which is an input due to leakage current, and input signals may not be cut off.

(Example) A non-contact switch with 5 mA of leakage current is connected to B2501.

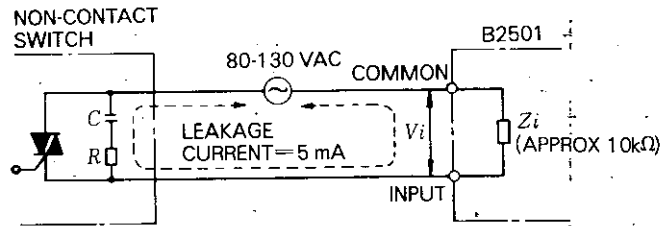


Fig. 8.7 Connection of a Non-contact Switch

If the leakage current is 5 mA, the input voltage (V_i) of B2501 becomes;

$$V_i = 5mA \times Z_i = 5mA \times 10k\Omega = 50V$$

Since this does not satisfy an input condition (OFF voltage = 30 V or less), input signals may not be cut off. In this case, add a proper dummy resistor to the input terminal of B2501.

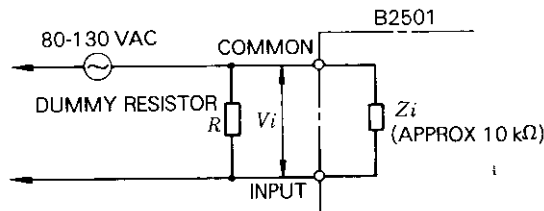


Fig. 8.8 Addition of a Dummy Resistor

The value of the dummy resistor R should be decided so that an input voltage V_i of B2501 becomes 30 V or less.

$$\frac{R \times Z_i}{R + Z_i} \times \text{leakage current} < 30 V$$

$$\frac{R \times 10k\Omega}{R + 10k\Omega} \times 5mA < 30 V \quad \therefore R < 15k\Omega$$

Thus, the value of R becomes 15 kΩ or less. However, if the value is too small, heating value increases, resulting in the need of a resistor with large wattage.

Assume that the value of R is 10 kΩ. Then the wattage W of the dummy resistor becomes;

$$W = \frac{(\text{power source})^2}{R} = \frac{(100 V)^2}{10k\Omega} = 1 W \quad \therefore W = 1 W$$

Generally, the wattage W of the dummy resistor is taken to be 3W to provide a surplus wattage about three times more than required.

(4) ON/OFF Conditions of DC Input Module

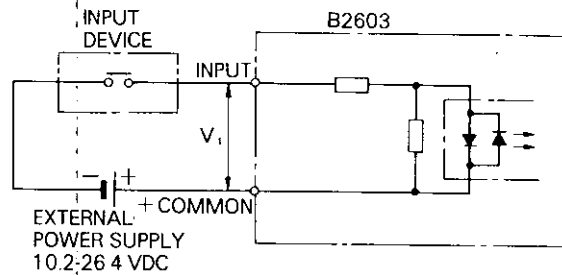


Fig. 8.9 ON/OFF Conditions of B2603

Input conditions of B2603 are;
 ON level: 9 VDC or more
 OFF level: 6 VDC or less

(Example) When a limit switch with LED is connected to B2603

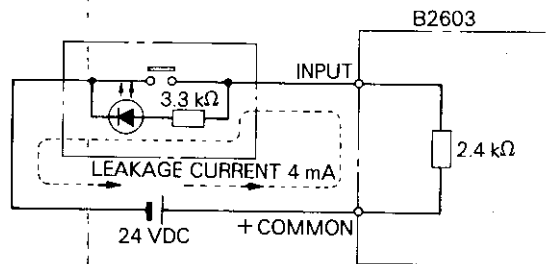


Fig. 8.10 Connection of a Limit Switch with LED

If the leakage current is 4 mA, then

$$V_i = 2.4 \text{ k}\Omega \times 4 \text{ mA} = 9.6 \text{ V}$$

This does not satisfy the input condition (OFF level = 6 V or less). Therefore, add a proper dummy resistor to the input terminal of B2603 so that the input condition is satisfied.

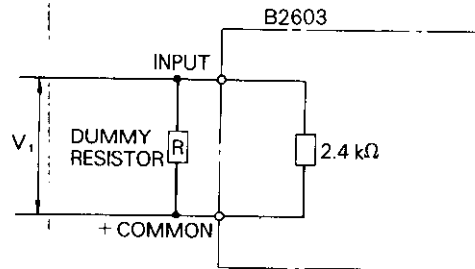


Fig. 8.11 Addition of a Dummy Resistor

The value of a dummy resistor R should be chosen such that the input voltage V_i of B2603 becomes 6 V or less.

$$4 \text{ mA} \times \frac{2.4 \text{ k}\Omega \times R}{2.4 \text{ k}\Omega + R} < 6 \text{ V} \quad \therefore R < 4 \text{ k}\Omega$$

Thus, the value of the resistor becomes 3 kΩ.

Necessary wattage W is;

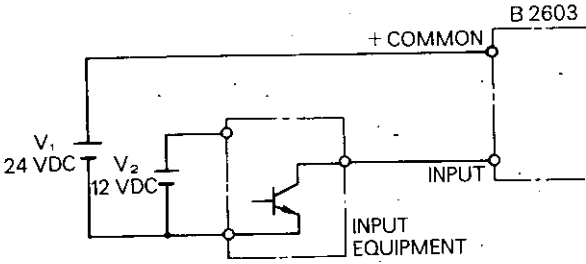
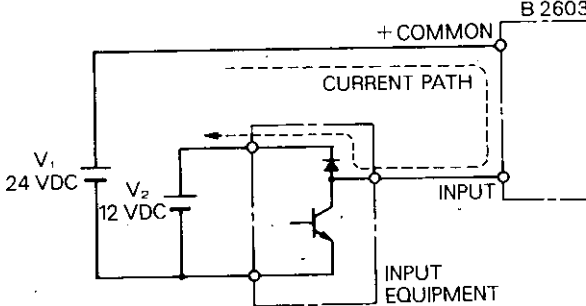
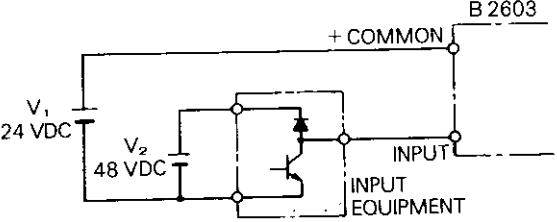
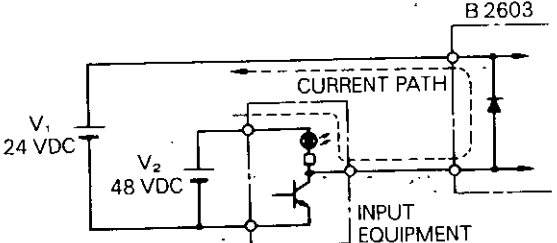
$$W = \frac{(\text{power supply voltage})^2}{R} = \frac{(24 \text{ V})^2}{3 \text{ k}\Omega} = \text{about } 200 \text{ mW}$$

In general, the wattage of a dummy resistor is taken to be 0.5-1 W to provide a surplus wattage about three times more than required.

(5) Connection to Input Equipment with Different Voltage

Usually, power voltage of input equipment should be matched that of input modules. However, Table 8.1 shows possibilities of connecting input equipment having different voltages.

Table 8.1 Possibilities of Connecting Input Equipment Having Different Voltages

Example of Input Equipment	Connection Possibilities
<p>① Open collector output ($V_1 > V_2$)</p> 	<p>Can be connected. However, the voltage resistance of the output transistor of the input equipment should be 40 V or more.</p>
<p>② With resistor, LED or diode ($V_1 > V_2$)</p> 	<p>Cannot be connected. When the input equipment is OFF, current shown by a dotted line in the left figure may flow and input does not become OFF. Especially, in case of LED, reverse voltage may be applied during the OFF time to the equipment with LED and the LED may be broken.</p>
<p>③ With open collector or diode ($V_1 < V_2$)</p> 	<p>Can be connected.</p>
<p>④ With resistor or LED ($V_1 < V_2$)</p> 	<p>Cannot be connected. When the input equipment is OFF, current shown by a dotted line in the left figure may flow and the LED of the input equipment comes on dimly.</p>

(6) Caution in Using B2603

Ambient temperature of B2603 (32-point) input modules depends on the external power supply voltage and the number of input points that are ON at the same time. Adjust the ambient temperature within a value shown in Fig. 8.12.

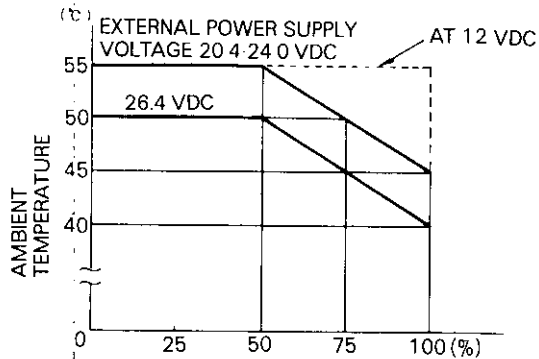
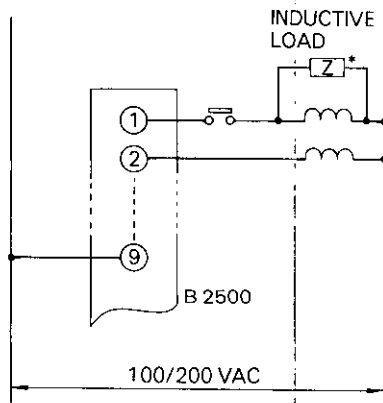


Fig. 8.12 Adjustment of Ambient Temperature

8.3.2 Output Module

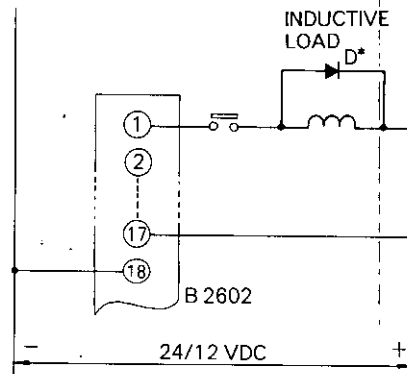
(1) Connection to Contacts

When connecting contacts to an inductive load of the output module, as shown in Figs. 8.13 and 8.14, always connect a surge absorber or a flywheel diode in parallel to the inductive load.



*The surge absorber capacity should be selected corresponding to the load. It is recommended that type CR 50500, (made by Okaya Electric Industries Co.) or equivalent be used.

Fig. 8.13 AC Output Module



*The flywheel diode should be selected corresponding to the load. It is recommended that type F14 series (made by NEC) or equivalent be used.

Fig. 8.14 DC Output Module

(2) Minimum Load Current

As the output switch of the AC output module, a triac is used. Since a triac cannot operate stably if the load is less than the specified minimum load current, make sure to use the load which is secure current levels above the minimum load current. If the minimum load current cannot be kept, connect a dummy resistor in parallel to the load so that the total load current is above the minimum load current.

(3) Maximum Load Current

Although an output point can accommodate a 1 A load, the total load for 8 output points must be up to 3 A. This should be taken into consideration for distributing loads.

(4) Output Fuse

The output fuse is used for preventing the trouble caused by shortcircuit of the load, but not for protecting the output element of the module.

(5) Status LED Indicator for AC Output Module

The status LED indicator for the AC output module lights up by power supply for the internal logic circuit.

(6) Leakage Current from the Output Module

AC output module and relay contact output module contain a surge suppressing circuit. Therefore, leakage current flows during OFF.

Table 8.2 Leakage Current in Output Modules

Output Module Type JAMSC-	Output Impedance during OFF (50 Hz)	Maximum Leakage Current
B2500	Approx. 68 kΩ	Approx. 2 mA at 130 VAC

When a light-load relay is connected to these output modules, the relay does not turn off due to the current.

(Example) When load impedance is 6 kΩ and the load responds incorrectly due to 1 mA of leakage current.

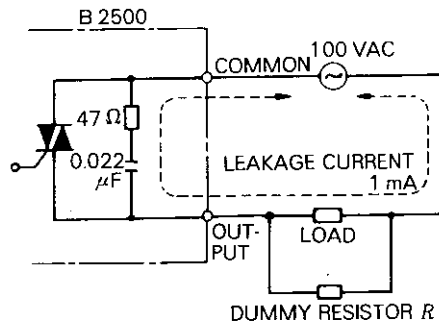


Fig. 8.15 Connection of Light Load

If 0.5 mA or less of current flow in the load does not cause any malfunction, then the value of dummy resistor R becomes;

$$1 \text{ mA} \times \frac{R}{R + 6 \text{ k}\Omega} < 0.5 \text{ mA}$$

$$R < 6 \text{ k}\Omega$$

Thus, make the value of R 6 kΩ.

Necessary wattage W is;

$$W = \frac{(\text{power source voltage})^2}{R} = \frac{(100 \text{ V})^2}{6 \text{ k}\Omega} = 1.7 \text{ W}$$

Generally, the wattage of the dummy resistor is taken to be 5 W to provide surplus wattage about 3 times more than required.

(7) Connection of Solenoid with Diode

Solenoids with diodes have the advantage being driven by half-wave rectification and less starting current. When solenoids with diodes are used as load of AC output module, be careful of the following points.

① When output is OFF, overvoltage is applied to load:

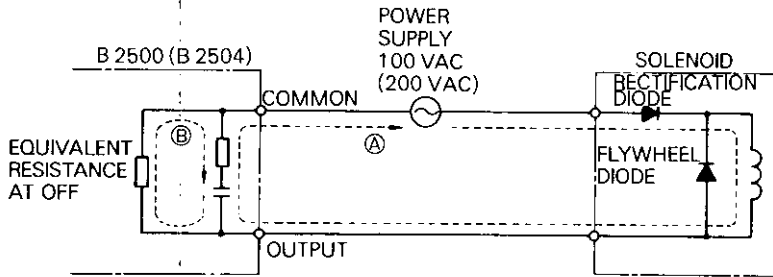


Fig. 8.16 Connection of Solenoid with Diode

When output of AC output module is OFF, current A, shown by a dotted line, flows at half-cycle of the power supply to be a forward-biased rectification diode, and is charged on capacitors. See Fig. 8.16.

With next half-wave, after polarity is reversed, rectification diode is reverse-biased and current A is blocked; discharge current B, shown by a dotted line, flows from the capacitor. At this time, supply voltage, and voltage charged on the capacitor are superimposed, and applied to the solenoid. The peak value of this voltage is approximately $2.2E$ (E : supply voltage). Rectification diode should require a withstand reverse voltage of $2.2E$ or more.

Connect a resistance of approximate multiples of $10\text{ k}\Omega$ to several hundred $\text{k}\Omega$ on solenoid ends so that voltage applied to the solenoid with rectification diode is reduced. See Fig. 8.17.

② When output is ON, solenoid may not turn ON:

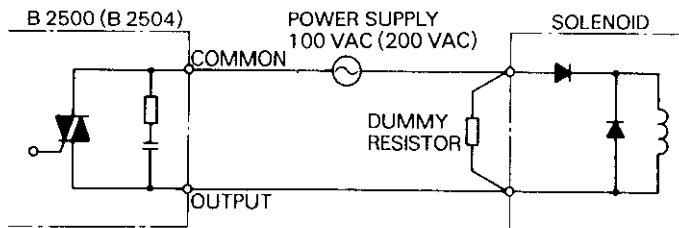


Fig. 8.17 Connection of Dummy Resistor

Where the solenoid with diode is connected, solenoid may not turn ON because voltage of output ends is not reduced to operation level by effect of voltage charged in the capacitor. Connect resistance of approximate multiples of $10\text{ k}\Omega$ to several hundred $\text{k}\Omega$ on solenoid ends so that voltage applied to solenoid is reduced. See Fig. 8.17.

(8) Connection of B2904

Bestact relay output module, B2904 has a polarity on contact output as shown in Fig. 8.18. When DC power supply is used as a load, observe the correct polarity. If using an opposite polarity, electrical life of the contact may be shortened.

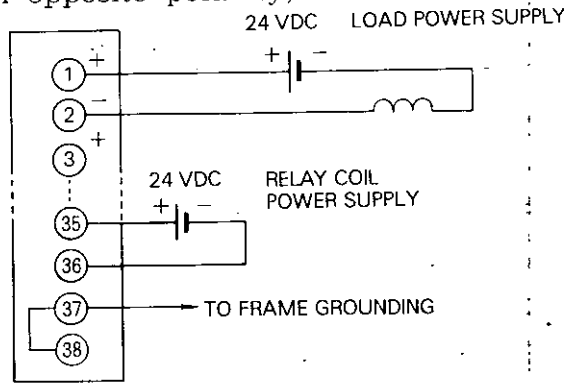


Fig. 8.18 Connection of B2904 and DC Load

8.3.3 Connection between I/O Modules

Where two or more GL60H or GL70H controllers are used in a system, and signals are exchanged between GL60H or GL70H controllers via I/O modules, connections should be as shown in Figs. 8.19 and 8.20. In this case, make sure to use modules of the same voltage rating, and connect a dummy resistor to the output module, to make stable operation.

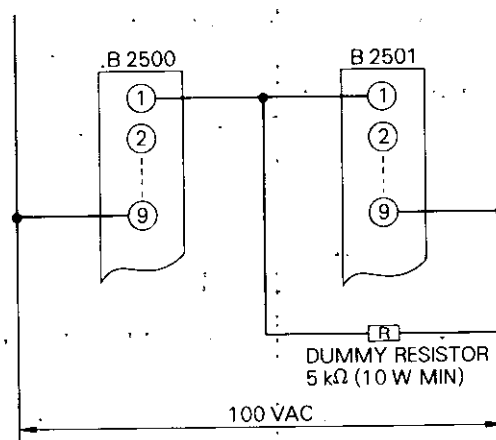


Fig. 8.19 Connection between AC I/O Modules

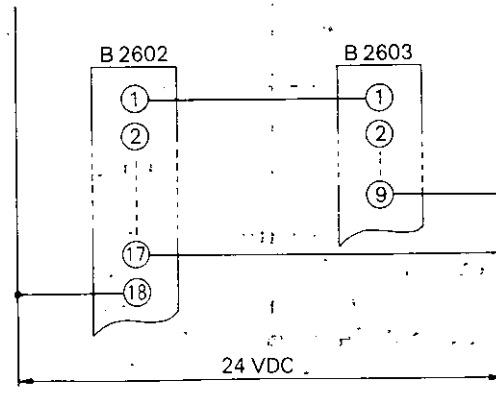


Fig. 8.20 Connection between DC I/O Modules

8.3.4 External Power Supply

General DC stabilized power supply should be used for DC I/O modules as an external power supply. Add a noise filter on the AC input side of the DC stabilized power supply for special modules such as a register module, analog module, or counter module. Do not run the primary and the secondary side of the noise filter and the DC output side in the same wire duct.

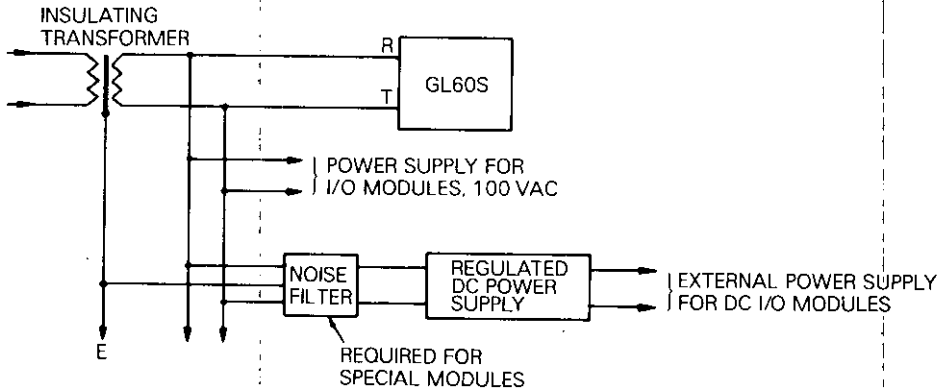


Fig. 8.21 External Power Supply for DC I/O Modules

If it is necessary to use a simple DC power supply such as full wave rectification, minimize the ripple by adding a smoothing capacitor. The following should be observed:

- Instantaneous output voltage with ripple should always be within the range of the operation voltage of the DC I/O modules.
- Output voltage, including that of the power ON time and power OFF time, should never exceed the transient voltage of the DC I/O modules.
- Prevent the introduction of surge voltage by adding a noise filter on the input side of a rectifying device.

8.3.5 Precautions when Installing I/O Module

The GL60H or GL70H system can be provided with up to 6 I/O modules on channel 1, rack 1 and up to 9 I/O modules on racks 2, 3, 4 and 5; up to 8 on channels 2 and 3, rack 1, up to 9 on racks 2, 3 and 4. However, the number of modules to be installed must be limited so that the total consumed current of the modules to be used will not exceed the capacity of the internal power (supplied from a power supply module GL60S or a module PS21). The total consumed current should be calculated in accordance with Fig. 8.22.

(1) GL60H, GL70H LOCAL STATION

(2) GL60H, GL70H REMOTE STATION

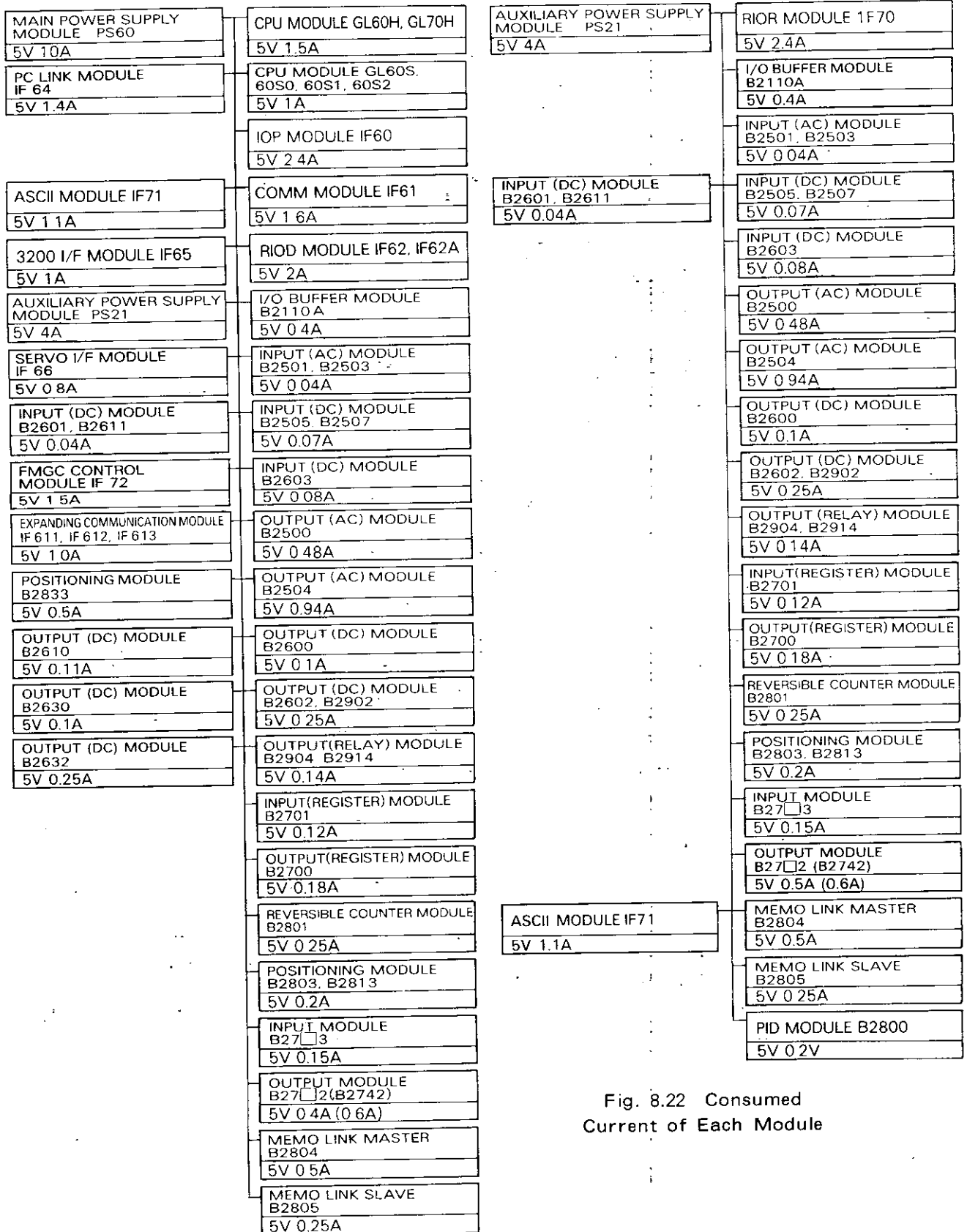


Fig. 8.22 Consumed Current of Each Module

8.4 CONSTRUCTION, INSTALLATION AND WIRING

The GL60H or GL70H systems are delivered with the CPU module, the power supply module, the input and output modules, the mounting base, the cable, etc. separated from each other.

These components must be installed in a control panel housing, the construction, layout, and wiring of which shall conform to the following standards. Where more strict conditions for the wiring, etc. are specified by separate specifications, etc., there should be given priority. For further details not specified below, applicable status or regulations apply.

8.4.1 Construction of Control Panel

For the control panel, the following construction is recommended.

- Enclosed steel housing, self-standing (or wall-mounting)
- Dustproof, or semi-dustproof
- Cooling: Where the panel-interior temperature (GL60H or GL70H ambient temperature) rises above 55°C, ceiling fan or other cooling devices must be used. A cooling fan should in principle be used to discharge air from the panel interior.

The respective modules have the heating value given in Table 8.1, the heating value for I/O modules applies when all the 16 points are simultaneously ON.

- Dimension
Determine the size, etc. of the control panel by referring to the dimension of the each unit modules (Appendix C) and the GL60S panel mounting dimension (Appendix D).
- Layout of mounting base
Install these in the relative positions as shown in Appendix D, taking the cooling and other conditions into consideration.

Table 8.3 Heating Value of Modules

Module (Type)	Heating Value W
CPU module (GL60H, GL70H)	7.5
Main power supply module (PS60)	70
Auxiliary power supply module (PS21)	40
I/O processor module (IF60)	12
COMM module (IF61)	8
Remote I/O driver (IF62, IF62A)	10
Remote I/O receiver module (IF70)	12
I/O buffer module (B2110)	2
100VAC Input module (B2501)	4
12/24VDC Input module (B2603)	10
100/200VAC Output module (B2500)	10
12/24VDC Output module (B2602)	10
Relay contact output module (B2902)	8
Relay contact output module (B2904)	8
PC link module (IF64)	7
3200 I/F module (IF65)	5
Servo I/F module (IF66)	4
ASCII module (IF71)	5.5
FMGC control module (IF72)	7.5

8.4.2 Device Configuration in Control Panel

The modules of the GL60H or GL70H are mounted on three types of mounting bases. Each module must be installed on any location of the mounting base.

When installing the GL60H or GL70H in a control panel, determine the device configuration by considering the layout of other devices and the following. In Appendix D, a sample layout of the GL60H or GL70H in a control panel is shown.

(1) Mounting Bases and I/O Cables

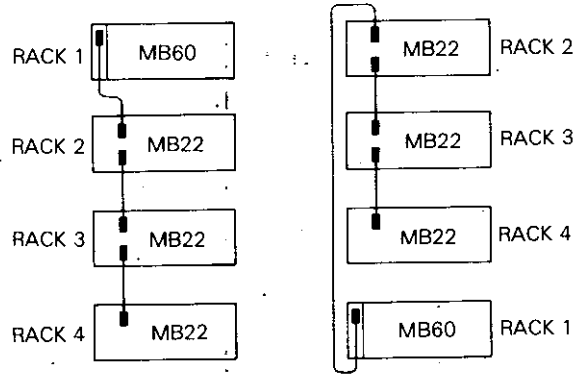


Fig. 8.23 Mounting Base Arrangement

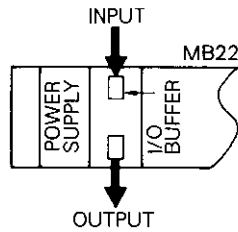
(a) Arrangement 1 (b) Arrangement 2

Mounting bases must be connected in the order shown in Fig. 8.23: Racks 1 to 5. The I/O cables shown in Table 8.4 are used for communication between mounting bases.

Table 8.4 I/O Cable Specifications

Type JZMSZ-	Length m	Application
W20-1	0.5	Used for connecting across each mounting base (MB60, MB22), respectively.
W20-2	1.5	

As for the two connectors provided on MB22 I/O buffer module, one on the top side is used for input lines and the other on the bottom side for output lines.



(2) Weight (Table 8.5)

Table 8.5 Weight of Module

Unit, Module (Type)	Approx Weight kg
CPU module	0.6
Main power supply module (PS60)	0.9
Auxiliary power supply module (PS21)	0.7
I/O processor module (IF60)	0.6
COMM module (IF61)	0.6
Remote I/O driver module (IF62, IF62A)	0.5
Remote I/O receiver module (IF70)	0.6
I/O buffer module (B2110)	0.4
100VAC input module (B2501,B2503/B2505,B2507)	0.4/0.6
12/24VDC input module (B2601, B2603)	0.5
100/200VAC output module (B2500/B2504)	0.6/0.8
12/24VDC output module (B2600/B2602)	0.8/0.5
Relay contact output module (B2902)	0.5
Relay contact output module (B2904)	0.8
Numerical input module (B2701)	0.5
Numerical output module (B2700)	0.5
Analog input (A/D) module (B2703)	0.5
Analog output (D/A) module (B2702)	0.6
Reversible counter module (B2801)	0.7
Preset counter module (B2802)	0.6
Positioning module (B2803)	0.6
Positioning module (B2813)	0.6
Mounting base (MB60)	1.4
Mounting base (MB22, MB70)	1.3
I/O cable (W20-1)	0.3
I/O cable (W20-2)	0.5
PC link module (IF64)	0.6
3200 I/F module (IF65)	0.6
Servo I/F module (IF66)	0.6
ASCII module (IF71)	1.0
FMGC control module (IF72)	0.6

8.4.2 Device Configuration in Control Panel (Cont'd)

(3) Electrical Noise

- Avoid installing GL60H or GL70H together with elements or wires carrying high-voltage and large current power* in the same panel.
- When installing GL60H or GL70H together with low-voltage main circuit† in the same panel, install the elements and wires related to the low-voltage main circuit as far apart from the GL60H or GL70H and its wiring as possible.
- Do not bind the GL60H or GL70H wiring together with general control circuit‡ wiring.
- Install the mounting base to a solid steel panel (frame). Never install these on an insulator. When the panel (frame) is painted, remove the paint from the area around the mounting holes before installing the base, in order to secure good grounding, and to prevent noise.

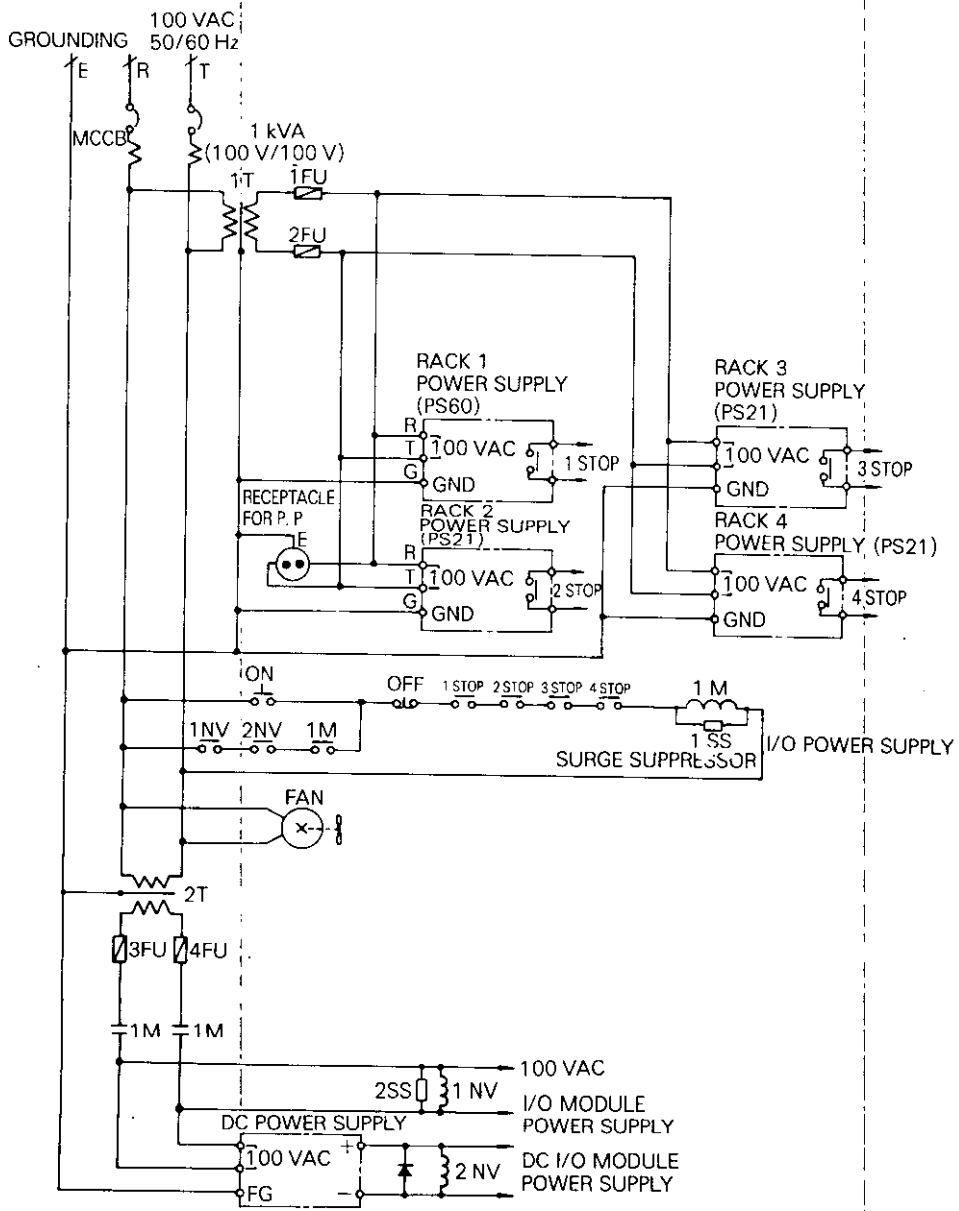
* Above 600 VAC, 750 VDC or 800 A

† Below 600 VAC or 750 VDC, with a current above 20 A

‡ Below 600 VAC or 750 VDC, with a current below 20 A

(4) Power Supply Circuit

- Fig. 8.24 shows the example of power supply circuit.
- When the power supply is in an unfavorable condition, connect a noise filter or an insulating transformer to the power supply line of a power supply module, I/O modules. Primary and secondary wirings of noise filter and transformer must be separated.
- The voltage and capacity of the power supply depend on the types of I/O modules and the connected loads.
- GL60H or GL70H starts deciphering processes immediately when the power supply is turned ON. In some systems, the power module of GL60H or GL70H may have to be energized only after connecting the I/O power supply and determining the I/O states.



NOTE

1. This example includes the minimum of necessary functions. Other required functions should be added according to each application.
2. Writing should be separated the primary side of a transformer from the secondary side.
3. "STOP" contact output from auxiliary power supply (PS60, PS21) is closed under normal I/O control of the GL60H or GL70H.

Fig. 8.24 Example of Power Supply Circuit

(5) Wiring in Panel

The wiring related to GL60H or GL70H in the panel is in types shown in Table 8.6. Use the wires of the listed sizes.

Table 8.6 Types of Wiring in Panel

Type of Wiring	Wire Size mm ²	Description
Power Supply	1.25	To be connected to the power supply terminal "100VAC" of power supply module, via circuit breaker, etc.
I/O Signal	0.3 - 1.25	To be connected to I/O signal lines and I/O module terminals (two 1.25 mm ² wires can be connected to one terminal).
Grounding	1.25	Connection between the GND terminal of the power supply module and the control panel housing (ground).

8.4.3 Grounding Wire

- The GND terminal of the power supply module should be connected to the control panel housing, as shown in Fig. 8.25 at E, and connecting point E should be connected to a ground pole.
- The grounding wire between point E and the ground pole should be larger than 8 mm² in the cross-sectional area, and should be as short as possible.
- The grounding resistance should be 100 Ω or less. (Ordinary building frames may be used. However, do not use a ground wire or ground pole in common with power lines, motors, etc.)

NOTE

When metal ducts, metal tubes or wiring racks are used, ground them in accordance with the accepted technical standards.

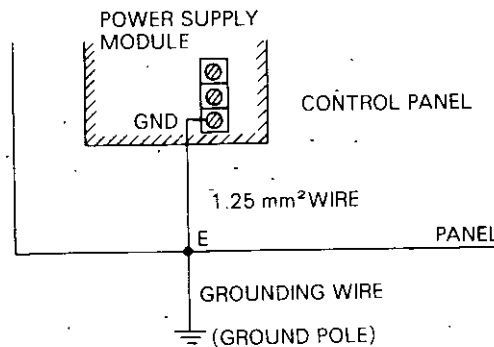


Fig. 8.25 Grounding Wire

8.4.4 External Wiring

(1) Cables for I/O Signal Lines

Cables to be used as external I/O signal lines should be selected in full consideration of the environmental conditions, mechanical strength, electrical noise, wiring length, operational voltage, etc. Isolate the I/O signal lines from each other and select cables on the basis of the guidelines given in Table 8.7.

(2) Installation of I/O Signal Line Cables

Since the I/O signal lines are low-voltage control circuit lines, separate these lines from ordinary control circuit lines and the main circuit lines as far as possible. Keep the space of minimum 10 centimeters between the lines. If they cannot be separated, use the totally shielded cables and place the iron plate between the lines to separate them completely.

Table 8.7 I/O Signal Line Cable Installation

Wiring Distance	Description
30 m max	<ul style="list-style-type: none">• DC output signal lines and DC input signal lines may be contained in the same duct. AC output signal lines and AC input signal lines also may be contained in the same duct.• DC I/O signal lines and AC I/O signal lines should be contained in their respective ducts, separately.
30 - 300 m	<ul style="list-style-type: none">• DC input signal lines, DC output signal lines, AC input signal lines and AC output signal lines should be contained in their respective ducts, separately.• Where induced voltage is high, connect a dummy resistor or use the totally shielded cables. (Shield to be grounded at GL60S side.)
300 m min	<ul style="list-style-type: none">• Do not use cables over 300 m, in view of the rush current to the output module.• Where the wiring distance is over 300 m, use a relay, and limit the wiring length between the relay and the control panel within 300 m.

8.5 SPARE PARTS

Generally, it is recommended that the following spare parts should be stocked.

- CPU module: one unit
- Main power supply module: one unit
- Auxiliary power supply module: one unit
- I/O processor module: one unit
- COMM module: one unit
- I/O buffer module: one unit
- I/O modules: At least one unit each of all the used modules. Where many modules are used, 3 % of the used number of modules are recommended as a spare parts quantity.

SECTION 9

GL60H or GL70H HANDLING AND MAINTENANCE

9.1 GL60H or GL70H INSTALLATION PROCEDURE

The mounting bases, modules, and I/O cables are shipped separately. Follow the procedures described below when installing the GL60H or GL70H in a control panel.

9.1.1 Installation of Mounting Bases

According to (1) of 8.4.2, determine the layout of mounting bases and drilled holes. Install wire ducts as necessary. There are four types of mounting bases. Install the mounting base by fastening four M5 screws through the four holes provided.

NOTE

The mounting base connectors are covered. When installing the mounting base leave the cover installed over the connectors to protect them from foreign matter.

9.1.2 Installation of Modules

Install modules of the GL60H or GL70H on the fixed mounting bases. Fig. 9.1 shows how to install a module on a mounting base.

Remove the connector cover. Fit the guide posts of the module into the guide holes of the mounting base and push the module in. Then fasten the module to the mounting base with the M4 screws provided with the module.

NOTE

Do not remove the connector cover if no module is to be installed.

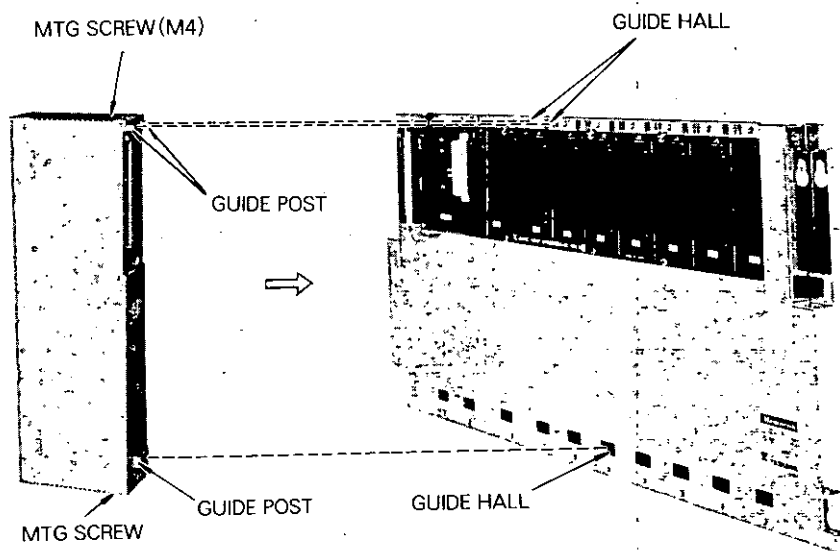


Fig. 9.1 Module Installation

The type of mounting base and the mounting location are determined depending on module types. Figs. 9.2 to 9.4 show mounting place of each module on the mounting base.

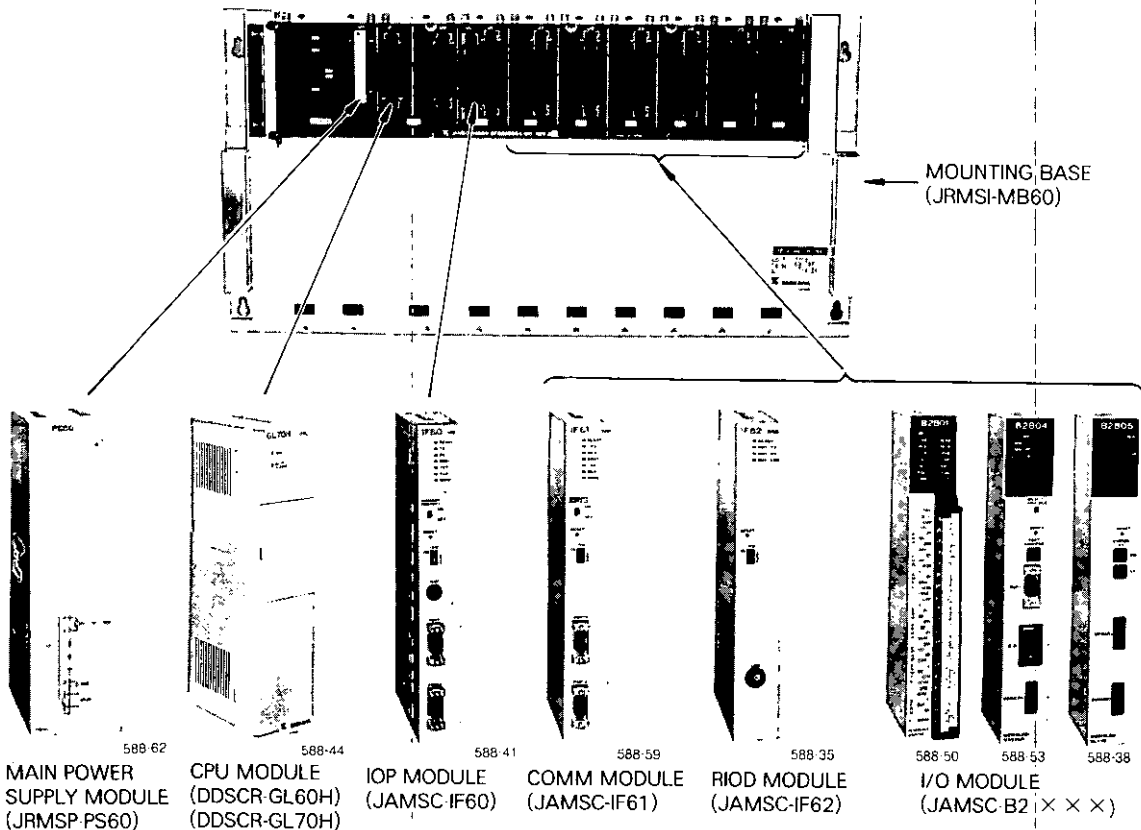


Fig. 9.2 Module Mounting on Mounting Base MB60

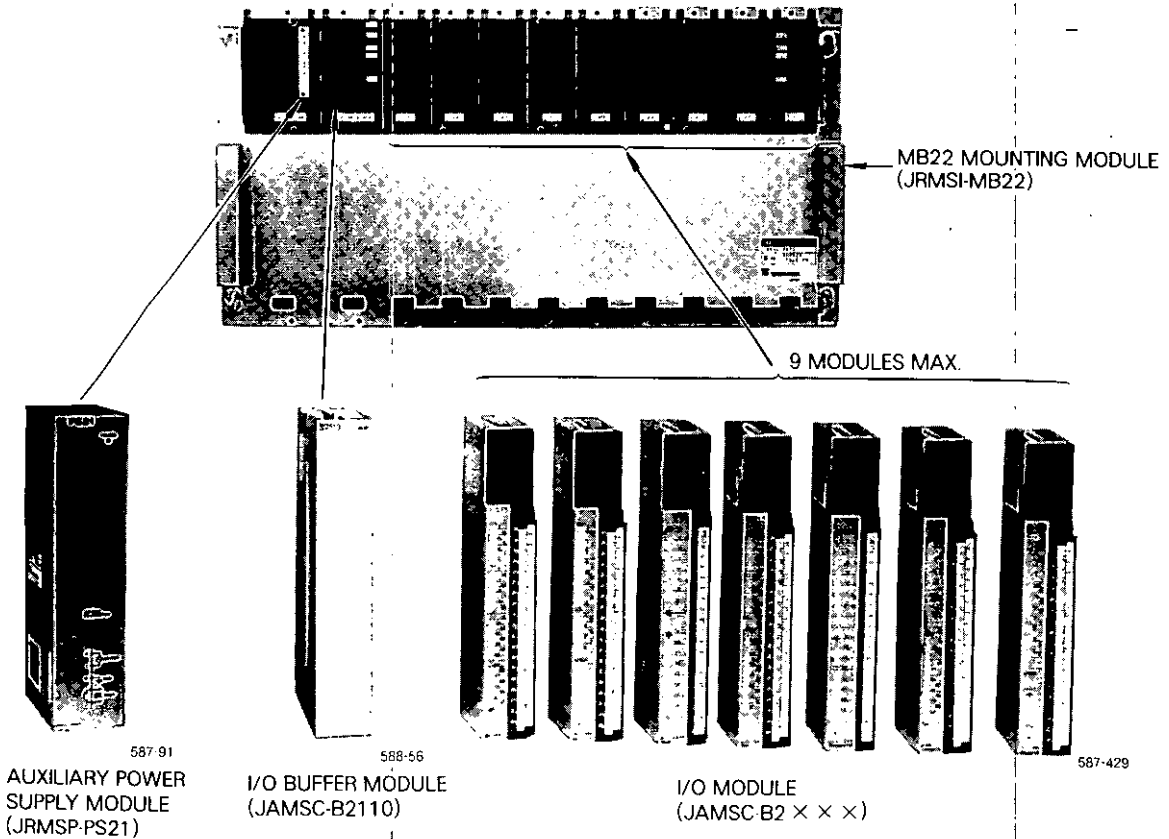


Fig. 9.3 Module Mounting on Mounting Base MB21

9.1.2 Installation of Modules (Cont'd)

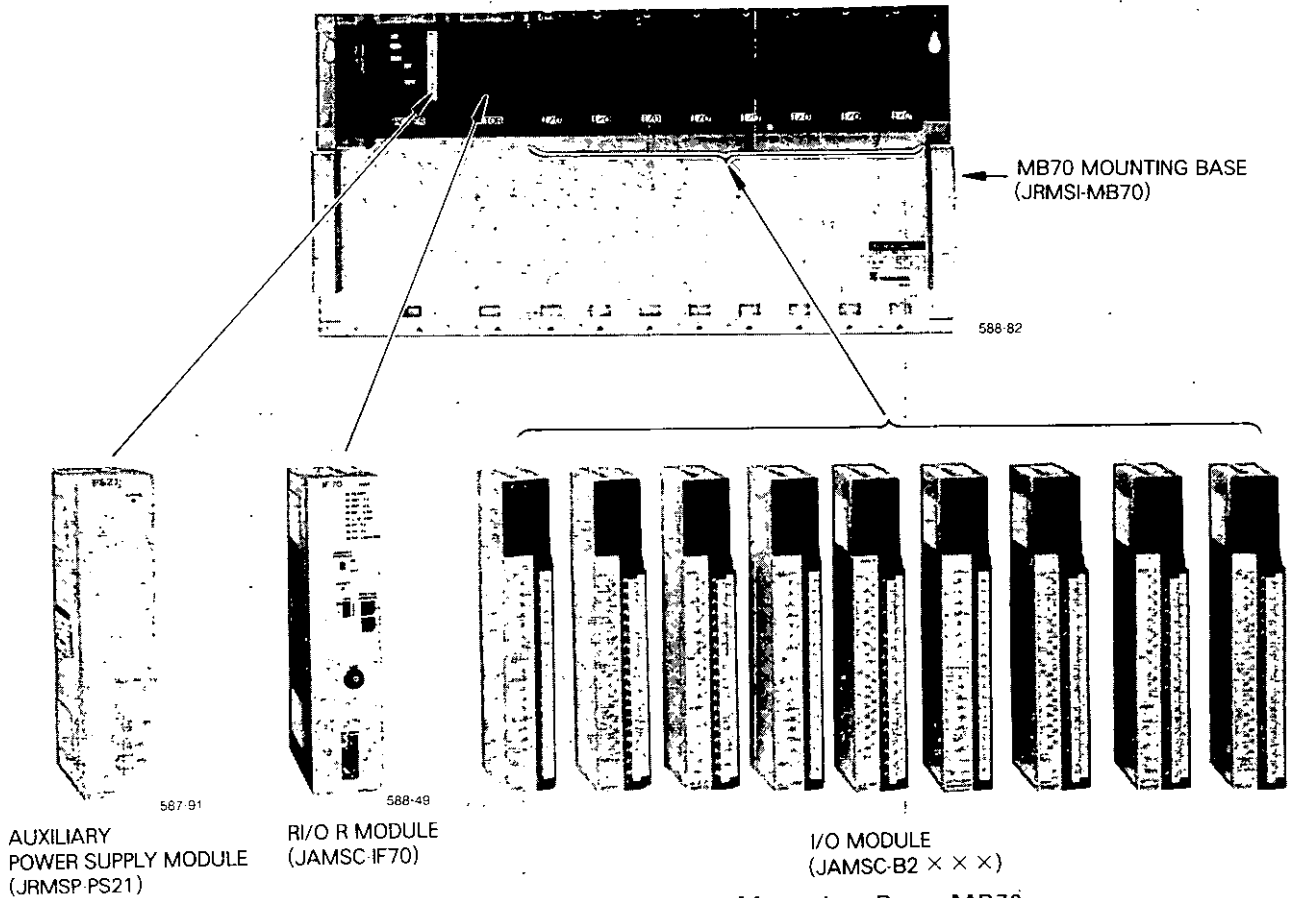
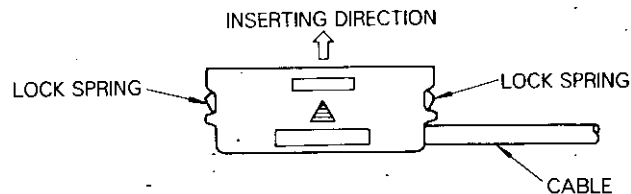


Fig. 9.4 Module Mounting on Mounting Base MB70

9.1.3 Connection of I/O Cables

Next, connect the mounting bases with I/O cables. Two I/O cables are provided: W20-1 (0.5m) and W20-2 (1.5m). Choose one according to setting conditions. Both ends of I/O cables have 50-pin connectors as shown in Fig. 9.5.



- NOTE**
1. I/O cable connector must be inserted until it is locked firmly.
 2. Remove I/O cable, depressing both sides of connector lock spring.

Fig. 9.5 I/O Cable Connector

For I/O cable connection, connector on the left of MB60 mounting base and connector on the front of I/O buffer module mounted on MB22 mounting base are used. (MB22 mounting base does not have its own I/O cable connector.)

Two connectors (IN and OUT) can be seen after removing I/O buffer module front cover.

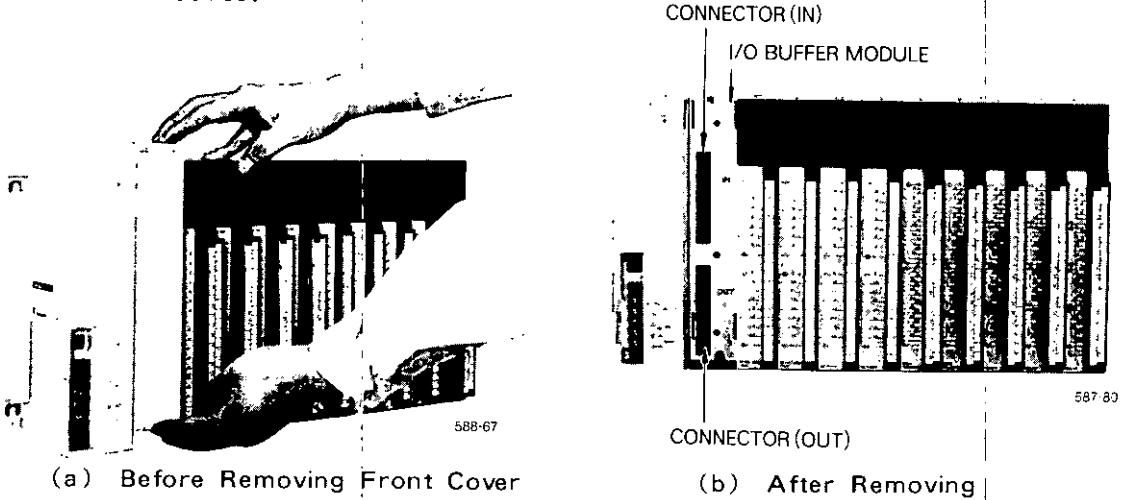
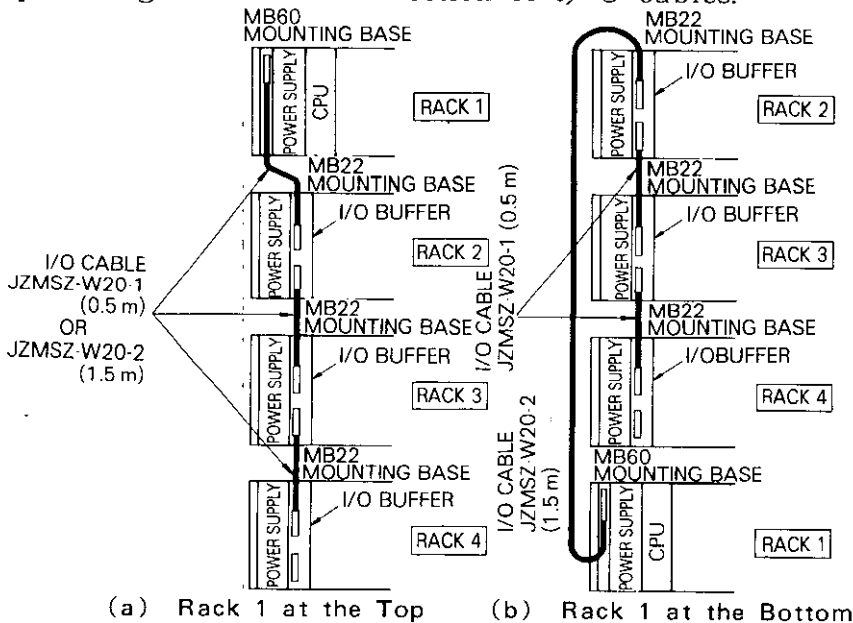


Fig. 9.6 I/O Buffer Module I/O Cable Connector

Connect the cables in the correct direction so that signal flows from output to input. Fig. 9.6 shows connection of I/O cables.



- NOTE**
1. I/O cable connector must be inserted until it is locked firmly.
 2. After completing I/O cable connection, provide covers to I/O buffer modules.

Fig. 9.7 I/O Cable Connection

9.1.4 Wiring of Modules

The power supply module and I/O modules have terminal blocks for connecting the power lines and I/O signal lines (Table 9.1).

Table 9.1 Terminal Blocks and Connectors for Wiring

No. of Pins	Power Supply Size	Applicable Pressure Terminals	Applicable Module Type	Remarks
20	1.25mm ²	Pressure terminal • R type: R1.25-3.5 • Fork type: E1.25-3.5	B2500, B2501	Removable from module
38			B2602, B2603 B2902	
5		Pressure terminal • R type: R1.25-4 • Fork type: 2-4Y F1.25-4	PS60, PS21	Not removable from module

Perform wiring to terminal block in the status with the terminal cover removed as shown in Figs. 8.8 and 8.9.

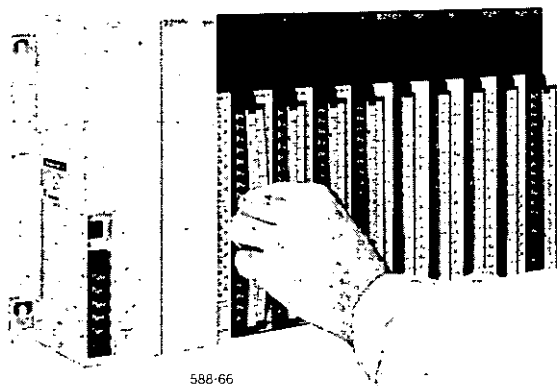


Fig. 9.8 Terminal Cover Removed

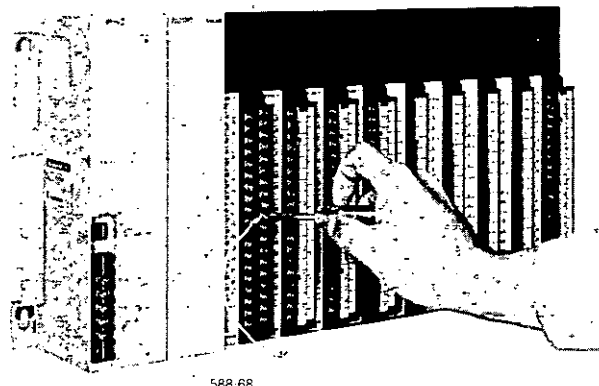


Fig. 9.9 Wiring to Terminal Block

9.2 IOP AND COMM SETTING AND ERROR INDICATION

(1) When installing the IOP and COMM modules, set the following modes by operating the DIP switches on the front panels of the modules.

① 1SW-1 (OUTPUT ENABLE setting)

Set the output state of the discrete output module to "Hold the state immediately before" or to "Off" when the CPU of GL60H or GL70H is stopped. Normally, set to OFF.

Table 9.2 OUTPUT ENABLE Setting

1SW - 1	Setting
ON	State during system down is held
OFF	All output forced OFF

② ISW-2, 3 (Setting of mode during communication command execution)

Setting is needed when executing the communication command (COM). Set as follows depending on whether the transparent or MEMOBUS mode transmission is used during communication command execution

Table 9.3 Setting of Transmission Mode during Communication Command Execution (ISW-2)

ISW-2	Setting
ON	Set Port 1 (3 for COMM) to transparent mode.
OFF	Set Port 1 (3 for COMM) to MEMOBUS mode.

Table 9.4 Setting of Transmission Mode during Communication Command Execution (ISW-3)

ISW-3	Setting
ON	Set Port 2 (4 for COMM) to transparent mode.
OFF	Set Port 2 (4 for COMM) to MEMOBUS mode.

③ ISW-4 (Setting of operation mode)

Set to the self diagnosis or normal operation mode. Must be normally set to "OFF."

Table 9.5 Operation Mode Setting

ISW-4	Setting
ON	Self diagnosis mode
OFF	Normal operaiton mode

(2) Error Indication

Status errors that occur inside the IOP and COMM modules are indicated by the LEDs on the front panel of the modules.

Table 9.6 Errors and LED Indications

Errors	LED Inidcation			Remarks
	READY	ERR1, 3	ERR2, 4	
ROM Error	●	●	○	○: Lit ●: Extinguished ◐: Blinked
RAM Error	●	○	●	
Common Memory Error	●	●	●	
Watchdog Error	●	●	●	

9.3 REMOTE LINE SETTING

(1) Set communication parameters by DIP switches on the front panels of the modules when installing a remote station using the RIOD and RIOR modules.

① RIOD module

Select a remote station (Channel 2 or 3) and set a transmission speed (4M, 2M, 1M or 0.5M).

Table 9.7 Remote Station Selection:

ISW-1	Remote Station Set:
ON	Remote 1 (Channel 2)
OFF	Remote 2 (Channel 3)

Table 9.8 Transmission Speed

ISW-3	ISW-4	Transmission Speed (M bps)
OFF	OFF	0.5
OFF	ON	1.0
ON	OFF	2.0
ON	ON	4.0

NOTE ISW-2 is not used.

② RIOR module

Set a transmission speed (4M, 2M, 1M or 0.5M).

Table 9.9 Transmission Speed

ISW-3	ISW-4	Transmission Speed (M bps)
OFF	OFF	0.5
OFF	ON	1.0
ON	OFF	2.0
ON	ON	4.0

NOTE ISW-1 and-2 are not used.

Note that communications are not conducted if the transmission speeds set by RIOD and by RIOR do not match.

(2) RIOR Station Address Setting

GL60S allows a maximum of 31 stations each to be installed with Channels 2 and 3 as remote stations which must have station Nos. intrinsic to them. These station Nos. are set using the station address switch installed on the front panel of the RIOR module; 1 to 31 can be set. The same station address cannot be used twice inside the same channel.

(3) Error Indication

① RIOD module

If any error occurs, LEDs on the module function as follows:

"READY" --Extinguished

"RMT ERR" --Blinked

For error contents, refer to Par. 9.7 "GL60S SYSTEM STATUS."

② RIOR module

If any error occurs, LEDs on the module show the contents of error as follows:

Table 9.10 Errors and LED Indications

Error	LED Indications				Remarks
	READY	RMT ERR	I/O ERR	PP COM ERR	
ROM Error	○	●	○	○	○: Lit ●: Blinded
RAM Error	○	○	●	○	
Watchdog Error	○	○	○	●	
Station Set Error	●	○	○	○	

9.4 MODULE REPLACEMENT

The indicator lamps of the modules and the programming panel permit locating a defective module.

It is possible to locate and replace the defective module so quickly that system down time will be minimized.

Follow the procedures given below to replace a module.

(1) Turn off supply power.

Remove the AC power from the GL60H or GL70H and external supply power from the I/O modules. If I/O modules are replaced with the power ON, outputs might be erroneous.

(2) Remove the terminal block and connector. [Fig. 9.10 (a)]

It is not necessary to remove the wires separately. Rather, remove the terminal block. Remove the terminal block two screws, top and bottom.

Remove the power lines one by one since power supply module terminal block is not removable.

(3) Loosen the screws fastening the module. [Fig. 9.10 (b)]

Loosen the two screws, top and bottom, fastening the module to the mounting base, until they are released from the base.

(4) Remove the module. [Fig. 9.10 (c)]

Holding the top and bottom of the module, pull it out toward you.

9.4 MODULE REPLACEMENT(Cont'd)

(5) Install a new module.

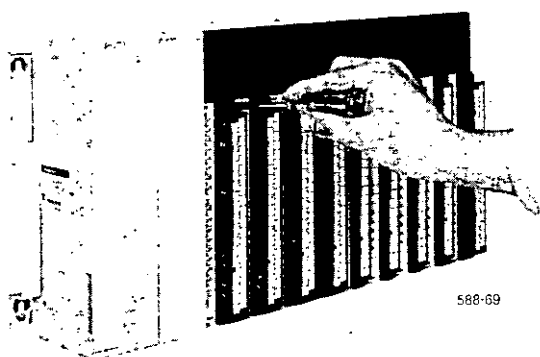
Fit the guide posts of the module into the guide holes of the mounting base and push the module in so that the guide posts slip into the holes completely. Then fasten the module to the mounting base with the screws provided with the module.

(6) Install the terminal block and connector.

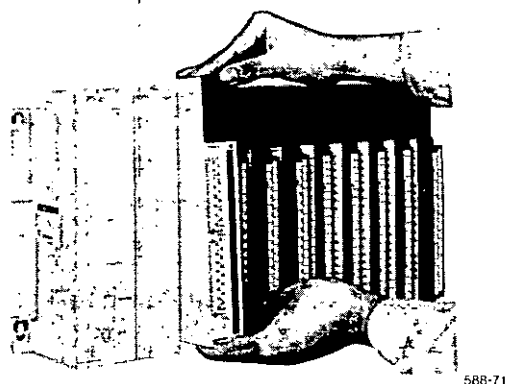
Replace the terminal block removed in step (2).

(7) Turn ON power.

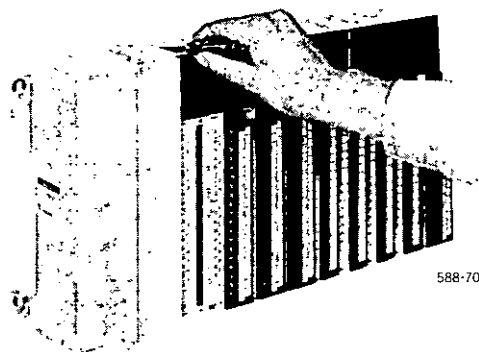
Make sure that the module and wiring are correct, before turning ON power.



(a) Removing Connector Terminal Block



(c) Pulling out Module



(b) Loosening Module Mounting Screws

Fig. 9.10 Module Replacement

9.5 BATTERY REPLACEMENT

Back-up power for the GL60S memory is provided by battery. It is recommended to replace the battery every two years.

NOTE Battery service life varies with the environmental conditions (temperature and humidity) and working time (AC power failure time).

(1) Battery Specifications

Table 9.11 gives the specifications of the battery used in the GL60H or GL70H.

Table 9.11 Battery Specifications

Item	Specifications
Name	Lithium battery
Type	BR-2/3A-1
Manufacturer	Matsushita Battery Industry Co., Ltd.
Nominal Voltage	3V
Nominal Capacity	1200 m Ah
Ambient Temperature	0 °C to + 55 °C
Storage Temperature	- 20 °C to + 45 °C
Life	• Warranty: 5 years at 25 °C • Actual protective period for memory: 1 year at 25 °C
Approx Weight	15g

NOTE When the battery in the Table above is required, contact Yaskawa representative.

(2) Battery Replacement Interval

The battery will last for a maximum of five years or until the total time of AC power failure reaches a year. If the "BATT ALARM" lamp of the CPU module lights, replace the battery within one month.

Approximate standards for battery replacement are given below.

Depending on GL60H or GL70H average daily operating hours

- 12 hours/day: replace every two years.
- 16 hours/day: replace every three years.
- 20 hours/day: replace every four years.

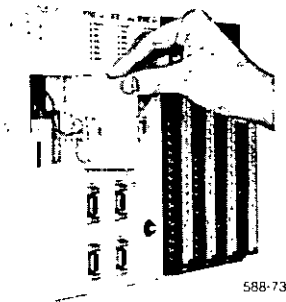
(3) Battery Replacement Procedures

Be sure to replace the battery with the CPU module connected to the base and receiving AC power to the main power supply module. If the battery is removed with no AC power supplied, the memory contents will be destroyed. But if the battery is removed with no AC power supplied for a short time (within 30 minutes), the memory contents will not be destroyed.

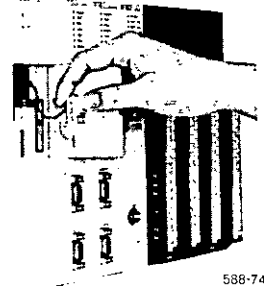
Replace the battery using the following procedures:

- ① Preheat the soldering iron.
- ② Confirm that the POWER lamp of the main power supply module is lit (AC power ON).
- ③ Remove the front cover. See Fig. 9.11 (a).
- ④ Take out the battery from the battery holder, then separate the connector attached at the ends of the leads from the CPU module. See Fig. 9.11 (b).
- ⑤ Remove the leads from the battery, using the preheated soldering iron. See Fig. 9.11 (c).
- ⑥ Solder the removed leads to the new battery, using care to observe the polarity. (Lead colors Red: Positive; Black: Negative)
- ⑦ Place the new battery in the battery holder and insert the connector attached at the ends of the leads into the CPU module connector.
- ⑧ Confirm that the "BATT ALARM" lamp of the CPU module goes off.
- ⑨ Provide the battery cover and turn off the power supply for the soldering iron.

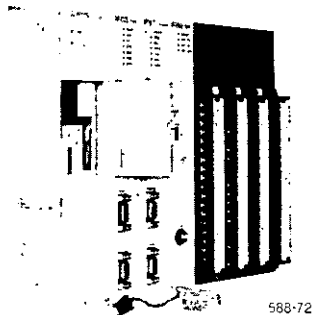
This completes the battery replacement.



(a) Opening CPU Module Front Cover



(b) Taking out Battery from Battery Holder



(c) Separating Connector from CPU Module

Fig. 9.11 Battery Replacement

9.6 REGISTER ACCESS PANEL

Connected to the connector on the front panel of the IOP module, the register access panel (RAP) is used to display ON/OFF states of coils and input relays, simulated operation (forced ON and OFF) and register data and to set and change data of holding registers.

Fig. 9.12 shows the register access panel. The RAP displays as follows when power is turned on.

GL70H

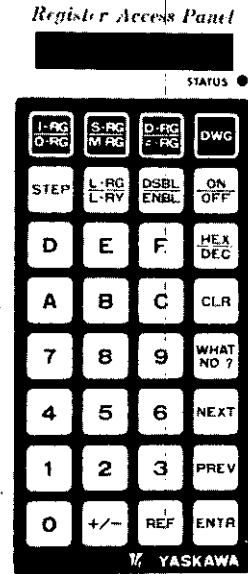


Fig. 9.12 Register Access Panel

9.6.1 Operation Keys

Table 9.12 RAP Operation Keys

Key	Function
0 to 9 A to F	Numerical keys to input Nos. and numerals. Keys A to F are invalid in decimal input.
$\frac{-}{+}$	① Set whether to monitor registers with or without signs. Valid only for decimal monitoring. (Invalid during monitoring of step timers and port parameters.) ② Set whether to input values in registers without signs or as positive and negative numerals. Valid for decimal input. (Invalid during step timer and port parameter changes.)
CLR	Depress this key to clear display to " ". Depressed to reset error display or to cancel key function during operation.
DSBL ENBL	Used in simulated operation of coils and input relays. Depress this key to set up simulated state (DISABLE). By depressing it again, the simulated state is released (ENABLED). The state reverts each time the key is depressed. Valid if the memory protect switch is OFF. Steps cannot be changed forcibly.

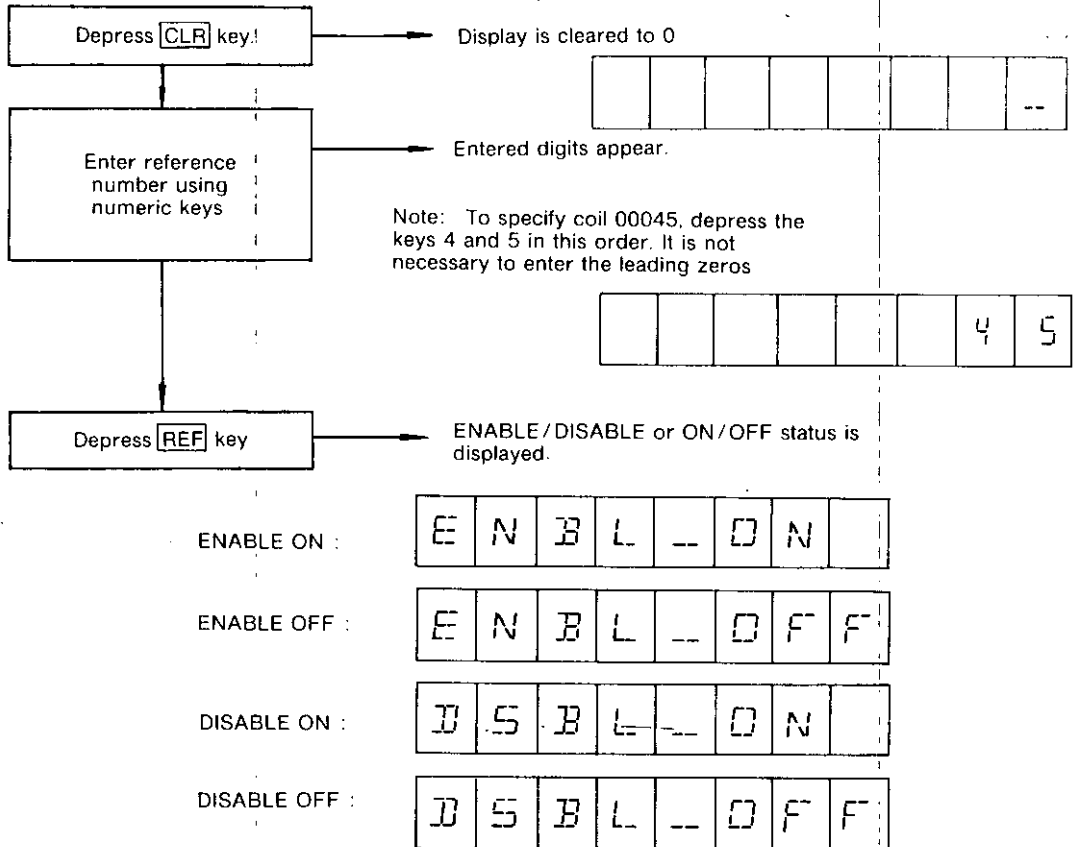
Table 9.12 RAP Operation Keys (Cont'd)

Key	Function
<div style="border: 1px solid black; padding: 2px; text-align: center;">ON</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">OFF</div>	Forcibly switches coil or input relay state in a simulated condition on or off. The state reverts each time the key is depressed. Valid if the memory protect switch is OFF. Steps cannot be forcibly switched ON and OFF.
<div style="border: 1px solid black; padding: 2px; text-align: center;">HEX</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">DEC</div>	Changes type of values and indicated value input (decimal or hexadecimal). Depress this key to alternately change from decimal to hexadecimal and vice versa.
<div style="border: 1px solid black; padding: 2px; text-align: center;">STEP</div>	The key sets step reference. Depress this key to clear indication and function, indicating only 5 and the cursor. Depress again to leave only the cursor.
<div style="border: 1px solid black; padding: 2px; text-align: center;">ENTR</div>	This key stores input value in the holding register or sets as a port parameter. Valid if memory protect switched off.
<div style="border: 1px solid black; padding: 2px; text-align: center;">WHAT NO?</div>	Depressed to show which reference No. or address is monitored at present.
<div style="border: 1px solid black; padding: 2px; text-align: center;">NEXT</div>	Depress this key to monitor data of reference No. or address next to what is being monitored at present.
<div style="border: 1px solid black; padding: 2px; text-align: center;">PREV</div>	Depress this key to monitor data of reference No. or address prior to what is being monitored at present.
<div style="border: 1px solid black; padding: 2px; text-align: center;">REF</div>	Depress this key to monitor data of set reference No. and address.
<div style="border: 1px solid black; padding: 2px; text-align: center;">L-RG</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">L-RY</div>	This key sets references for link registers and link relays. Depress once to set up link register reference acknowledge state; depress twice to set up link relay reference acknowledge state; and depress three times to normal reference acknowledge state.
<div style="border: 1px solid black; padding: 2px; text-align: center;">I-RG</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">O-RG</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">S-RG</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">M-RG</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">D-RG</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">#-RG</div> <div style="border: 1px solid black; padding: 2px; text-align: center;">DWG</div>	Keys for GL60V and are always invalid with GL60H or GL70H.

9.6.2 Operation

(1) Displaying ON/OFF Status of Input Relay 1 × × × × or Coil 0 × × × ×

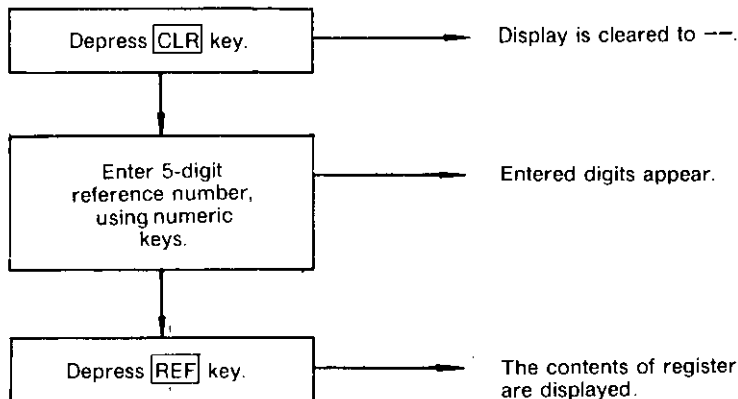
Set reference number, and the status for example, of the specified coil will be displayed enabled (observing ladder logic) or disabled (turned ON or OFF independently of ladder logic).



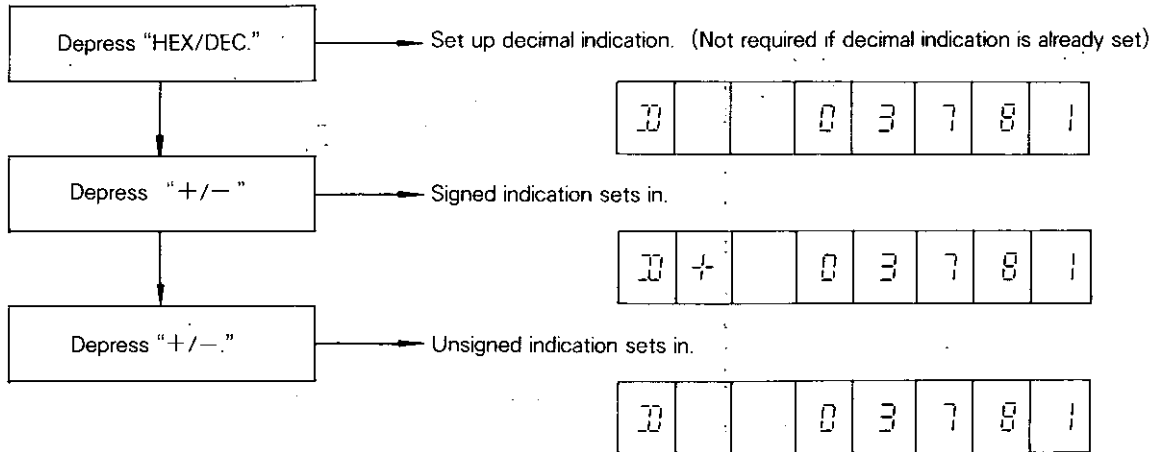
- The status display is refreshed in every scanning cycle.
- If you push **NEXT** (**PREV**) at this time, the reference number is increased (decreased) by one and the status of the input relay or coil identified by the updated reference number appears.

(2) Displaying Contents of Input Register 30 × × × , Constant Register 3 × × × × or Holding Register 4 × × × ×

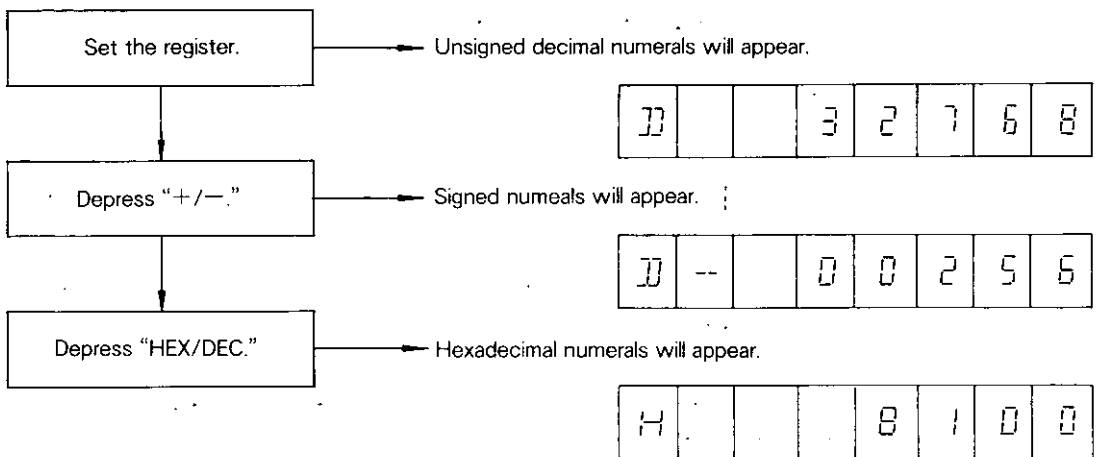
Set a reference number, and the contents of the register will be displayed. It is possible to select decimal or hex for the displayed data.



- The data will be refreshed in every scanning cycle.
- The reference number is increased (decreased) by one every time you push **NEXT** (**PREV**).
- Display of the data changed over in decimal form and hex form alternately as you push **HEX/DEC**. The decimal mode is selected first.
- Signed indication is operated as follows.



- By indicating negative numerals hexadecimally, "1" sets in the most significant bit. If negative numerals are indicated without sign, hexadecimal numerals are converted into decimal numerals as is and unusual numerals will be indicated. An example of register data of -256 (256 is 100h hexadecimally) is shown below.

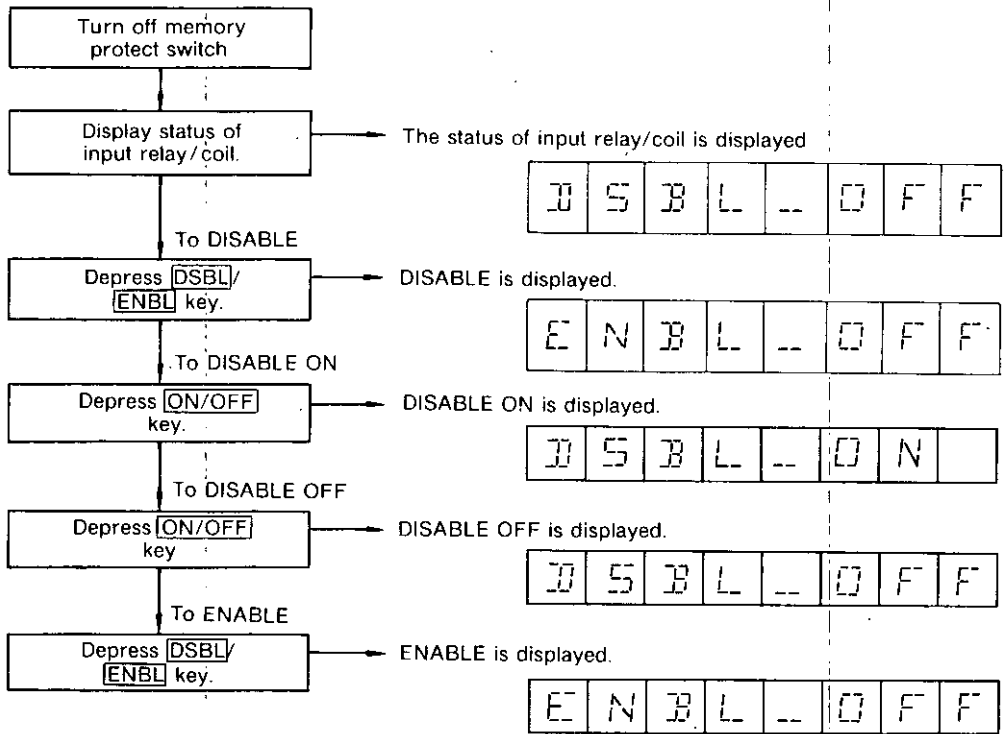


(3) DISABL/ENABL of input Relay or Coil

It is possible to DISABLE or ENABLE the input relay or actual input signal of which status is read by the method of (1).

As soon as you disable an input relay or coil, the input relay becomes independent of the actual signal and the coil of the ladder circuit, holding the ON or OFF status it has kept so far.

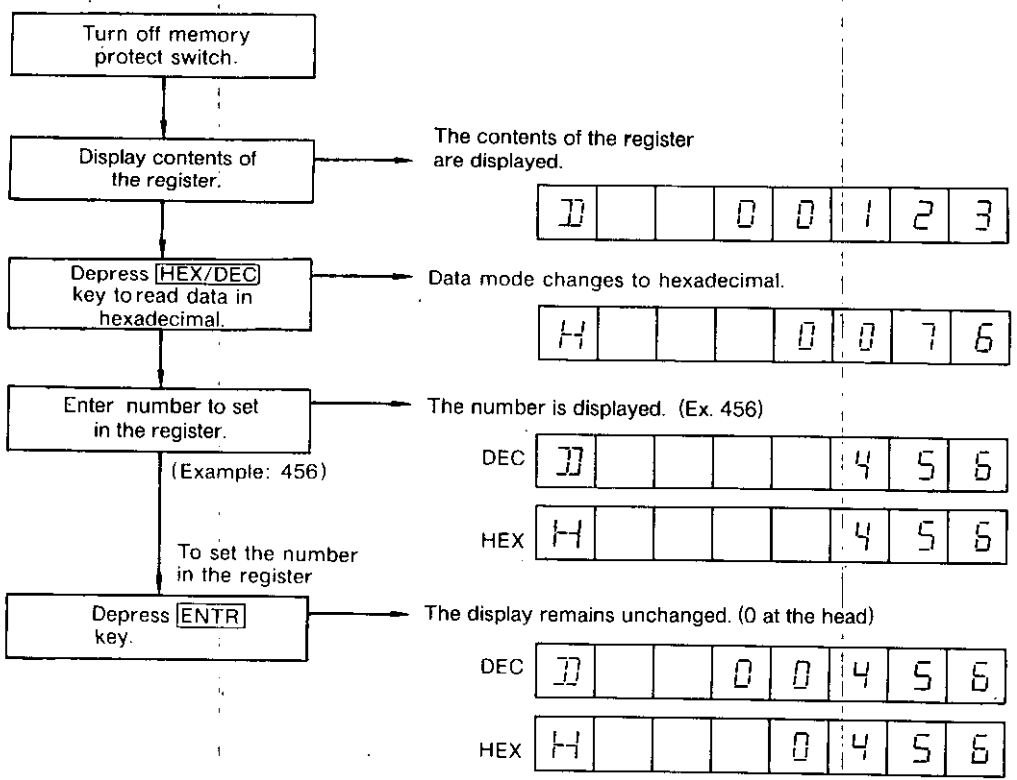
The disabled input relax and coil can be turned ON and OFF unconditionally. When enabled, the input relay is activated according to the actual signal and the coil to the logic of the ladder circuit.



- **DSBL/ENBL** and **ON/OFF** , change mode alternately when depressed.
- The reference number is increased (decreased) by one every time **NEXT** (**PREV**) is depressed.

(4) Setting or Altering of Constant Register, Holding Register

It is possible to set a value in the constant register, holding register of which contents are read by the method of (2).



- To input signs also, set decimal indication and set signs by depressing “ + / - ” after inputting numerals. Then depress “ENTR” to store numerals in the register.

Display	Sign	Applicable Setting Value Range
DEC	without	0 to 65535
	with	- 32767 to 32767
HEX	—	0 to FFFF

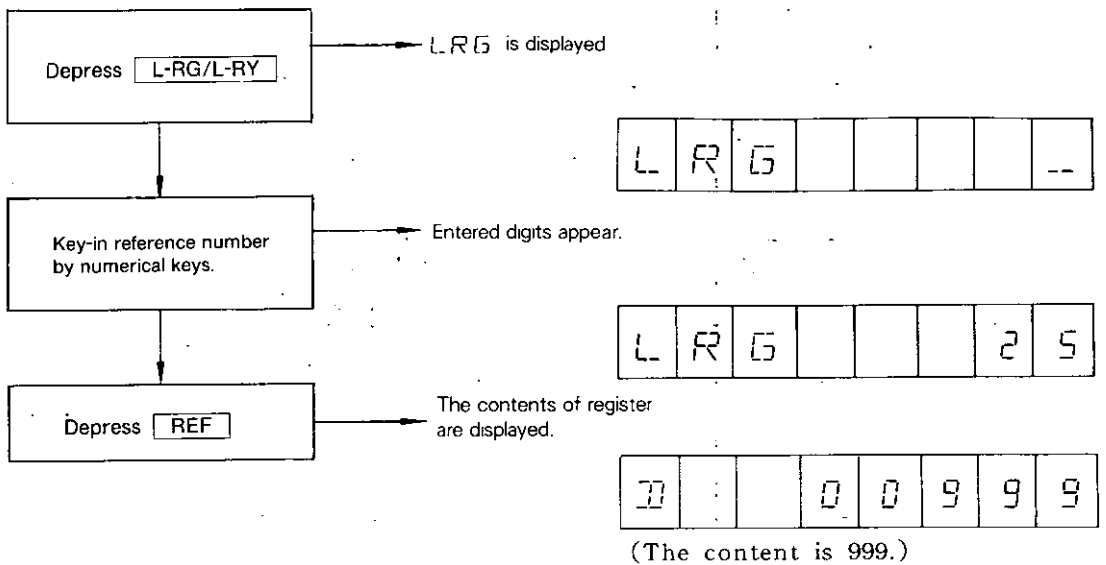
- **HEX/DEC** changes mode alternately when depressed.
- The reference number is increased (decreased) by one every time **NEXT** (**PREV**) is depressed.

(5) Indication of Data of Timer Register 50 × × ×

- Operate as in operating constant and holding registers. This is possible only with unsigned decimal indication.
- Setting or change of timer register data cannot be performed as on the programming panel.

(6) Indication of Link Register Data

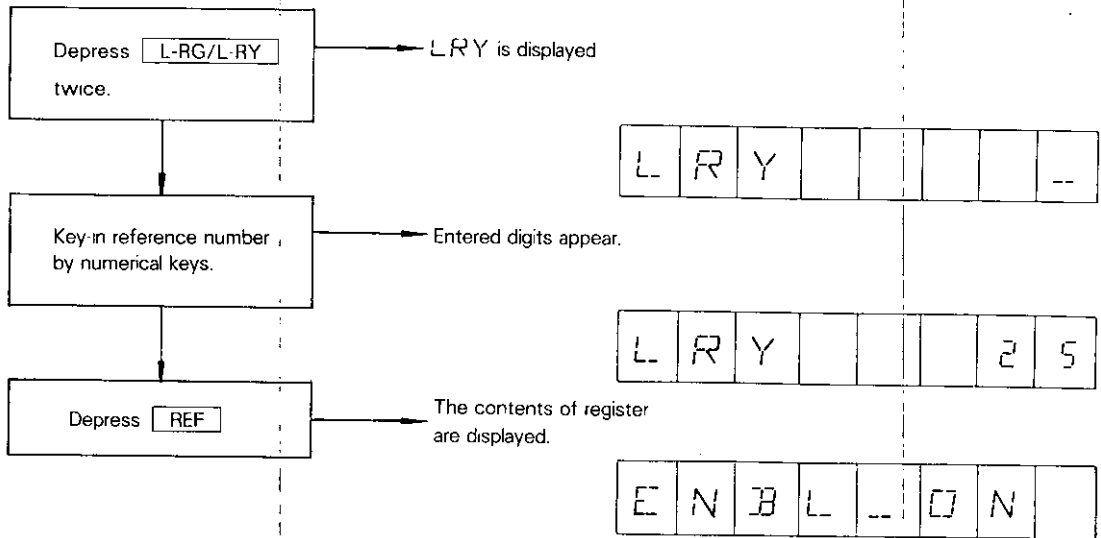
Set the reference No. to indicate the data of the register.



- Data is updated each scan.
- Reference No. is increased (decreased) by 1 each time **NEXT** (**PREV**) key is depressed.
- **HEX/DEC** changes mode alternately when depressed.
- The contents of link register cannot be set or changed.

(7) Status Indication of Link Coil (Relay)

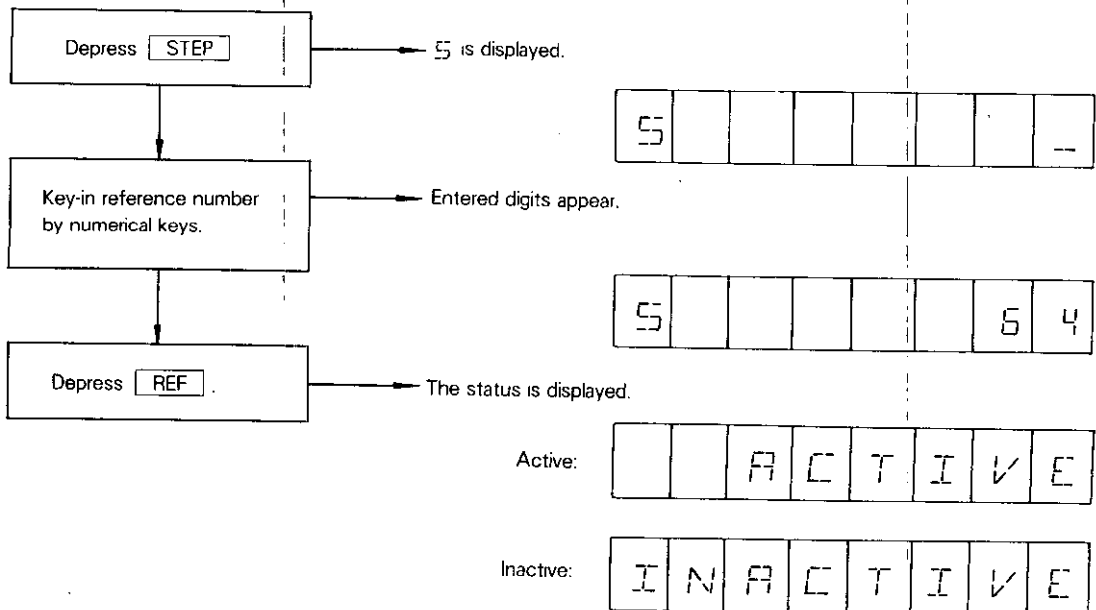
Set the reference No. to indicate the link coil (relay).



- The data will be refreshed in every scanning cycle.
- The reference number is increased (decreased) by one every time **NEXT** (**PREV**) is depressed.
- The contents of link coil (relay) cannot be set or changed.

(8) Status Indication of SFC Step

Set the reference No. to indicate whether the step is active or inactive.

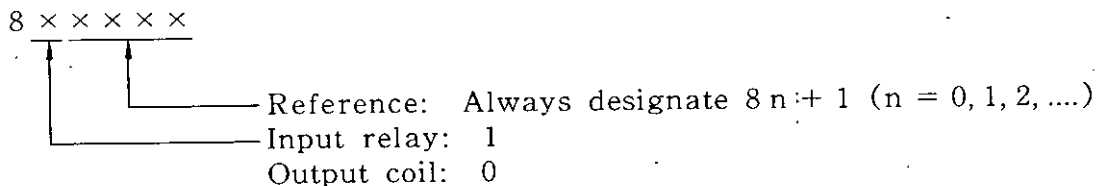


- The data will be refreshed in every scanning cycle.
- The reference number is increased (decreased) by one every time **NEXT** (**PREV**) is depressed.
- Step status cannot be changed forcibly from active to inactive and vice versa.

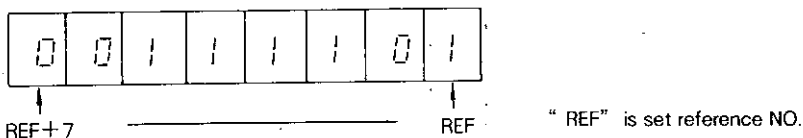
(9) 8-point Indication of Input Relay and Output Coil Status

Data of eight references in succession can be indicated only with input relays and output coils by inputting specified 6-digit No.

- Operate as in (1). The specified No. will be as follows. "8" is fixed.



- ON and OFF state is indicated as follows:

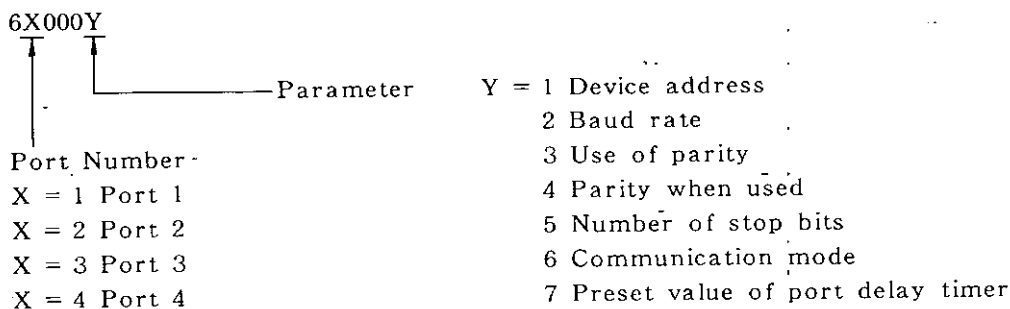


"0" and "1" indicate OFF and ON state

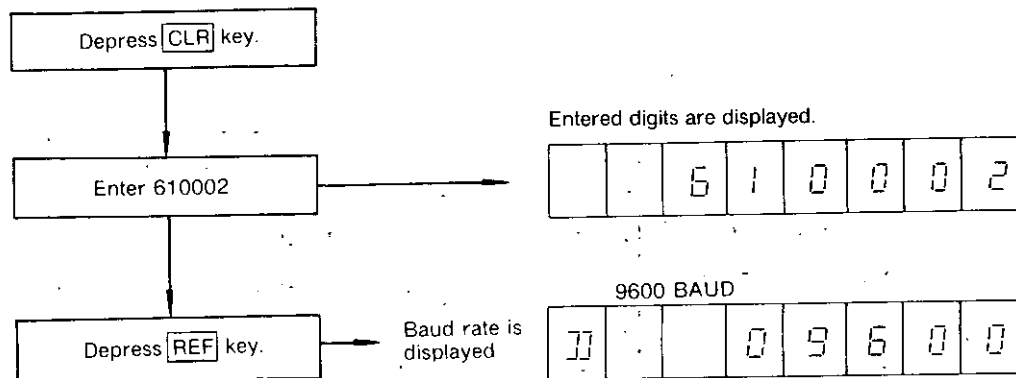
- Reference No. is increased (decreased) by 8 each time "NEXT" ("PREV") is depressed.

(10) Displaying Port Parameters

It is possible to display a communication parameter between the communication module and a device (the programming panel, etc.) connected to a port by entering a special number which is described below. 6 and 000 are fixed.



Operate as follows to look at the baud rate of the device connected to port 1, for example.



Communication parameters associated with Y are as follows.

Y = 1 Device address: 1 to 247
 Y = 2 Baud rate: 150, 300, 600, 1200, 2400, 4800, 9600, or 19200

Y = 3

P	A	R	R	I	T	Y	O	N
---	---	---	---	---	---	---	---	---

 • • Parity check performed.

N	O	P	A	R	I	T	Y
---	---	---	---	---	---	---	---

 • • Parity check not performed.

Y = 4

E	V	E	N				
---	---	---	---	--	--	--	--

 • • Even parity.

					O	D	D
--	--	--	--	--	---	---	---

 • • Odd parity.

Y = 5 Number of stop bits: 1 or 2

Y = 6 Communication mode

R	T	U					
---	---	---	--	--	--	--	--

 • • RTU

			A	S	C	I	I
--	--	--	---	---	---	---	---

 • • ASCII

Y = 7 Preset value of port delay timer (GL60S delays response by this value): 0 to 255 (correspond to 0 to 2550 msec)

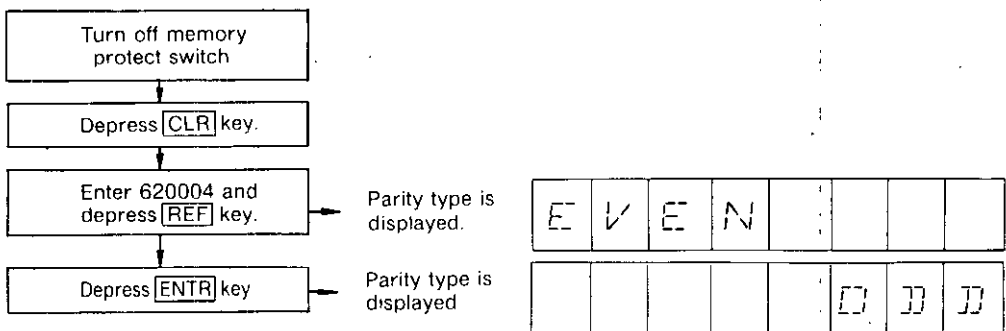
- The value of Y is increased (decreased) by one every time **NEXT** (**PREV**) is depressed.
- Device address, baud rate and delay timer are displayed only with unsigned decimal.

(11) Setting or Altering of Communication Parameters

To set a device address or baud rate, enter the number to be set and push **ENTR**, then the number is set.

In other cases, push **ENTR** only. Then one of two choices changes to the other.

To change even parity of port 2 to odd, for example;

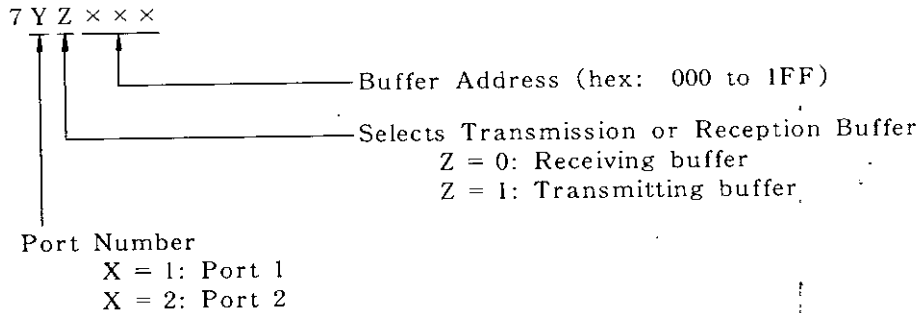


- The value of Y is increased (decreased) by one every time **NEXT** (**PREV**) is depressed.

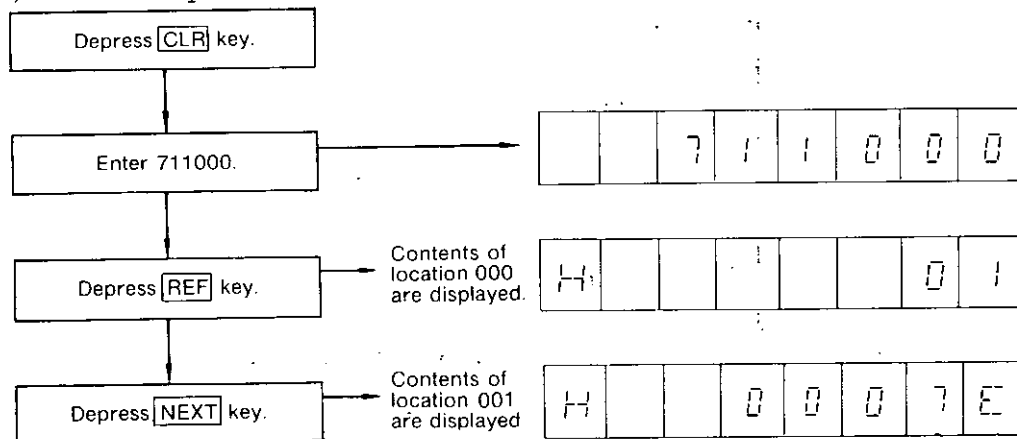
9.6.2 Operation (Cont'd)

(12) Displaying of Contents of Communication Buffer

As in communication parameters, it is possible to display the contents of the communication buffer of a MEMOBUS port by entering a special number which is described below. 7 is fixed.



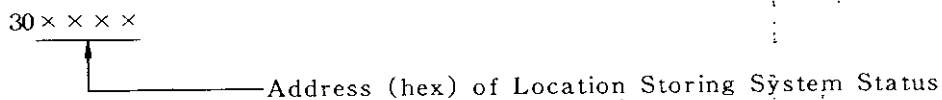
Operate as follows to look at the contents of transmitting buffer of port 1, for example.



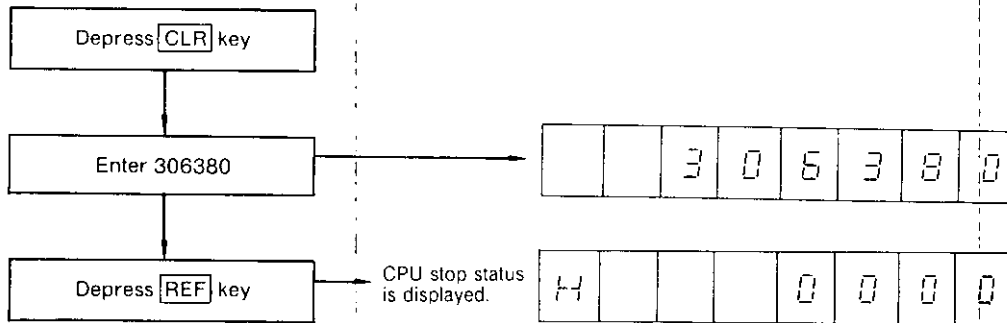
- The data is displayed in hex form at first.
- The buffer address × × × is increased (decreased) by one every time **NEXT** (**PREV**) is depressed.
- Transmission/reception buffer contents of ports 3 and 4 cannot be displayed from RAP.

(13) Displaying of GL60H or GL70H System Status

It is possible to display the GL60H or GL70H system status by entering a special number which is described below. 3 and 0 are fixed.



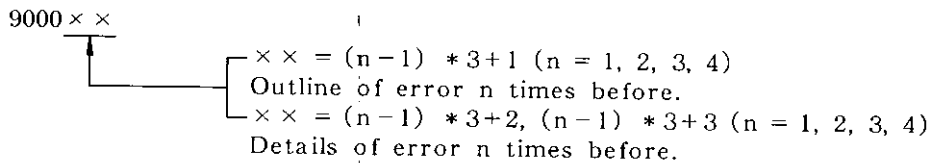
GL60H or GL70H status storing address numbers are described in Par. 9.6. Operate as follows to read the CPU stop status, for example.



- The data is displayed in hex form at first.
- The address $\times\times\times\times$ is increased (decreased) by one every time (**PREV**) is depressed. **NEXT**

(14) Failure History State Display

Input the following specified No. to indicate the states of past four CPU stoppages.



9.6.3 Error Codes

(1) Wrong Operation

Table 9.12 summarizes the codes indicating wrong operations. Wrong data would not be stored in memory, but the normal state can be restored by recovery operation. Note that no error code will be displayed on RAP if a similar mistake is made on the programming panel: it appears only when RAP is operated.

Table 9.13 Error Codes of Wrong Operations

Error Code	Error Description
ERROR 01	Reference is not correct.
ERROR 02	Expansion memory (for future) is not provided.
ERROR 03	For future expansion.
ERROR 04	Setting and changing data are out of range.
ERROR 05	Input relay, coil are not DISABLE.
ERROR 06	Setting and changing are not permitted.
ERROR 07	Memory protect is switched ON.
ERROR 08	Other wrong operations.

9.6.2 Operation (Cont'd)

- Return-home method

Depress CLR to indicate data or to return to indicate data if error occurred during data setting. In other cases, indication is cleared.

NOTE [CLR] not only clears the digital and character displays to 0 but also cancels the function specified. Before making a new entry, it is recommended to depress [CLR] twice in order to cancel the data and function entered.

(2) System Errors

Table 9.14 System Errors

Error Code	Content
SYSERR 10	Transmission time-out error with CPU module
SYSERR 20	Transmission time-out error with IOP module
SYSERR 99	ROM total sum error of RAP

9.6.4 Status LED Lamp

The LED lamp located at lower right of the display panel is lit if the following occurs:

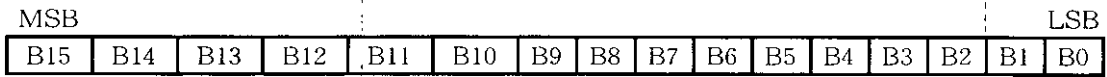
- The relay switches ON during indication of the input relay data. (Except for 8-point simultaneous indication)
- The coil switches ON during indication of the output coil data. (Except for 8-point simultaneous indication)
- The link relay switches ON during display of the link relay data.

Table 9.15 GL60H or GL70H System Status

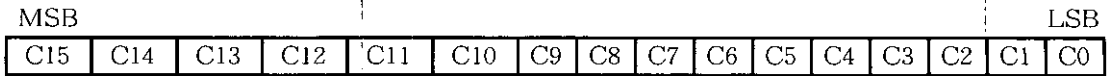
Hex			
6380	GL60H, GL70H STOP STATUS	63B5	RIOD2 STATION 1 STATUS
6381	TOTAL SUM ERROR	63B6	RIOD2 STATION 2 STATUS
6382	DATA ERROR	63B7	RIOD2 STATION 3 STATUS
6383	FOR FUTURE EXPANSION	63B8	RIOD2 STATION 4 STATUS
6384	FOR FUTURE EXPANSION	63B9	RIOD2 STATION 5 STATUS
6385	GL60H, GL70H MACHINE STATUS	63BA	RIOD2 STATION 6 STATUS
6386	RIOD1 STATUS	63BB	RIOD2 STATION 7 STATUS
6387	RIOD2 STATUS	63BC	RIOD2 STATION 8 STATUS
6388	IOP STATUS	63BD	RIOD2 STATION 9 STATUS
6389	COMM STATUS	63BE	RIOD2 STATION 10 STATUS
638A	LINK STATUS	63BF	RIOD2 STATION 11 STATUS
638B	3200IF STATUS	63C0	RIOD2 STATION 12 STATUS
638C	EXTENSIVE MEMORY STATUS	63C1	RIOD2 STATION 13 STATUS
638D	SVIF 1 STATUS	63C2	RIOD2 STATION 14 STATUS
638E	SVIF 2 STATUS	63C3	RIOD2 STATION 15 STATUS
638F	SVIF 3 STATUS	63C4	RIOD2 STATION 16 STATUS
6390	SVIF 4 STATUS	63C5	RIOD2 STATION 17 STATUS
6391	FOR FUTURE EXPANSION	63C6	RIOD2 STATION 18 STATUS
6392	FOR FUTURE EXPANSION	63C7	RIOD2 STATION 19 STATUS
6393	FOR FUTURE EXPANSION	63C8	RIOD2 STATION 20 STATUS
6394	LOCAL I/O STATUS	63C9	RIOD2 STATION 21 STATUS
6395	RIOD1 STATION 1 STATUS	63CA	RIOD2 STATION 22 STATUS
6396	RIOD1 STATION 2 STATUS	63CB	RIOD2 STATION 23 STATUS
6397	RIOD1 STATION 3 STATUS	63CC	RIOD2 STATION 24 STATUS
6398	RIOD1 STATION 4 STATUS	63CD	RIOD2 STATION 25 STATUS
6399	RIOD1 STATION 5 STATUS	63CE	RIOD2 STATION 26 STATUS
639A	RIOD1 STATION 6 STATUS	63CF	RIOD2 STATION 27 STATUS
639B	RIOD1 STATION 7 STATUS	63D0	RIOD2 STATION 28 STATUS
639C	RIOD1 STATION 8 STATUS	63D1	RIOD2 STATION 29 STATUS
639D	RIOD1 STATION 9 STATUS	63D2	RIOD2 STATION 30 STATUS
639E	RIOD1 STATION 10 STATUS	63D3	RIOD2 STATION 31 STATUS
639F	RIOD1 STATION 11 STATUS	63D4	FOR FUTURE EXPANSION
63A0	RIOD1 STATION 12 STATUS	63D5	LOCAL I/O ERROR (RACK 1)
63A1	RIOD1 STATION 13 STATUS	63D6	LOCAL I/O ERROR (RACK 2)
63A2	RIOD1 STATION 14 STATUS	63D7	LOCAL I/O ERROR (RACK 3)
63A3	RIOD1 STATION 15 STATUS	63D8	LOCAL I/O ERROR (RACK 4)
63A4	RIOD1 STATION 16 STATUS	63D9	LOCAL I/O ERROR (RACK 5)
63A5	RIOD1 STATION 17 STATUS	63DA	STOP STATUS, TOTAL SUM
63A6	RIOD1 STATION 18 STATUS	63DB	ERROR AND DATA ERROR
63A7	RIOD1 STATION 19 STATUS	63DC	AT LAST POWER CUT-OFF.
63A8	RIOD1 STATION 20 STATUS	63DD	STOP STATUS, TOTAL SUM ERROR
63A9	RIOD1 STATION 21 STATUS	63DE	AND DATA ERROR OF ONE POWER
63AA	RIOD1 STATION 22 STATUS	63DF	CUT-OFF BEFORE THE LAST.
63AB	RIOD1 STATION 23 STATUS	63E0	STOP STATUS, TOTAL SUM ERROR
63AC	RIOD1 STATION 24 STATUS	63E1	AND DATA ERROR OF TWO POWER
63AD	RIOD1 STATION 25 STATUS	63E2	CUT-OFF BEFORE THE LAST.
63AE	RIOD1 STATION 26 STATUS	63E3	STOP STATUS, TOTAL SUM ERROR
63AF	RIOD1 STATION 27 STATUS	63E4	AND DATA ERROR OF THREE POWER
63B0	RIOD1 STATION 28 STATUS	63E5	CUT-OFF BEFORE THE LAST.
63B1	RIOD1 STATION 29 STATUS	63E6	LOCAL RACK STATUS
63B2	RIOD1 STATION 30 STATUS	63E7	LOCAL RACK ACK STATUS
63B3	RIOD1 STATION 31 STATUS	63E8	LOCAL RACK ACK HISTORY
63B4	FOR FUTURE EXPANSION	63E9	
		63EA	LINK STATION STATUS

(1) The following names for individual bits are set in explaining the semantics of the status.

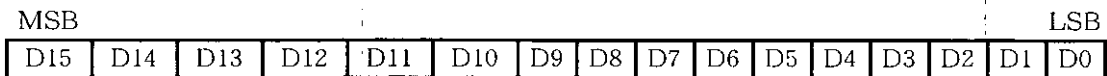
- CPU stop status



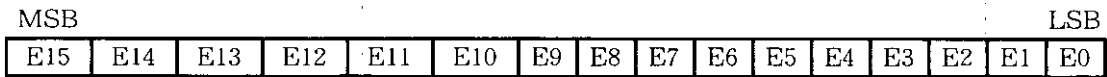
- Table total sum error



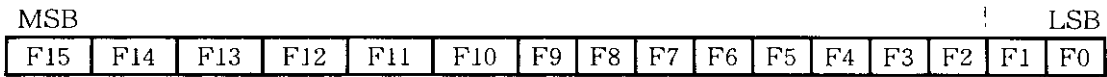
- Table data error



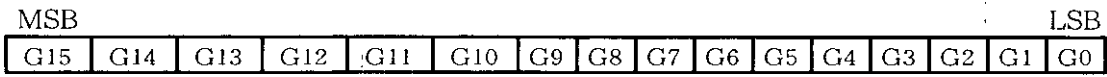
- GL60S machine status



- RIO station status

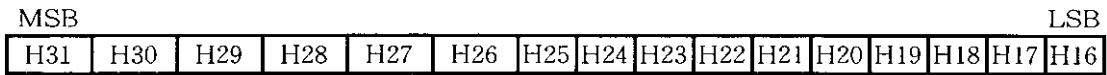


- Local rack status

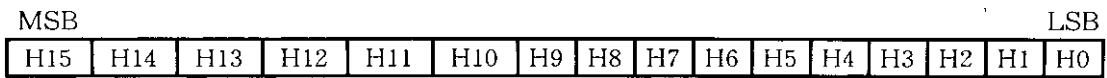


- Link station status

63E9



63EA



(2) GL60H or GL70H stop status (Address 6380)

Table 9.16 GL60H or GL70H stop status (Address 6380)

Bit	Semantics
B15	Set if either 1 or B14 to B0 is 1 in case CPU is stopping.
B14	1 if watchdog timer error.
B13	1 if real time clock error.
B12	1 if bit ALU error.
B11	1 if invalid memory allocation.
B10	1 if ladder circuit error (Set if C15, D15, D11, D10 or D6 is 1.)
B9	1 if action circuit error. (Set if C14, C6, D14, D9, or D5 is 1.)
B8	1 if transition condition circuit error (Set if C13, C5, D13, D8 or D4 is 1.)
B7	1 if subroutine circuit error (Set if C12, C4, D12, D7 or D3 is 1.)
B6	1 if SFC program error (Set if C2, D2 or D1 is 1.)
B5	1 if user status table total sum error.
B4	1 if T-COP table error. (Set if C11 or C3 is 1.)
B3	Other allocated table error (Set if C10, C9, C8 or C7 is 1.)
B2	CP213 allocated table total sum error
B1	PC link allocated table total sum error
B0	Subsystem trouble

(3) Table Sum Total Errors

Table 9.17 Table Sum Total Errors

Bit	Semantics
C15	Ladder circuit total sum error
C14	Action circuit total sum errors
C13	Transition condition circuit sum total errors
C12	Subroutine circuit sum total errors
C11	T-COP table sum total error
C10	2-level scan allocated table sum total error
C9	ASCII allocated table error
C8	User memory allocated table sum total errors
C7	Transmission parameter table sum total errors
C6	Step entry table sum total errors
C5	Transition entry table sum total errors
C4	Subroutine entry table sum total errors
C3	T-COP entry table sum total errors
C2	SFC flow table sum total error
C1	SFC mode table sum total errors
C0	Not used.

(4) Table Data Errors

Table 9.18 Table Data Error

Bit	Semantics
D15	Ladder circuit EOL error
D14	Action circuit EOL error
D13	Transition condition circuit EOL error
D12	Subroutine circuit EOL error
D11	Ladder circuit EOS error
D10	Ladder circuit SON error
D9	Action circuit SOA error
D8	Transition condition circuit SOT error
D7	Subroutine circuit SOS error
D6	Not used.
D5	Not used.
D4	Not used.
D3	Not used.
D2	Step data error
D1	Transition data error
D0	Not used.

(5) GL60H or GL70H Machine Status

Table 9.19 GL60H or GL70H Machine Status

Bit	Semantics	Bit	Semantics
E15	For future expansion	E7	
E14	Local I/OD error	E6	
E13	RIOD1 error	E5	
E12	RIOD2 error	E4	
E11	IOP error	E3	
E10	COMM error	E2	
E9	Link error	E1	
E8	CP213 error	E0	

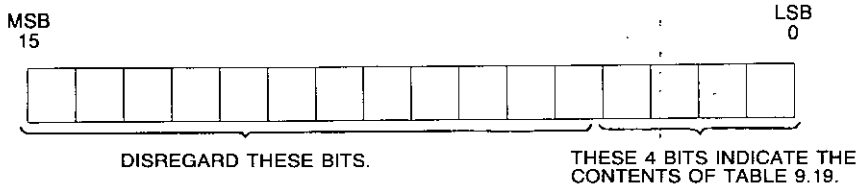
(6) RIOD 1/2, IOP, COMM, LINK Status, 3200 IF Status, SVIF Status

These status data are read out from I/F RAM of each module. If status data are error codes, corresponding bit in machine status is set at "1."

Table 9.20 RIOD, IOP, COMM, LINK Status, 3200 IF Status, SVIF Status

CODE (H)	Meaning
0	Normal
1	ROM error
2	RAM error
3	Common memory error
8	LINK stop (For only LINK status)
F	Watchdog error

RIOD1 : 6386 ADDRESS	3200 IF : 638B ADDRESS
RIOD2 : 6387 ADDRESS	SVIF1 : 638D ADDRESS
IOP : 6388 ADDRESS	SVIF2 : 638E ADDRESS
COMM : 6389 ADDRESS	SVIF3 : 638F ADDRESS
LINK : 638A ADDRESS	SVIF4 : 6390 ADDRESS



RIOD 1, RIOD 2 : JAMSC - IF 62 A	3200 IF : JAMSC - IF 65
IOP : JAMSC - IF 60	SVIF 1 to 4 : JAMSC - IF 66
COMM : JAMSC - IF 61	
LINK : JAMSC - IF 64	

(7) RIOD 1 station status 1 to RIOD 2 station status 31 (Address 6395 - 63 D 3)
Status data of the stations read from the remote I/O drivers RIOD 1 and 2 enter.

Table 9.21 RI/O Station Status

Bit	Semantics
F15	1 without transmission error and allocated station
F14	T-COP allocated data error
F13	Wrong output data length, without output data
F12	Not used.
F11	Not used.
F10	Not used.
F9	Not used.
F8	BUS check error
F7	TCOP requested from RIOR (1 at power-on);
F6	Not used.
F5	Not used.
F4	Not used.
F3	1 when I/O service is less than the specified time.
F2	BUSY check error
F1	Not used.
F0	0 with normal RIOR

NOTE Though 2008H error occurs momentarily at RIOR power-on; this does not mean malfunctioning.

(8) Local I/O Error Table (Address 63 D 5 - 63 D 9)

Data to show errors of modules mounted in the local I/O racks are stored. Fig. 9.13 shows relationships between bits and slots. Numerals indicate slot Nos.

	MSB						LSB			
63D5	1	2	3	4	5	6				
63D6	1	2	3	4	5	6	7	8	9	
63D7	1	2	3	4	5	6	7	8	9	
63D8	1	2	3	4	5	6	7	8	9	
63D9	1	2	3	4	5	6	7	8	9	

Fig. 9.13 Local I/O Error Table

(9) Local Rack status (Address 63 E 6)

Data to show error of each local I/O rack are stored.

Table 9.22 Local Rack Status

Bits	Meaning
G15 to G5	Not used
G4	Rack 5 error
G3	Rack 4 error
G2	Rack 3 error
G1	Rack 2 error
G0	Rack 1 error

(10) LINK Station Status (Address 63 E 9)

For each link station (32 stations max), CPU module stores data to show module-mounted condition and errors. Bit in normal station is set to "1." Bit in troubled station (including station with no module) is set to "0."

Table 9.23 LINK Station Status

Bits	Stations	Bits	Stations
H31	Station 16	H15	Station 32
H30	Station 15	H14	Station 31
H29	Station 14	H13	Station 30
H28	Station 13	H12	Station 29
H27	Station 12	H11	Station 28
H26	Station 11	H10	Station 27
H25	Station 10	H9	Station 26
H24	Station 9	H8	Station 25
H23	Station 8	H7	Station 24
H22	Station 7	H6	Station 23
H21	Station 6	H5	Station 22
H20	Station 5	H4	Station 21
H19	Station 4	H3	Station 20
H18	Station 3	H2	Station 19
H17	Station 2	H1	Station 18
H16	Station 1	H0	Station 17

9.8 TROUBLESHOOTING

In order that repairs can be made as easily and quickly as possible without a knowledge of details, the GL60H or GL70H is maintained in basic units of modules, that is, by module replacement.

If a failure occurs, the first thing to do is to identify the problem accurately and follow the prescribed procedure for maintenance. Make use of the general GL60H or GL70H troubleshooting flowcharts provided in this section.

Maintenance of the GL60H or GL70H requires spare parts listed on Par. 8.4.

NOTE

1. The symbols used in the flow charts have the following meanings:

□ : Action ○ : Terminal or comment ◇ : Decision ○ : Connector

2. Power on/off operations are omitted from the flow charts. Normally, power will be turned off before a maintenance action, and turned on after the action has been completed.

(1) Power Supply Module Check

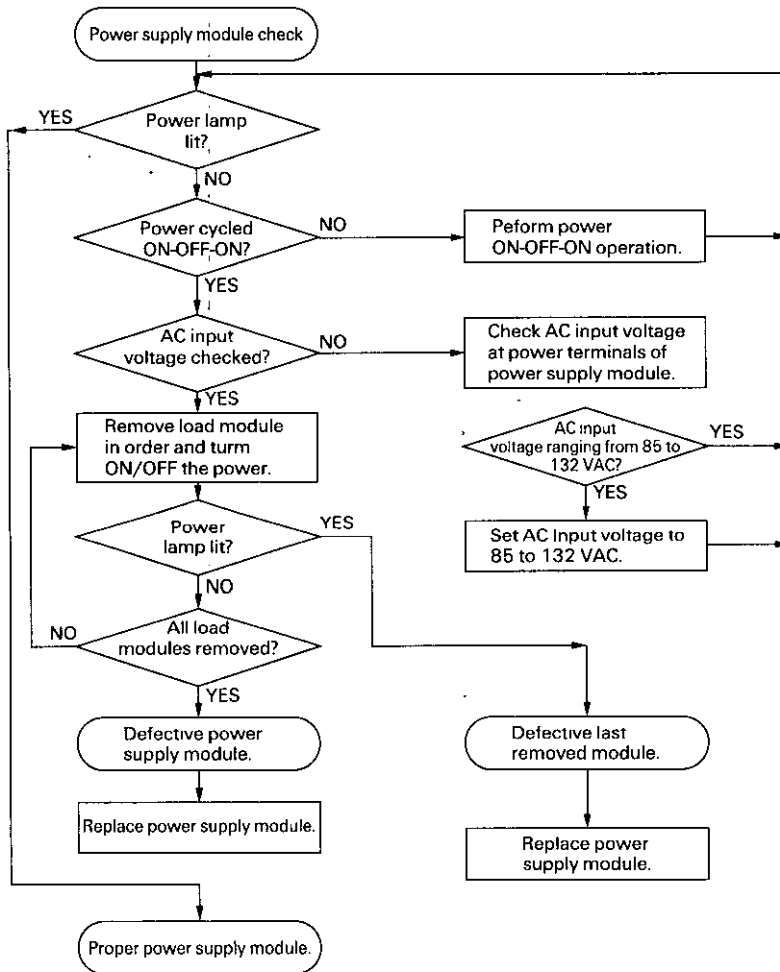


Fig. 9.14 Power Supply Module Check

NOTE

When circuit protector trips at checking PS21, reset the circuit protector for normal operation.

(2) CPU Module Check

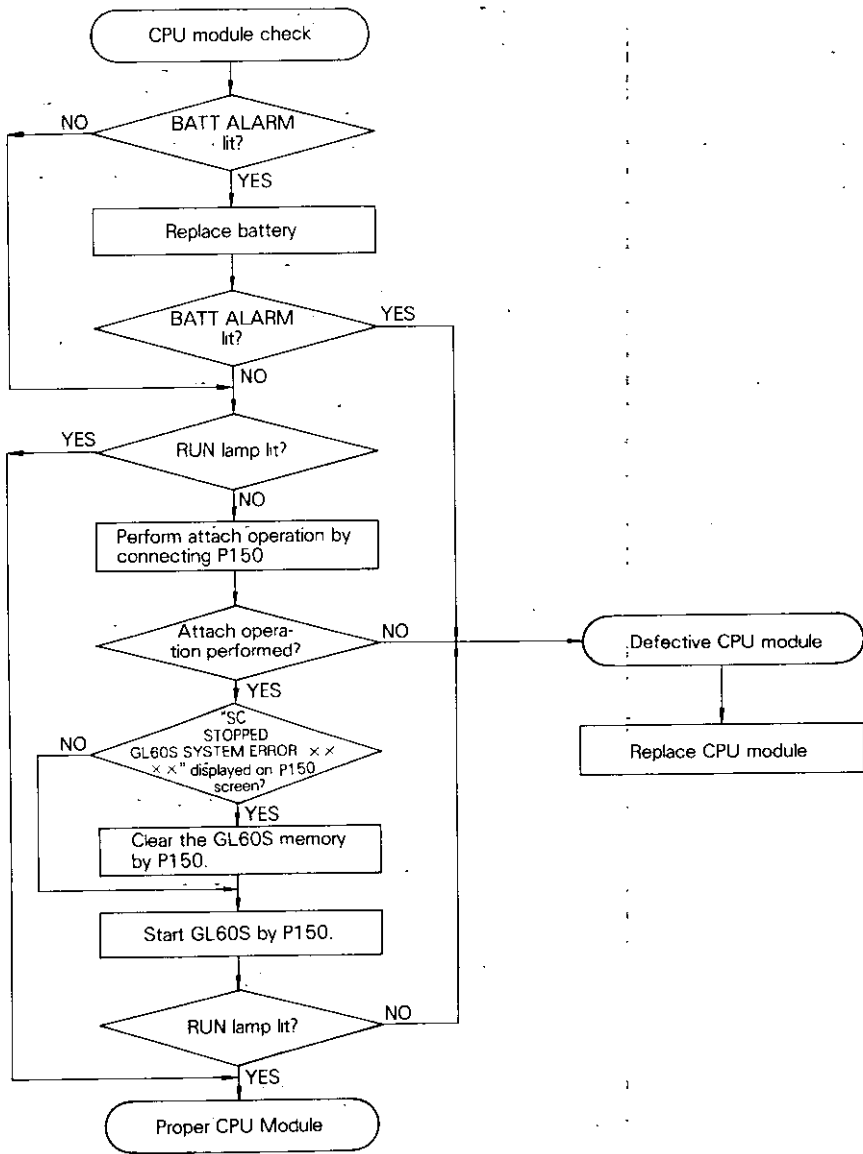


Fig. 9.15 CPU Module Check

(3) I/O Processor Module Check

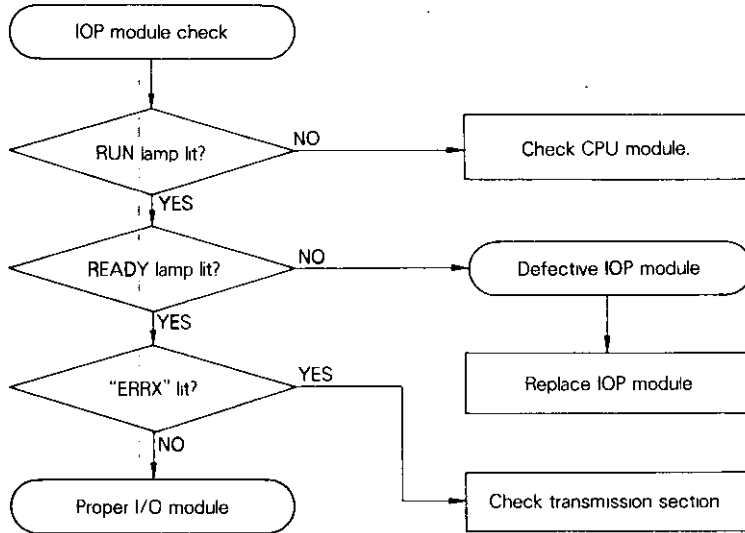


Fig. 9.16 I/O Processor Module Check

(4) COMM Module Check

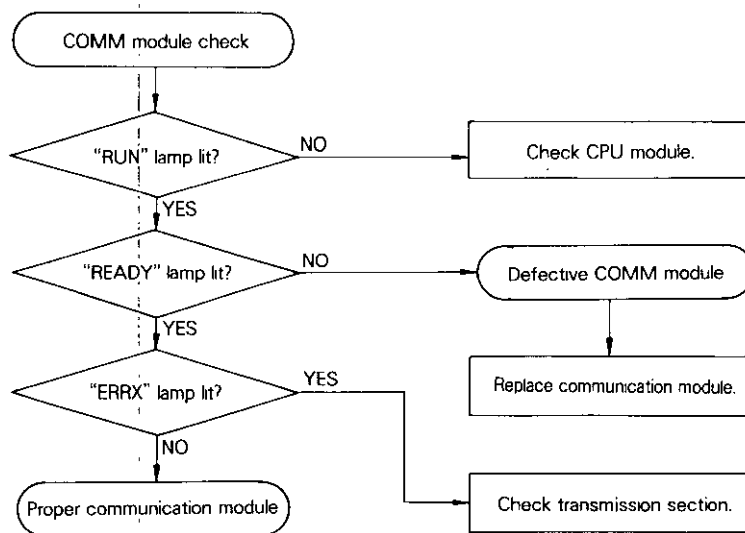


Fig. 9.17 COMM Module Check

(5) RIOD Module Check

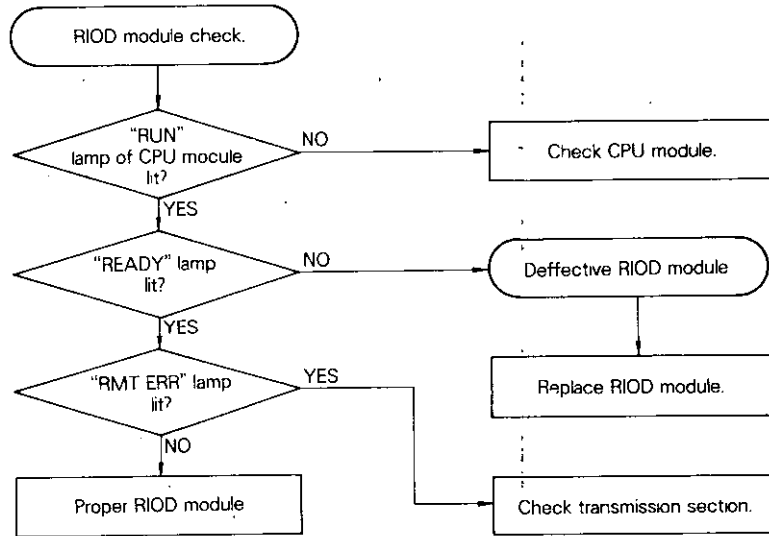


Fig. 9.18 RIOD Module Check Flow

(6) RIOR Module Check

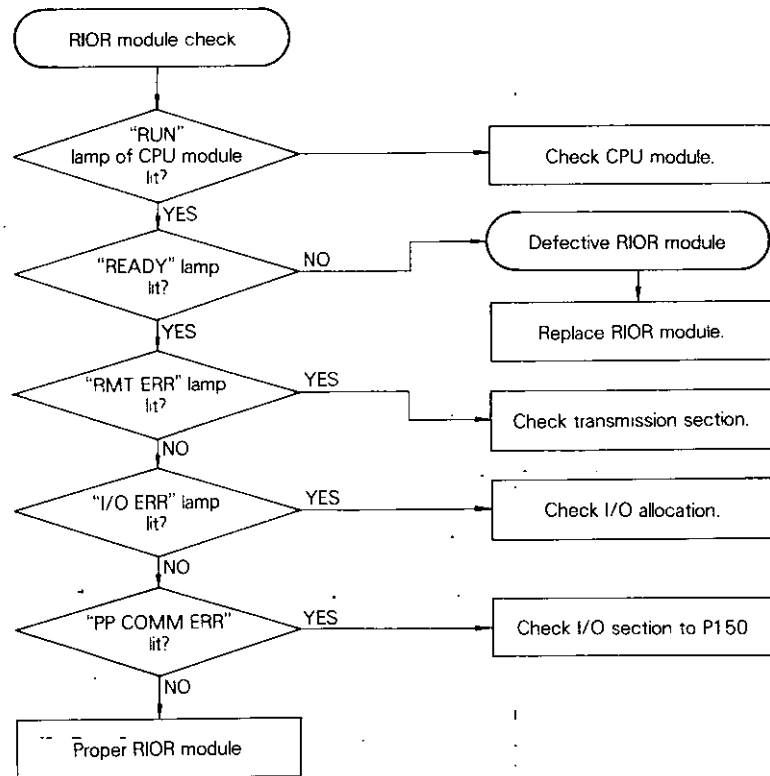


Fig. 9.19 RIOR Module Check Flow

(7) I/O Section Check

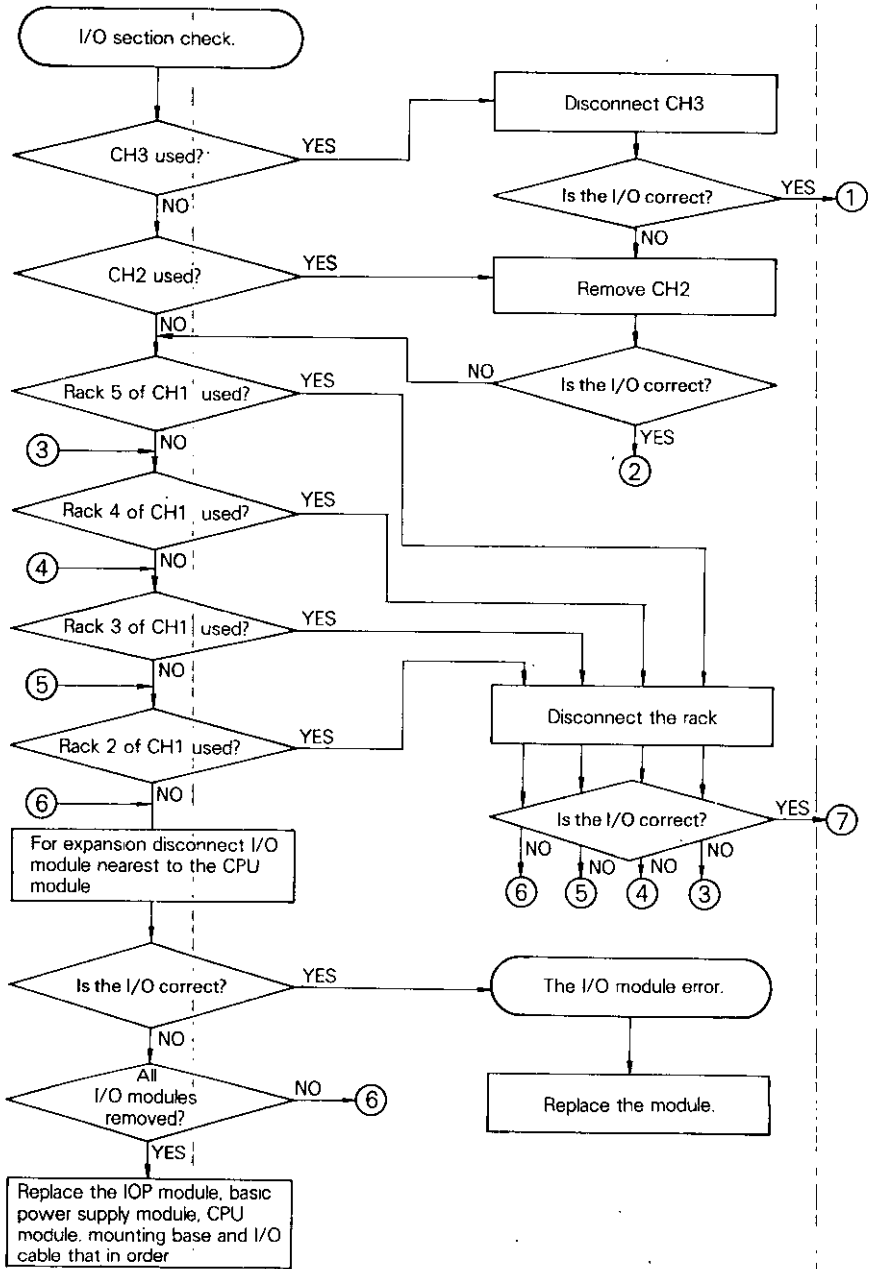


Fig. 9.20 (a) I/O Section Check

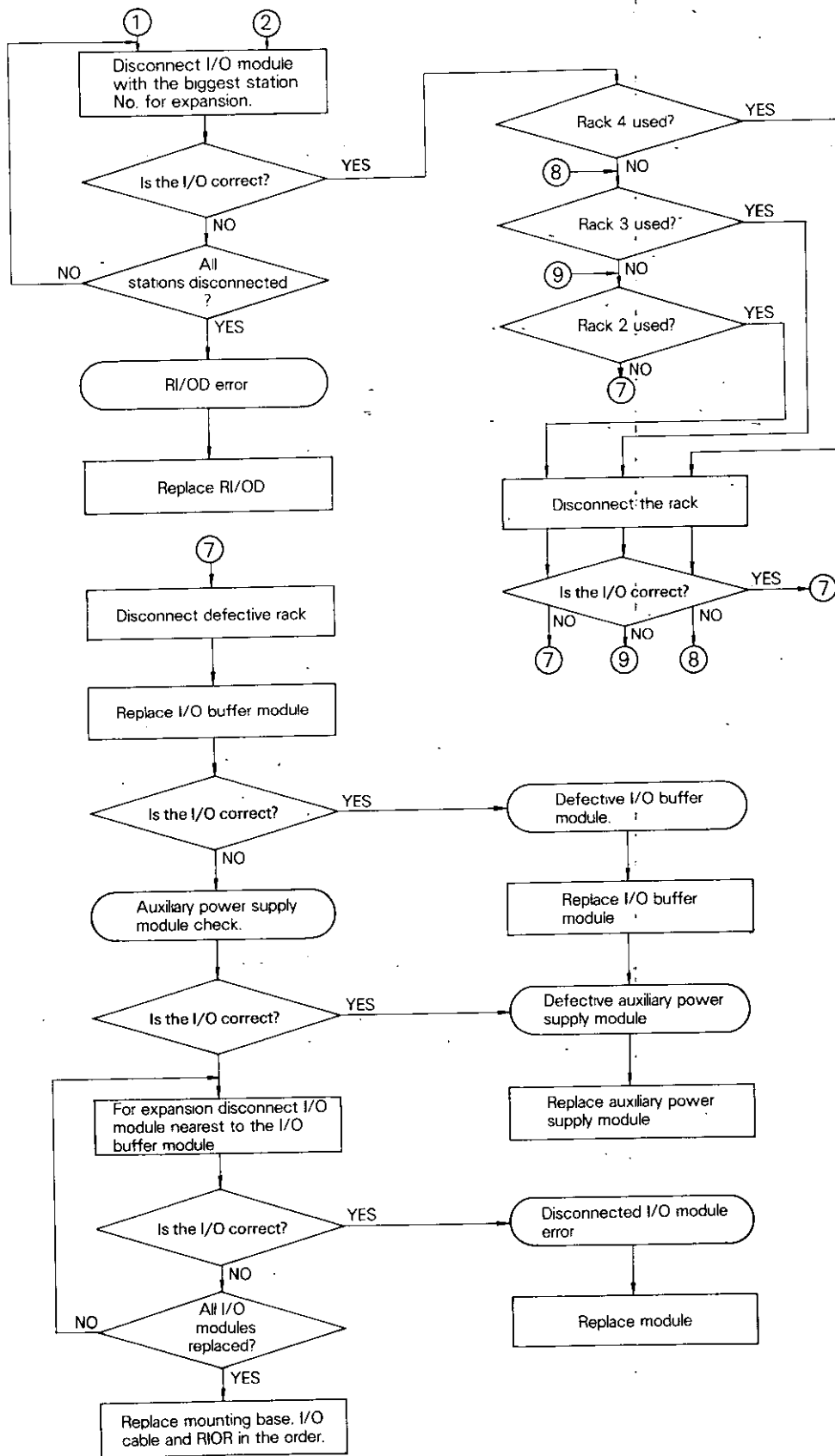


Fig. 9.20 (b) I/O Section Check.

(8) Input Module Check

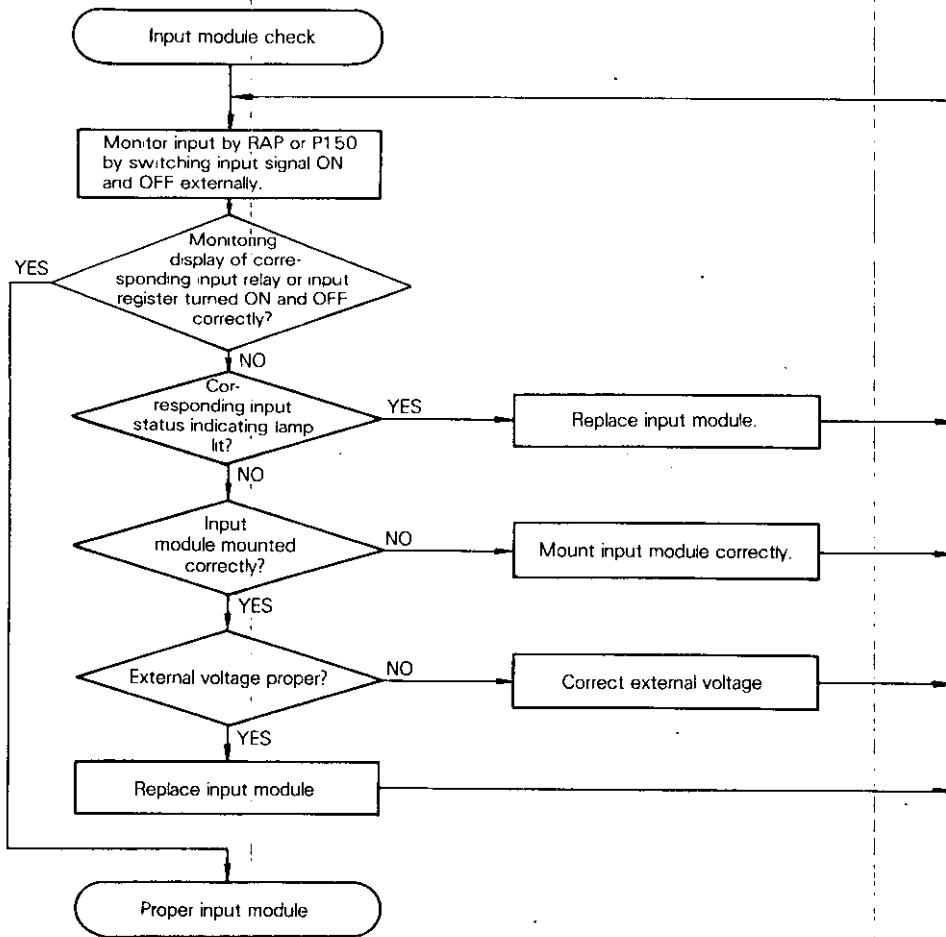


Fig. 9.21 Input Module Check

(9) Output Module Check

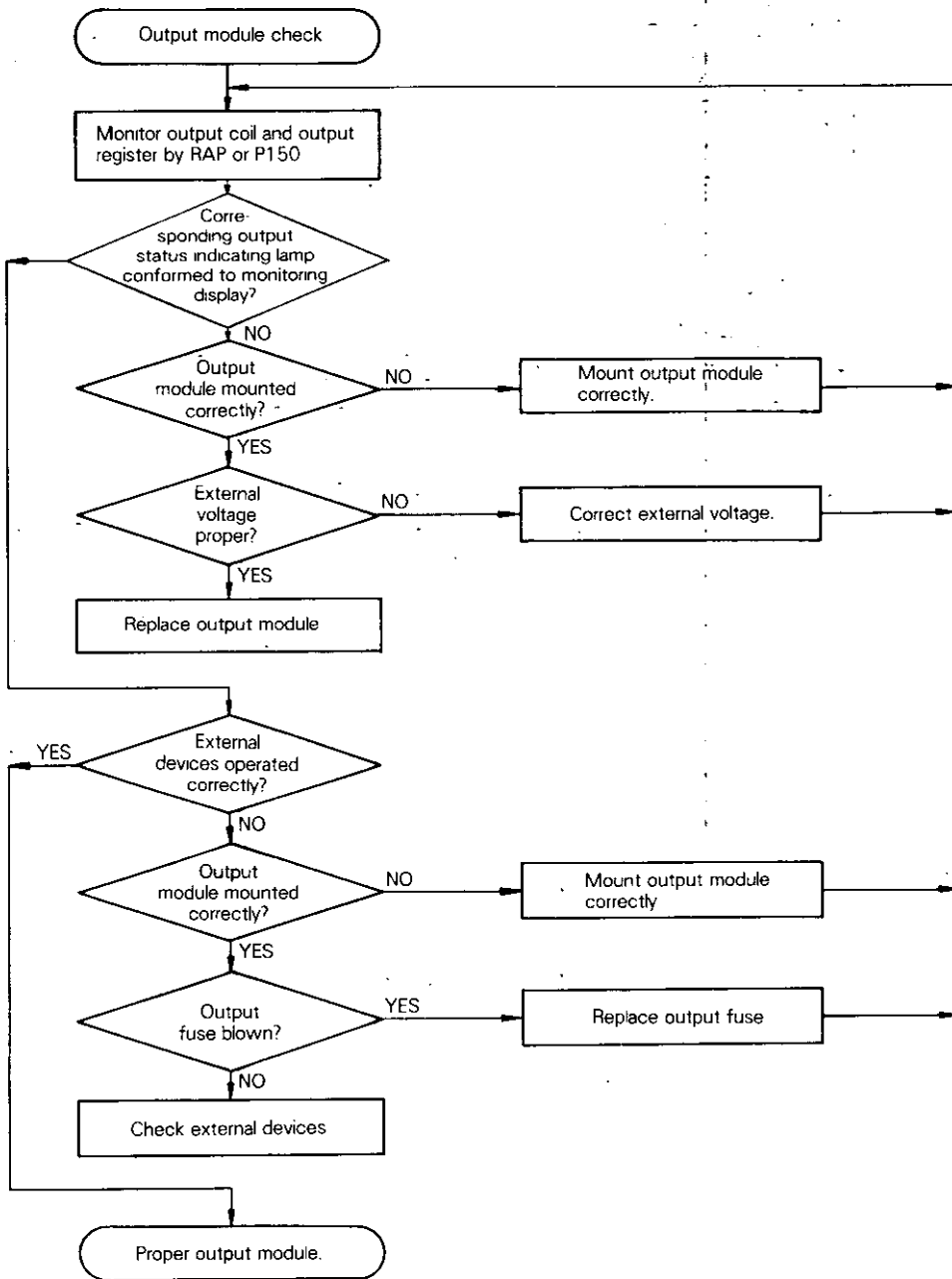


Fig. 9.22 Output Module Check

APPENDIX A

GL60H, GL70H COMPONENTS LIST

Component	Type	Function or Application	Remarks
CPU Module	DDSCR-GL70H	Program memory 64k-word (24 bits/word)	Mounted on M60
CPU Module	DDSCR-GL60H	Program memory 32k-word (24 bits/word)	Mounted on MB60
Main Power Supply Module	JRMSP-PS60	CPU, IOP, I/O modules (up to 6)	Mounted on MB60
Auxiliary Power Supply Module	JRMSP-PS21	B2100, I/O module (up to 9)	Mounted on MB21 or MB70
IOP Module	JAMSC-IF65	For communication module, P150, MEMOBUS RAP	Mounted on MB60
COMM Module	JAMSC-IF61	For communication module, P150, MEMOBUS	Mounted on MB60
RIOD Module	JAMSC-IF62A	Driver module for remote lines (up to 2)	Mounted on MB60
PC Link Module	JAMSC-IF64	For PC link communication interface	Mounted on MB60
3200 I/F Module	JAMSC-IF65	For YENET-3200 communication interface	Mounted on MB60
Servo I/F Module	JAMSC-IF66	For servo interface (up to 4)	Mounted on MB60
RIOR Module	JAMSC-IF70	Receiver module for remote lines	Mounted on MB70
ASCII Module	JAMSC-IF71	For ASCII device interface (up to 8)	Mounted on MB60, MB70, MB71 or MB21
FMGC Control Module	JAMSC-IF72	For FMGC control	Mounted on MB60
Register Access Panel (RAP)	DISCT-IF69	Monitor modules for status and parameter	Connected to IOP
I/O Buffer Module	JAMSC-B2110A	Used for I/O bus buffer, racks 2 to 5	With I/O cable connector
MB60 Mounting Base	JRMSI-MB60	For mounting CPU, power supply, IOP, COMM, RIOD and I/O modules	For rack 1
MB21 Mounting Base	JRMSI-MB21	For mounting I/O buffer, auxiliary power supply, I/O modules	For racks 2 to 5
MB70 Mounting Base	JRMSI-MB70	For mounting RIOR, auxiliary power supply, I/O modules	For remote station rack 1
I/O Cables	JZMSZ-W20-1	Connection between racks 0.5m long	
	JZMSZ-W20-2	Connection between racks 1.5m long	
I/O Modules	JAMSC-B2501	100VAC 16-point input, input current 10mA/100VAC, 60Hz	

Component	Type	Function or Application	Remarks
I/O Modules	JAMSC-B2503	200VAC 16-point input, input current 10mA/200VAC, 60Hz	
	JAMSC-B2505	100VAC 32-point input, input current 10mA/100VAC, 60Hz	
	JAMSC-B2507	200VAC 32-point input, input current 10mA/200VAC, 60Hz	
	JAMSC-B2601	12/24VDC 16-point input, input current 10mA/24VDC, 5mA/20VDC	
	JAMSC-B2603	12/24VDC, 32-point input, input current 10mA/24VDC, 5mA/12VDC	
	JAMSC-B2605	12/24VDC 64-point input	
	JAMSC-B2607	5VDC 32-point input	
	JAMSC-B2500	100/200VAC 16-point output, rated output current 1A/circuit, 3A/8 circuits	
	JAMSC-B2504	100/200VAC 32-point output, rated output current 0.3A/circuit, 1.2A/8 circuits	
	JAMSC-B2600	12/24VDC 16-point output, rated output current 2A/circuit, 5A/8 circuits	With status indicator
	JAMSC-B2602	12/24VDC 32-point output, rated output current 0.3A/circuit, 0.6A/4 circuits	With status indicator
	JAMSC-B2604	12/24VDC 64-point output	
	JAMSC-B2606	5VDC 32-point output	
	JAMSC-B2902	Relay contact 32-point output, relay coil voltage 24VDC, rated current 1A/circuit (220VAC), 1A/circuit (24VDC)	
	JAMSC-B2904	Bestact relay contact 16-point output (independent contact), relay coil voltage 24VDC, small load rated current 0.5A/circuit (220VAC), 0.3A/circuit (110VDC), min. rating 5V, 1mA	
	JAMSC-B2914	Bestact relay contact 16-point output (independent contact), relay coil voltage 24VDC, general-use type rated current 0.5A/circuit (220VAC), 0.3A/circuit (110VDC), min. rating 24V, 10mA	

Component		Type	Function or Application	Remarks
I/O Modules		JAMSC-B2701	Register input, numerical data (16-bit) × 8, strobe variable period (64/32ms)	
		JAMSC-B2700	Register output, numerical data (16-bit) × 8, strobe variable period (64/32ms)	
		JAMSC-B2703	Analog input (A/D) 0 to + 10V, 8 circuits	
		JAMSC-B2733	Analog input (A/D) - 10 to + 10V, 8 circuits	
		JAMSC-B2743	Analog input (A/D) 4 to 20mA, 8 circuits	
		JAMSC-B2702	Analog output (D/A) 2 circuits	
		JAMSC-B2712	Analog output (D/A) 0 to + 5V, 2 circuits	
		JAMSC-B2722	Analog output (D/A) -5 to + 5V, 2 circuits	
		JAMSC-B2732	Analog output (D/A) -10 to 10V, 2 circuits	
		JAMSC-B2742	Analog output (D/A) 4 to 20mA, 2 circuits	
		JAMSC-B2800	PID module circuit	
		JAMSC-B2801	Variable counter 2 circuits	
		JAMSC-B2802	Print counter 1 circuit	
		JAMSC-B2803	Positioning (Speed analog command type) 1-axis corresponding to absolute encoder	
		JAMSC-B2813	Positioning (speed pulse command type) 1-axis	
	JAMSC-B2804	Memo link master		
	JAMSC-B2805	Memo link slave		
Programming Panel		DISCT-P150-10	Plasma display, full-key type, 2FD	P150 system disk is needed.
		DISCT-P150-11	Plasma display, sheet key type, 2FD	P150 system disk is needed.
		DISCT-P140	Liquid crystal display, sheet key type	P140 ROM pack is needed.
Floppy Disk Unit		DISCT-FD400	Floppy disk unit for P140	Mounted on MB60
P150 Pro- gram Disk	Programmer	F70H-E001	Program creation, change, storing, monitor	Used in common with GL60H (pair)
	Ladder Lister	F70H-E002	For program print out	Used in common with GL60H
	Blank Disk	F150-000	Blank disk	Format completed
P140 ROM Pack		P70H-E001	Program creation, change, storing, monitor	



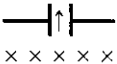
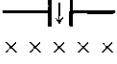
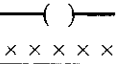
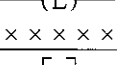

APPENDIX A

GL60H, GL70H COMPONENTS LIST (Cont'd)

Component	Type	Function or Application	Remarks
Remote I/O Junction Cable	JZMSZ-W60	From RIOD, RIOR to main cable (3C-2V)	
Remote I/O Main Cable	JZMSZ-W453	From RIOD, RIOR to main cable (5C-FB)	
Adapter	T-298	Repeater	
Interface Cable	JZMSZ-W1015-T1	For connection between P150, IOP-COMM module ports	2.5m long
	JZMSZ-W1015-T2		15m long
	JZMSZ-W1015-21	For connection between ACGC 400 series-GL60S, IOP-COMM module ports	2.5m long
	JZMSZ-W1015-22		15m long
	JZMSZ-W1017-T1	For connection between J1078 modem-GL60S, IOP-COMM module ports	5m long
	JZMSZ-W1017-T2		15m long
	JZ-MSZ-W1019-1	For connection between	5m long
	TZ-MSZ-W1019-2	U84/U84S-GL60	15m long

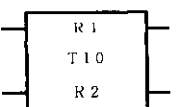
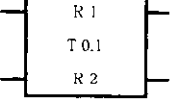
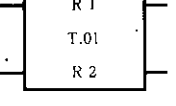
APPENDIX B OPERATION FUNCTION REFERENCE LIST

(1) Relay, Coil

Element Type		Specifiable Reference				
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit
 x x x x x	Normally Open Contact	Coil	00001 to 08192*	00001 to 08192*	00001 to 08192*	00001 to 08192*
		Input relay	10001 to 14096	10001 to 14096	10001 to 14096	10001 to 14096
		Step	S 001 to S 512	—	—	—
		Link coil	D 0001 to D 1024	—	—	—
 x x x x x	Normally Closed Contact	Coil	00001 to 08192	00001 to 08192	00001 to 08192	00001 to 08192
		Input relay	10001 to 14096	10001 to 14096	10001 to 14096	10001 to 14096
		Step	S 001 to S 512	—	—	—
		Link coil	D 0001 to D 1024	—	—	—
 x x x x x	Transitional Contact (OFF to ON)	Coil	00001 to 08192	00001 to 08192	00001 to 08192	00001 to 08192
		Input relay	10001 to 14096	10001 to 14096	10001 to 14096	10001 to 14096
		Step	S 001 to S 512	—	—	—
		Link coil	D 0001 to D 1024	—	—	—
 x x x x x	Transitional Contact (ON to OFF)	Coil	00001 to 08192	00001 to 08192	00001 to 08192	00001 to 08192
		Input relay	10001 to 14096	10001 to 14096	10001 to 14096	10001 to 14096
		Step	S 001 to S 512	—	—	—
		Link coil	D 0001 to D 1024	—	—	—
—	Horizontal	None				
I	Vertical Short-Circuit					
 x x x x x	Coil	Coil	00001 to 08191	00001 to 08191	—	00001 to 08191
		Link coil	D 0001 to D 1024	D 0001 to D 1024	—	D 0001 to D 1024
 x x x x x	Latch Coil	Coil	00001 to 08191	00001 to 08191	—	00001 to 08191
		Link coil	D 0001 to D 1024	D 0001 to D 1024	—	D 0001 to D 1024
 T x x x	Transition	Transition coil	—	—	T 001 to T 512	—

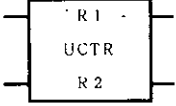
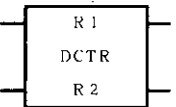
*Coil 08192 is battery coil.

(2) Timer

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	1-sec Timer	R 1	Constant	00000 to 09999	00000 to 09999	00000 to 09999	00000 to 09999
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
		R 2	Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
	0.1-sec Timer	R 1	Constant	00000 to 09999	00000 to 09999	00000 to 09999	00000 to 09999
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
		R 2	Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
	0.01-sec Timer	R 1	Constant	00000 to 09999	00000 to 09999	00000 to 09999	00000 to 09999
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
		R 2	Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999

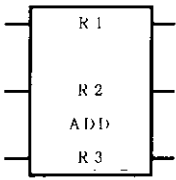
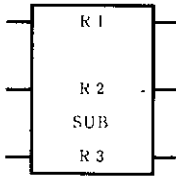
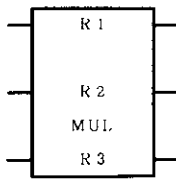
*Register

(3) Counter

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	Up-Counter	R 1	Constant	00000 to 09999	00000 to 09999	00000 to 09999	00000 to 09999
			Input R*	0001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		R 2	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
	Down-Counter	R 1	Constant	00000 to 09999	00000 to 09999	00000 to 09999	00000 to 09999
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		R 2	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024

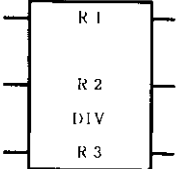
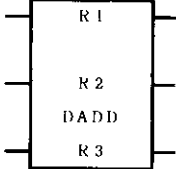
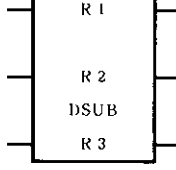
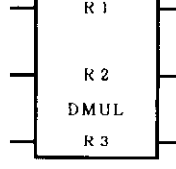
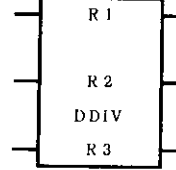
*Register

(4) Arithmetic Operations — 1

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	Addition	R 1	Constant	00000 to 09999	00000 to 09999	00000 to 09999	00000 to 09999
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
		R 2	Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Timer R*	50001 to 50512	50001 to 50512	50001 to 50512	50001 to 50512
R 3	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999		
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	
	Subtraction	R 1	Constant	00000 to 09999	00000 to 09999	00000 to 09999	00000 to 09999
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		R 2	Timer R*	50001 to 50512	50001 to 50512	50001 to 50512	50001 to 50512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
	Multiply	R 1	Constant	00000 to 09999	00000 to 09999	00000 to 09999	00000 to 09999
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
		R 2	Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Timer R*	50001 to 50512	50001 to 50512	50001 to 50512	50001 to 50512
R 3	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998		
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	

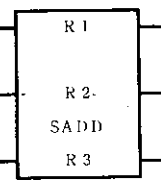
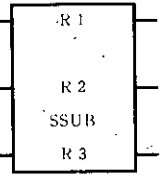
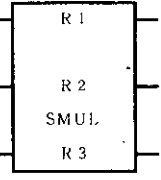
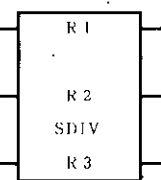
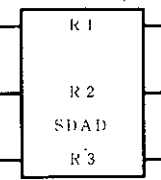
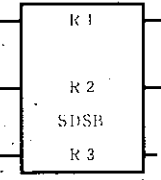
*Register

(4) Arithmetic Operations — 2

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	Divide	R 1	Constant	00000 to 09999	00000 to 09999	00000 to 09999	00000 to 09999
			Input R*	30001 to 30511	30001 to 30512	30001 to 30512	30001 to 30511
			Holding R*	40001 to 49998	40001 to 49999	40001 to 49999	40001 to 49998
			Constant R*	31001 to 35095	31001 to 35096	31001 to 35096	31001 to 35095
			Link R*	R 0001 to R 1023	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1023
			Timer R*	50001 to 50511	50001 to 50512	50001 to 50512	50001 to 50511
		R 2	Constant	00000 to 09999	00000 to 09999	00000 to 09999	00000 to 09999
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Timer R*	50001 to 50512	50001 to 50512	50001 to 50512	50001 to 50512
		R 3	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
			Double-Addition	R 1	Input R*	30001 to 30511	30001 to 30511
Holding R*	40001 to 49998				40001 to 49998	40001 to 49998	40001 to 49998
R 2	Constant R*			31001 to 35095	31001 to 35095	31001 to 35095	31001 to 35095
	Link R*			R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
R 3	Holding R*			40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
	Link R*			R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
	Double-Subtraction	R 1	Input R*	30001 to 30511	30001 to 30511	30001 to 30511	30001 to 30511
			Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
		R 2	Constant R*	31001 to 35095	31001 to 35095	31001 to 35095	31001 to 35095
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
		R 3	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
	Double-Multiply	R 1	Input R*	30001 to 30511	30001 to 30511	30001 to 30511	30001 to 30512
			Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
		R 2	Constant R*	31001 to 35095	31001 to 35095	31001 to 35095	31001 to 35095
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
		R 3	Holding R*	40001 to 49996	40001 to 49996	40001 to 49996	40001 to 49996
			Link R*	R 0001 to R 1021	R 0001 to R 1021	R 0001 to R 1021	R 0001 to R 1021
	Double-Divide	R 1	Input R*	30001 to 30509	30001 to 30509	30001 to 30509	30001 to 30509
			Holding R*	40001 to 49996	40001 to 49996	40001 to 49996	40001 to 49996
			Constant R*	31001 to 35093	31001 to 35093	31001 to 35093	31001 to 35093
			Link R*	R 0001 to R 1021	R 0001 to R 1021	R 0001 to R 1021	R 0001 to R 1021
		R 2	Input R*	30001 to 30511	30001 to 30511	30001 to 30511	30001 to 30511
			Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Constant R*	31001 to 35095	31001 to 35095	31001 to 35095	31001 to 35095
		R 3	Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
			Holding R*	40001 to 49996	40001 to 49996	40001 to 49996	40001 to 49996
			Link R*	R 0001 to R 1021	R 0001 to R 1021	R 0001 to R 1021	R 0001 to R 1021

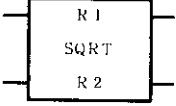
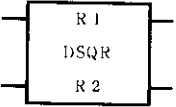
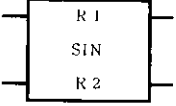
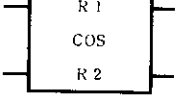
*Register

(5) Signed Arithmetic Operation

Element Type		Specifiable Circuit								
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine				
	Signed Addition	R 1	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512			
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999			
		R 2	Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096			
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024			
		R 3	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999			
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024			
	Signed Subtraction	R 1	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512			
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999			
		R 2	Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096			
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024			
		R 3	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999			
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024			
	Signed Multiply	R 1	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512			
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999			
		R 2	Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096			
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024			
		R 3	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998			
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023			
	Signed Divide	R 1	Input R*	30001 to 30511	30001 to 30512	30001 to 30512	30001 to 30512			
			Holding R*	40001 to 49998	40001 to 49999	40001 to 49999	40001 to 49999			
			Constant R*	31001 to 35095	31001 to 35065	31001 to 35096	31001 to 35096			
			Link R*	R 0001 to R 1023	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024			
		R 2	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512			
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999			
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096			
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024			
		R 3	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998			
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023			
				Signed Double-Precision Addition	R 1	Input R*	30001 to 30511	30001 to 30511	30001 to 30511	30001 to 30511
						Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
R 2	Constant R*	31001 to 35095			31001 to 35095	31001 to 35095	31001 to 35095			
	Link R*	R 0001 to R 1023			R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023			
R 3	Holding R*	40001 to 49998			40001 to 49998	40001 to 49998	40001 to 49998			
	Link R*	R 0001 to R 1023			R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023			
	Signed Double-Precision Subtraction	R 1	Input R*	30001 to 30511	30001 to 30511	30001 to 30511	30001 to 30511			
			Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998			
		R 2	Constant R*	31001 to 35095	31001 to 35095	31001 to 35095	31001 to 35095			
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023			
		R 3	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998			
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023			

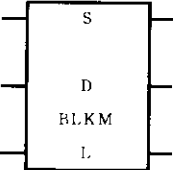
* Register

(6) Operations of Square Root and Trigonometric Function

Element Type		Specifiable Reference						
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit		
	Square Root	R 1	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512	
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999	
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096	
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	
		R 2	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
	Double-Precision Square Root	R 1	Input R*	30001 to 30511	30001 to 30511	30001 to 30512	30001 to 30512	
			Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998	
			Constant R*	31001 to 35095	31001 to 35095	31001 to 35095	31001 to 35095	
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	
		R 2	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
	Sine	R 1	Input R*	30001 to 30511	30001 to 30511	30001 to 30511	30001 to 30511	
			Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998	
			Constant R*	31001 to 35095	31001 to 35095	31001 to 35095	31001 to 35095	
			Link R*	R 0001 to R 1023	R 0001 to R 102	R 0001 to R 102	R 0001 to R 102	
		R 2	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49995	40001 to 49995
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 102	R 0001 to R 102
	Cosine	R 1	Input R*	30001 to 30511	30001 to 30511	30001 to 30511	30001 to 30511	
			Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998	
			Constant R*	31001 to 35095	31001 to 35095	31001 to 35095	31001 to 35095	
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	
		R 2	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023

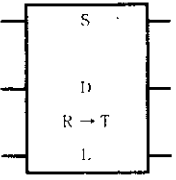
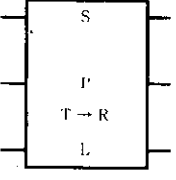
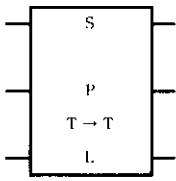
* Register

(7) Data Transfer - 1

Element Type		Specifiable Reference						
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit		
	Block Move	S	Coil G†	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177	
			Input relay G†	10001 to 14081	10001 to 14081	10001 to 14081	10001 to 14081	
			Step relay G†	S 001 to S 497	S 001 to S 497	S 001 to S 497	S 001 to S 497	
			Link Coil G†	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512	
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999	
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096	
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	
		D	Timer R*	50001 to 50512	50001 to 50512	50001 to 50512	50001 to 50512	50001 to 50512
			Coil G†	00001 to 08177	—	—	—	00001 to 08177
			Link Coil G†	D 0001 to D 1024	—	—	—	D 0001 to D 1009
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		L	Table length	1 to 100	1 to 100	1 to 100	1 to 100	1 to 100
			Note: Discrete group table length	1 to 32 (Step relay)	1 to 64 (Link coil)	1 to 64 (Link coil)	1 to 64 (Link coil)	1 to 64 (Link coil)
			Note: Discrete group table length	1 to 32 (Step relay)	1 to 64 (Link coil)	1 to 64 (Link coil)	1 to 64 (Link coil)	1 to 64 (Link coil)
			Note: Discrete group table length	1 to 32 (Step relay)	1 to 64 (Link coil)	1 to 64 (Link coil)	1 to 64 (Link coil)	1 to 64 (Link coil)

* Register † Group

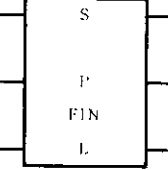
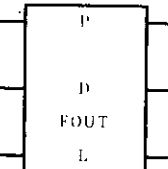
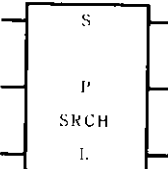
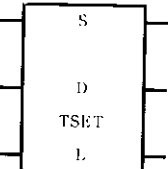
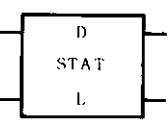
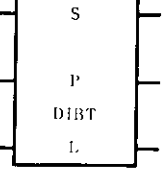
(7) Data Transfer - 2

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	Register-to-Table Move	S	Coil G†	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177
			Input relay G†	10001 to 14081	10001 to 14081	10001 to 14081	10001 to 14081
			Step relay G†	S 001 to S 497	S 001 to S 497	S 001 to S 497	S 001 to S 497
			Link coil G†	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009
			Input R*-	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		D	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
		L	Table length	1 to 999	1 to 999	1 to 999	1 to 999
			Table-to-Register Move	S	Coil G†	00001 to 08177	00001 to 08177
Input relay G†	10001 to 14081				10001 to 14081	10001 to 14081	10001 to 14081
Step relay G†	S 001 to S 497				S 001 to S 497	S 001 to S 497	S 001 to S 497
Link coil G†	D 0001 to D 1009				D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009
Input R*	30001 to 30512				30001 to 30512	30001 to 30512	30001 to 30512
Holding R*	40001 to 49999				40001 to 49999	40001 to 49999	40001 to 49999
Constant R*	31001 to 35096				31001 to 35096	31001 to 35096	31001 to 35096
Link R*	R 0001 to R 1024				R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
P	Holding R*			40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
	Link R*			R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
L	Table length			1 to 999	1 to 999	1 to 999	1 to 999
	Table-to-Table Move			S	Coil G†	00001 to 08177	00001 to 08177
		Input relay G†	10001 to 14081		10001 to 14081	10001 to 14081	10001 to 14081
		Step relay G†	S 001 to S 497		S 001 to S 497	S 001 to S 497	S 001 to S 497
		Link coil G†	D 0001 to D 1009		D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009
		Input R*	30001 to 30512		30001 to 30512	30001 to 30512	30001 to 30512
		Holding R*	40001 to 49999		40001 to 49999	40001 to 49999	40001 to 49999
		Constant R*	31001 to 35096		31001 to 35096	31001 to 35096	31001 to 35096
		Link R*	R 0001 to R 1024		R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		P	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
		L	Table length	1 to 999	1 to 999	1 to 999	1 to 999

* Register † Group

- NOTE Discrete group table length-
- 1 to 512 (Coil group)
 - 1 to 256 (Input relay group)
 - 1 to 32 (Step relay group)
 - 1 to 64 (Link coil group)

(7) Data Transfer - 3; Block Move With Index - 1

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	First In	S	Coil G**	0001 to 08177	0001 to 08177	0001 to 08177	0001 to 08177
			Input relay G**	1001 to 14081	1001 to 14081	1001 to 14081	1001 to 14081
			Step relay G**	S 001 to S 497	S 001 to S 497	S 001 to S 497	S 001 to S 497
			Link coil G**	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009
			Input R*	3001 to 30512	3001 to 30512	3001 to 30512	3001 to 30512
			Holding R*	4001 to 49999	4001 to 49999	4001 to 49999	4001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Timer R*	50001 to 50512	50001 to 50512	50001 to 50512	50001 to 50512
		P	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
L	Table length	1 to 100	1 to 100	1 to 100	1 to 100		
	First Out	P	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
		D	Coil G**	00001 to 08177	—	—	00001 to 08177
			Link coil G**	D 0001 to D 1009	—	—	D 0001 to D 1009
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
		L	Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
Table length	1 to 100		1 to 100	1 to 100	1 to 100		
	Table Search	S	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		P	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
L	Table length	1 to 100	1 to 100	1 to 100	1 to 100		
	Table Set	S	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		D	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
L	Table length	1 to 100	1 to 100	1 to 100	1 to 100		
	Get Controller System Status	D	Coil G**	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177
			Link coil G**	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
L	Table length	1 to 128	1 to 128	1 to 128	1 to 128		
	Block Move 1 with Destination Index	S	Coil G**	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177
			Input relay G**	10001 to 14081	10001 to 14081	10001 to 14081	10001 to 14081
			Step relay G**	S 001 to S 497	S 001 to S 497	S 001 to S 497	S 001 to S 497
			Link coil G**	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Timer R*	50001 to 50512	50001 to 50512	50001 to 50512	50001 to 50512
		P	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
L	Table length	1 to 100*	1 to 100†	1 to 100†	1 to 100†		

* Register ** Group
 † Discrete group table length:
 · 1 to 32 (Step relay)
 · 1 to 64 (Link coil)

(8) Block Move with Index-2

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	Block Move 2 with Destination Index	S	Coil G**	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177
			Input relay G**	10001 to 14081	10001 to 14081	10001 to 14081	10001 to 14081
			Step relay G**	S 001 to S 497	S 001 to S 497	S 001 to S 497	S 001 to S 497
			Link coil G**	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Timer R*	50001 to 50512	50001 to 50512	50001 to 50512	50001 to 50512
		P	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
L	Table length	1 to 100†	1 to 100†	1 to 100†	1 to 100†		
	Block Move 1 with Source Index	P	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		D	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		L	Table length	1 to 100	1 to 100	1 to 100	1 to 100
	Block Move 1 with Source Index 2	P	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		D	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		L	Table length	1 to 100	1 to 100	1 to 100	1 to 100

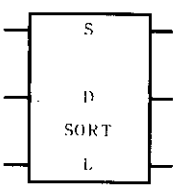
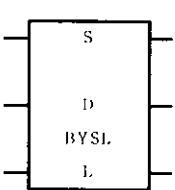
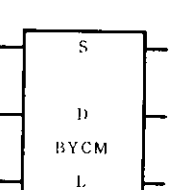
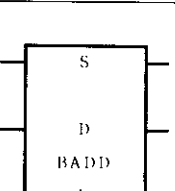
* Register ** Group
 † Discrete group table length:
 · 1 to 32 (Step relay)
 · 1 to 64 (Link coil)

(9) Data Convert - 1

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	BCD to BIN	S	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		P	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
L	Table length	1 to 16	1 to 16	1 to 16	1 to 16		
	BIN to BCD	S	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		P	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
L	Table length	1 to 16	1 to 16	1 to 16	1 to 16		
	Swap	S	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		D	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
L	Table length	1 to 100	1 to 100	1 to 100	1 to 100		

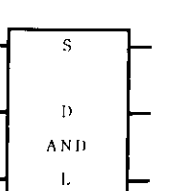
* Register

(9) Data Convert - 2

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	Sort	S	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		D	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		L	Table length	1 to 100	1 to 100	1 to 100	1 to 100
	Byte Solution	S	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		D	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
		L	Table length	1 to 100	1 to 100	1 to 100	1 to 100
	Byte Combination	S	Input R*	30001 to 30511	30001 to 30511	30001 to 30511	30001 to 30511
			Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Constant R*	31001 to 35095	31001 to 35095	31001 to 35095	31001 to 35095
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
		D	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		L	Table length	1 to 100	1 to 100	1 to 100	1 to 100
	Block Addition	S	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		D	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998	40001 to 49998
			Link R*	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023	R 0001 to R 1023
		L	Table length	1 to 100	1 to 100	1 to 100	1 to 100

* Register

(10) Matrix - 1

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	Logical AND	S	Coil G**	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177
			Input relay G**	10001 to 14081	10001 to 14081	10001 to 14081	10001 to 14081
			Step relay G**	S 001 to S 497	S 001 to S 497	S 001 to S 497	S 001 to S 497
			Link coil G**	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Timer R*	50001 to 50512	50001 to 50512	50001 to 50512	50001 to 50512
			D	Coil G**	00001 to 08177	—	—
		Link coil G**		D 0001 to D 1009	—	—	D 0001 to D 1009
		Holding R*		40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
		Link R*		R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		L	Table length	1 to 100†	1 to 100†	1 to 100†	1 to 100†

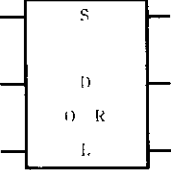
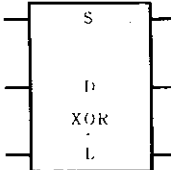
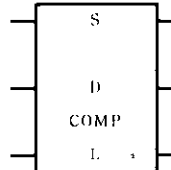
* Register ** Group

† Discrete group table length:

· 1 to 32 (Step relay)

· 1 to 64 (Link coil)

(10) Matrix-2

Element Type		Specifiable Reference							
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit			
	Logical OR	S	Coil G**	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177		
			Input relay G**	10001 to 14081	10001 to 14081	10001 to 14081	10001 to 14081		
			Step relay G**	S 001 to S 497	S 001 to S 497	S 001 to S 497	S 001 to S 497		
			Link coil G**	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009		
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512		
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999		
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096		
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024		
		D	Coil G**	00001 to 08177	————	————	————	00001 to 08177	
			Link coil G**	D 0001 to D 1009	————	————	————	D 0001 to D 1009	
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999	
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	
		L	Table length	1 to 100†	1 to 100†	1 to 100†	1 to 100†	1 to 100†	
			Logical Exclusive OR	S	Coil G**	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177
					Input relay G**	10001 to 14081	10001 to 14081	10001 to 14081	10001 to 14081
Step relay G**	S 001 to S 497				S 001 to S 497	S 001 to S 497	S 001 to S 497		
Link coil G**	D 0001 to D 1009				D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009		
Input R*	30001 to 30512				30001 to 30512	30001 to 30512	30001 to 30512		
Holding R*	40001 to 49999				40001 to 49999	40001 to 49999	40001 to 49999		
Constant R*	31001 to 35096				31001 to 35096	31001 to 35096	31001 to 35096		
Link R*	R 0001 to R 1024				R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024		
D	Coil G**			00001 to 08177	————	————	————	00001 to 08177	
	Link coil G**			D 0001 to D 1009	————	————	————	D 0001 to D 1009	
	Holding R*			40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999	
	Link R*			R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	
L	Table length			1 to 100†	1 to 100†	1 to 100†	1 to 100†	1 to 100†	
	Logical Complement			S	Coil G**	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177
					Input relay G**	10001 to 14081	10001 to 14081	10001 to 14081	10001 to 14081
		Step relay G**	S 001 to S 497		S 001 to S 497	S 001 to S 497	S 001 to S 497		
		Link coil G**	D 0001 to D 1009		D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009		
		Input R*	30001 to 30512		30001 to 30512	30001 to 30512	30001 to 30512		
		Holding R*	40001 to 49999		40001 to 49999	40001 to 49999	40001 to 49999		
		Constant R*	31001 to 35096		31001 to 35096	31001 to 35096	31001 to 35096		
		Link R*	R 0001 to R 1024		R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024		
		D	Coil G**	00001 to 08177	————	————	————	00001 to 08177	
			Link coil G**	D 0001 to D 1009	————	————	————	D 0001 to D 1009	
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999	
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	
		L	Table length	1 to 100†	1 to 100†	1 to 100†	1 to 100†	1 to 100†	

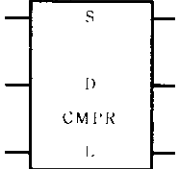
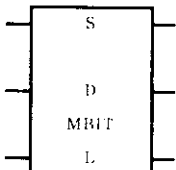
* Register ** Group

† Discrete group table length:

· 1 to 32 (Step relay)

· 1 to 64 (Link coil)

(10) Matrix - 3

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	Logical Compare	S	Coil G**	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177
			Input relay G**	10001 to 14081	10001 to 14081	10001 to 14081	10001 to 14081
			Step relay G**	S 001 to S 497	S 001 to S 497	S 001 to S 497	S 001 to S 497
			Link coil G**	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Timer R*	50001 to 50512	50001 to 50512	50001 to 50512	50001 to 50512
			D	Holding R*	40001 to 49998	40001 to 49998	40001 to 49998
Link R*	R 0001 to R 1023	R 0001 to R 1023		R 0001 to R 1023	R 0001 to R 1023		
L	Table length	1 to 100†	1 to 100†	1 to 100†	1 to 100†		
	Logical Bit Modify	S	Constant	1 to 9600	1 to 9600	1 to 9600	1 to 9600
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		D	Coil G**	00001 to 08177	————	————	00001 to 08177
			Link coil G**	D 0001 to D 1009	————	————	D 0001 to D 1009
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
		L	Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Table length	1 to 600*	1 to 600	1 to 600	1 to 600

* Register ** Group

† Discrete group table length:

- 1 to 32 (Step relay)
- 1 to 64 (Link coil)

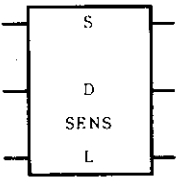
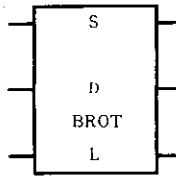
Discrete group table length:

- 1 to 64 (Link coil)

† Discrete group table length:

- 1 to 512 (Coil)
- 1 to 64 (Link coil)

(10) Matrix-4

Element Type		Specifiable Reference						
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit		
	Logical Bit Sense	S	Constant	1 to 9600	1 to 9600	1 to 9600	1 to 9600	
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512	
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999	
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	
		D	Coil G**	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177	00001 to 08177
			Input relay G**	10001 to 14081	10001 to 14081	10001 to 14081	10001 to 14081	10001 to 14081
			Link coil G**	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009
			Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
		L	Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
			Table length	1 to 600†	1 to 600†	1 to 600†	1 to 600†	1 to 600†
			Logical Bit Rotate	S	Coil G**	00001 to 08177	00001 to 08177	00001 to 08177
Input relay G**	10001 to 14081				10001 to 14081	10001 to 14081	10001 to 14081	
Link coil G**	D 0001 to D 1009				D 0001 to D 1009	D 0001 to D 1009	D 0001 to D 1009	
Input R*	30001 to 30512				30001 to 30512	30001 to 30512	30001 to 30512	
Holding R*	40001 to 49999				40001 to 49999	40001 to 49999	40001 to 49999	
Constant R*	31001 to 35096				31001 to 35096	31001 to 35096	31001 to 35096	
Link R*	R 0001 to R 1024				R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	
D	Coil G**			00001 to 08177	————	————	————	00001 to 08177
	Link coil G**			D 0001 to D 1009	————	————	————	D 0001 to D 1009
	Holding R*			40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
	Link R*			R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
L	Table length			1 to 100†	1 to 100†	1 to 100†	1 to 100†	1 to 100†

* Register ** Group

† Discrete group table length: † Discrete group table length:

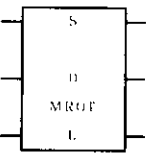
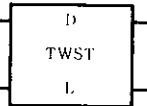
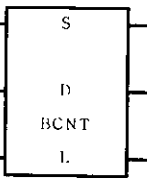
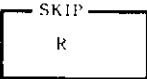
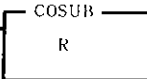
· 1 to 512 (Coil)

· 1 to 64 (Link coil)

· 1 to 256 (I/O relay)

· 1 to 64 (Link coil)

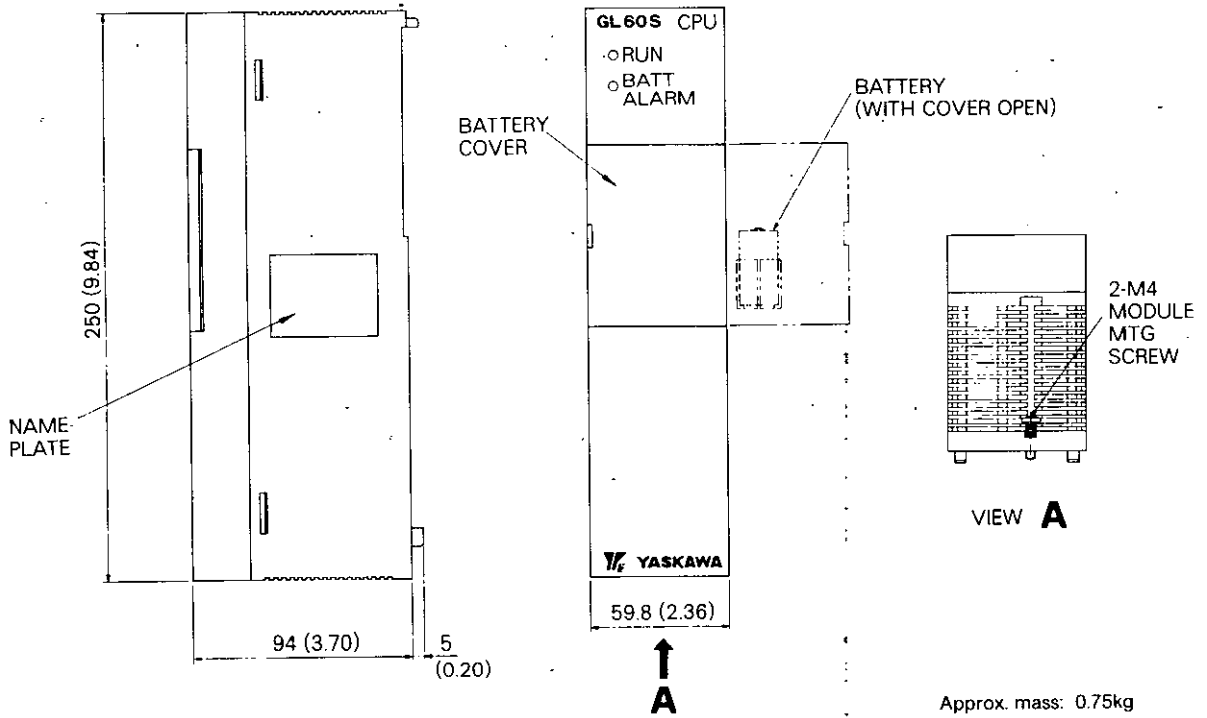
(10) Matrix-5, Skip, Subroutine

Element Type		Specifiable Reference					
Symbol	Name	Reference	Ladder Circuit	Action Circuit	Transition	Subroutine Circuit	
	Logical Multi-Bit Rotate	S	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		D	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
L	Table length	1 to 100	1 to 100	1 to 100	1 to 100		
	Logical Twist	D	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		L	Table length	1 to 100	1 to 100	1 to 100	1 to 100
	Logical Bit Count	S	Input R*	30001 to 30512	30001 to 30512	30001 to 30512	30001 to 30512
			Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Constant R*	31001 to 35096	31001 to 35096	31001 to 35096	31001 to 35096
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
		D	Holding R*	40001 to 49999	40001 to 49999	40001 to 49999	40001 to 49999
			Link R*	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024	R 0001 to R 1024
L	Table length	1 to 100	1 to 100	1 to 100	1 to 100		
	Skip	R	Constant	1 to 9999			
			Input R*	30001 to 30512			
			Holding R*	40001 to 49999	—	—	—
			Constant R*	31001 to 35096			
			Link R*	R 0001 to R 1024			
	Subroutine	R	Constant	G 00 to G 99	—	—	—

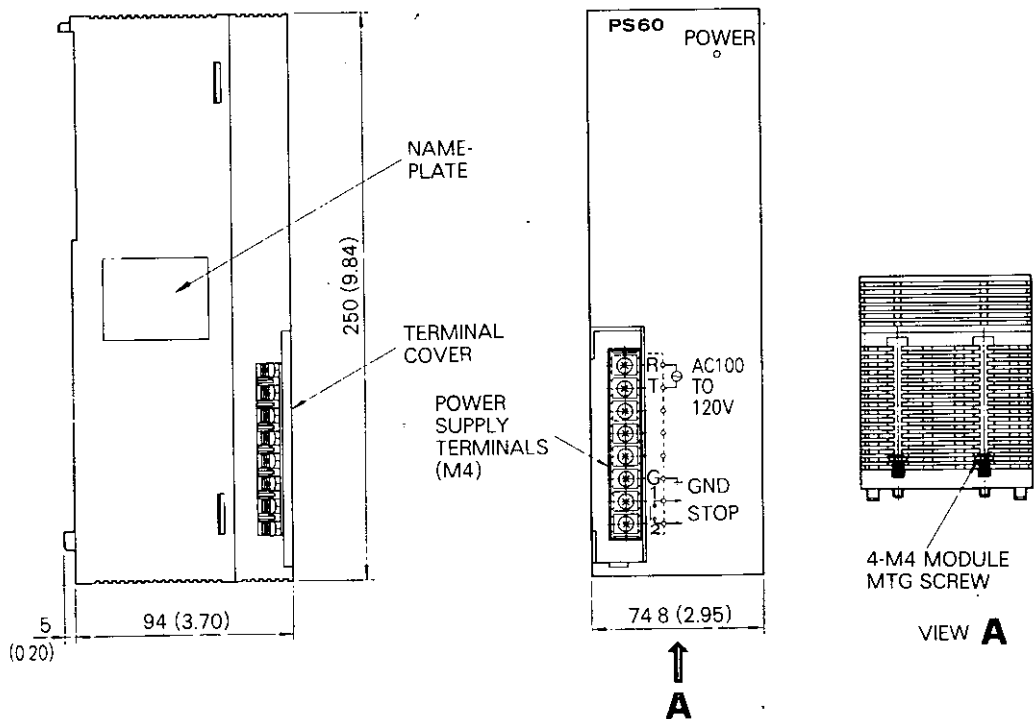
* Register

APPENDIX C Dimensions in mm (inch)

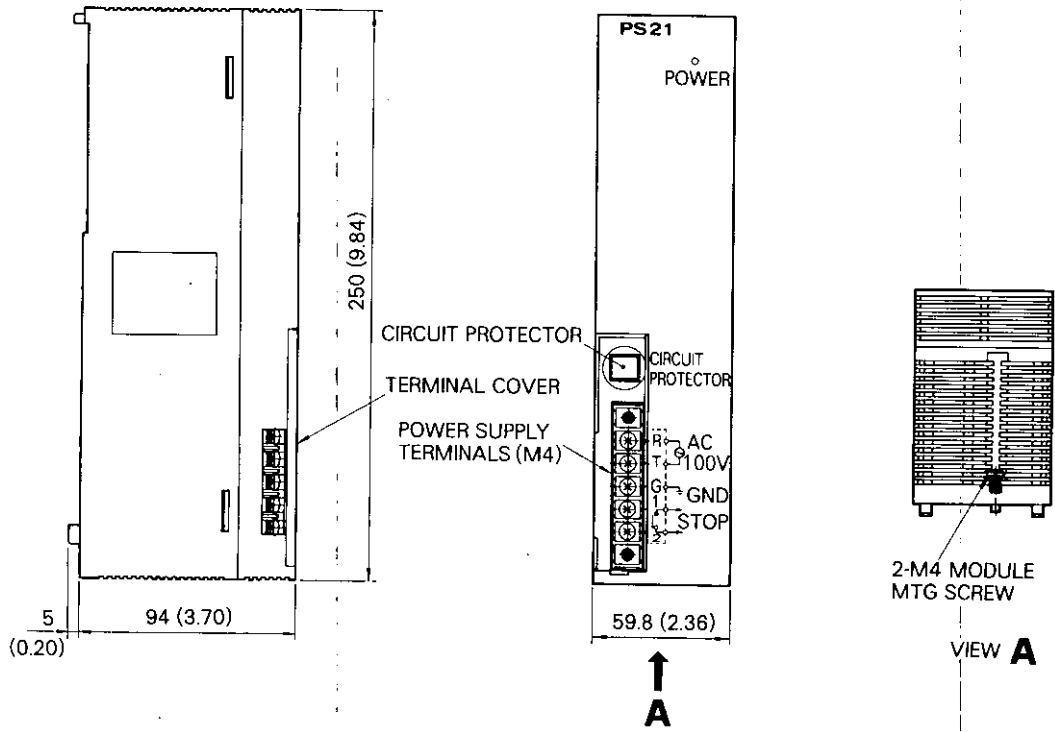
(1) CPU Module (Type DDSCR-GL 60 H, GL70H)



(2) Main Power Supply Module (Type JRMSP-PS 60)

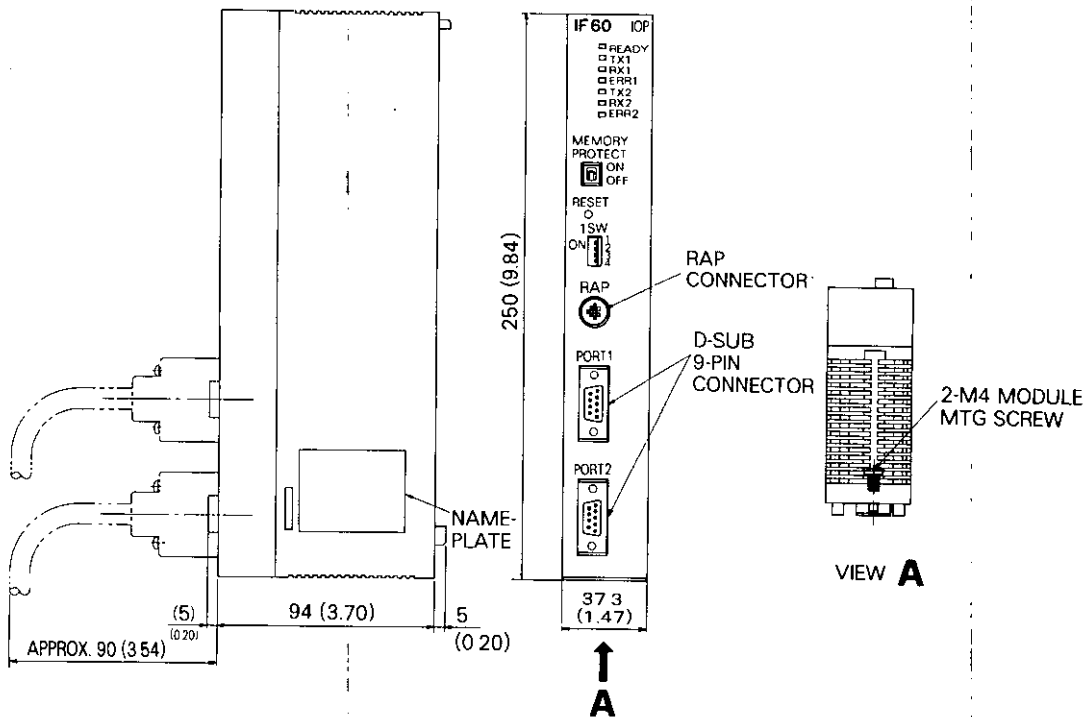


(3) Auxiliary Power Supply Module (Type JRMSP-PS 21)



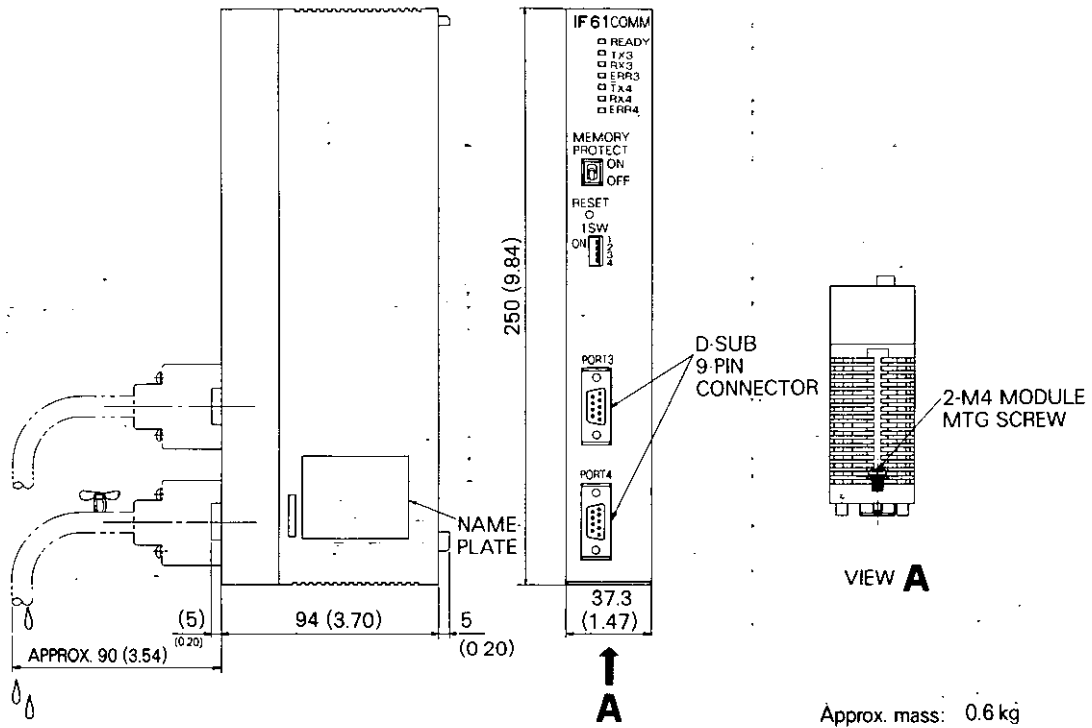
Approx. mass: 0.7 kg

(4) IOP Module (Type JAMSC-IF 60)

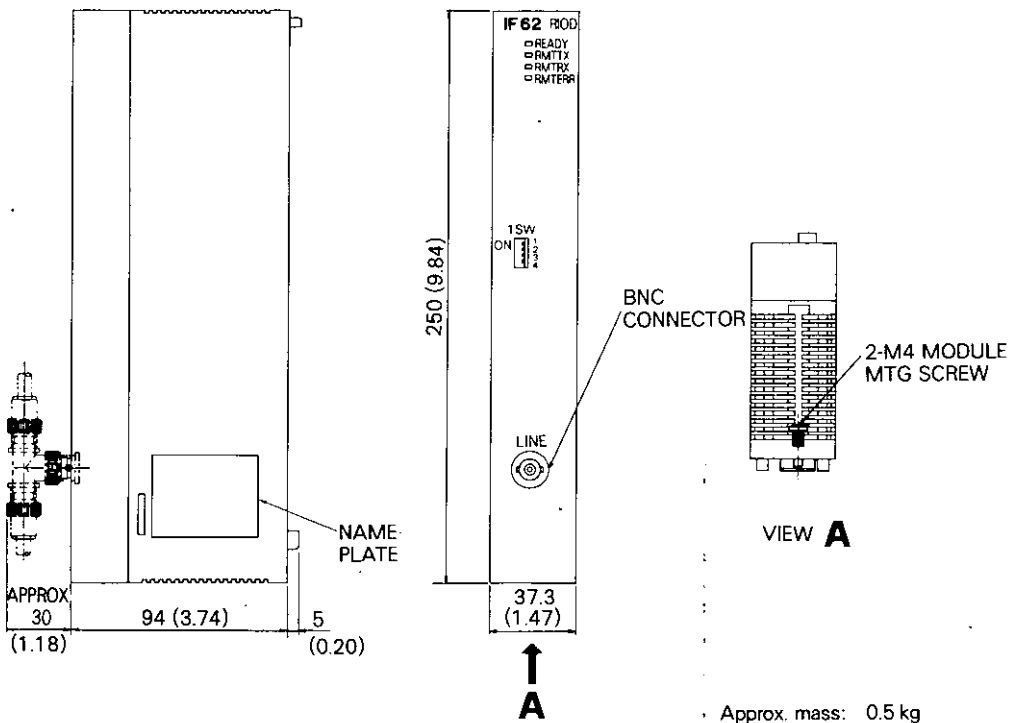


Approx. mass: 0.6 kg

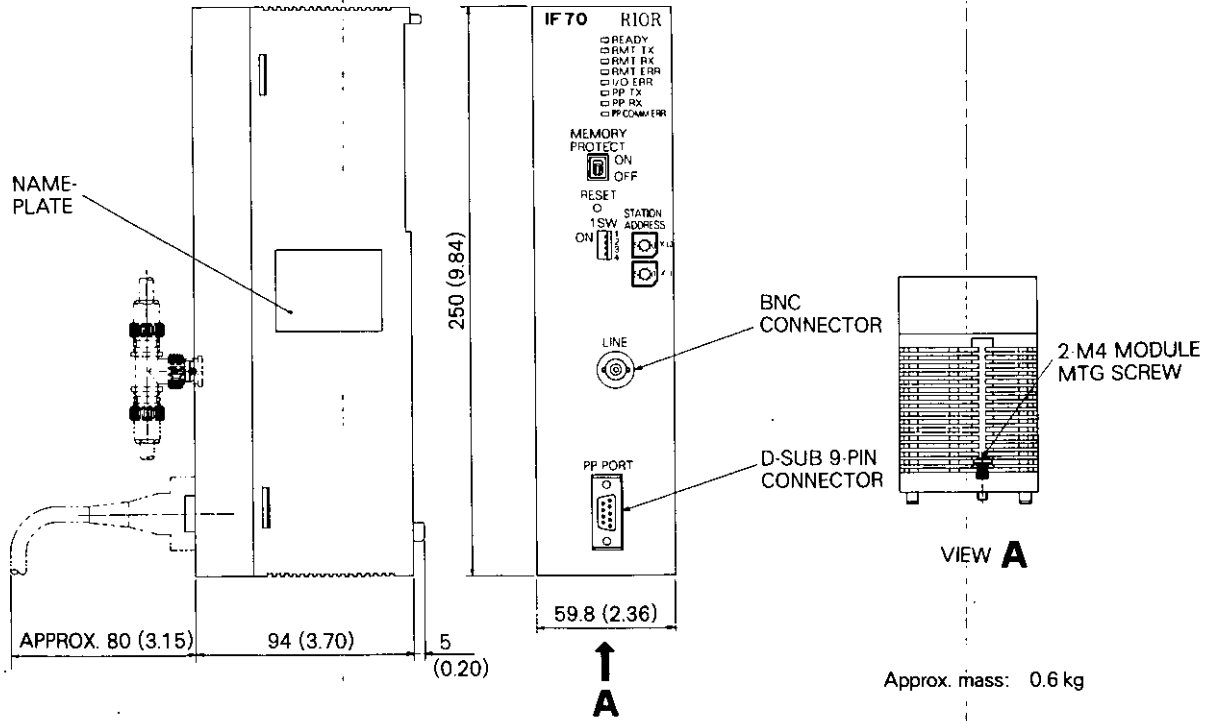
(5) COMM Module (Type JAMSC-IF 61)



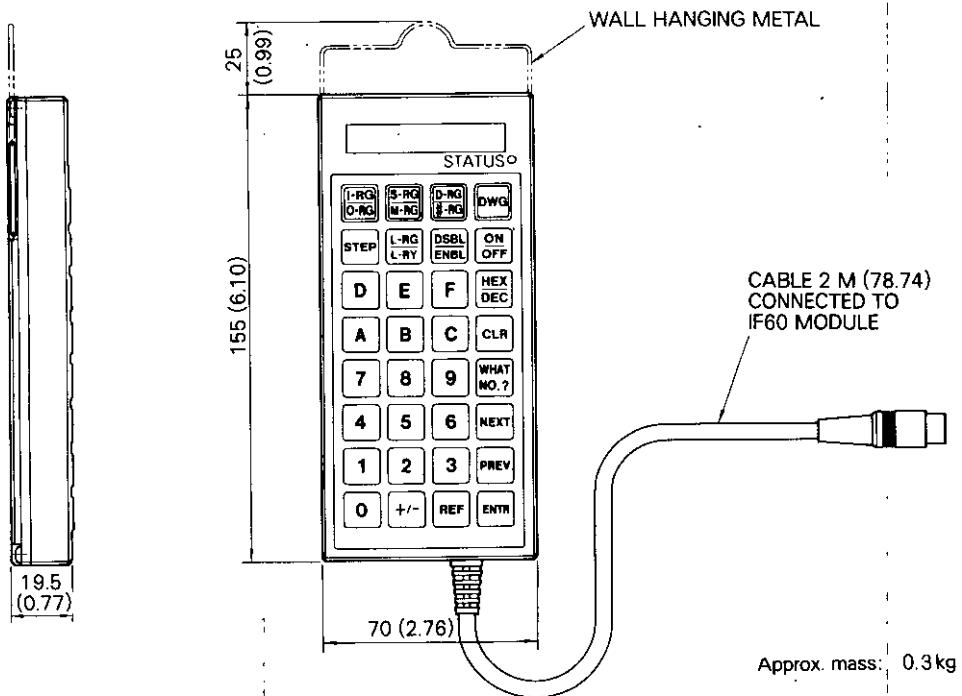
(6) RIOD Module (Type JAMSC-IF 62)



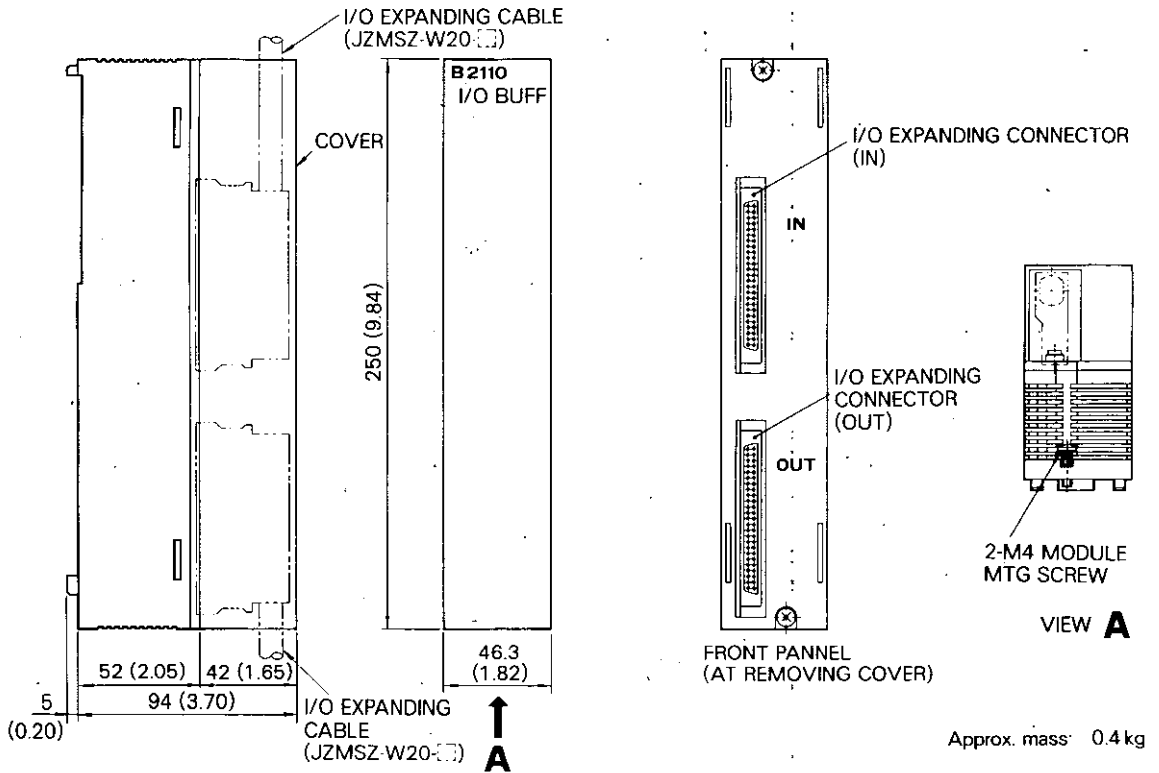
(7) RIOR Module (Type JAMSC-IF 70)



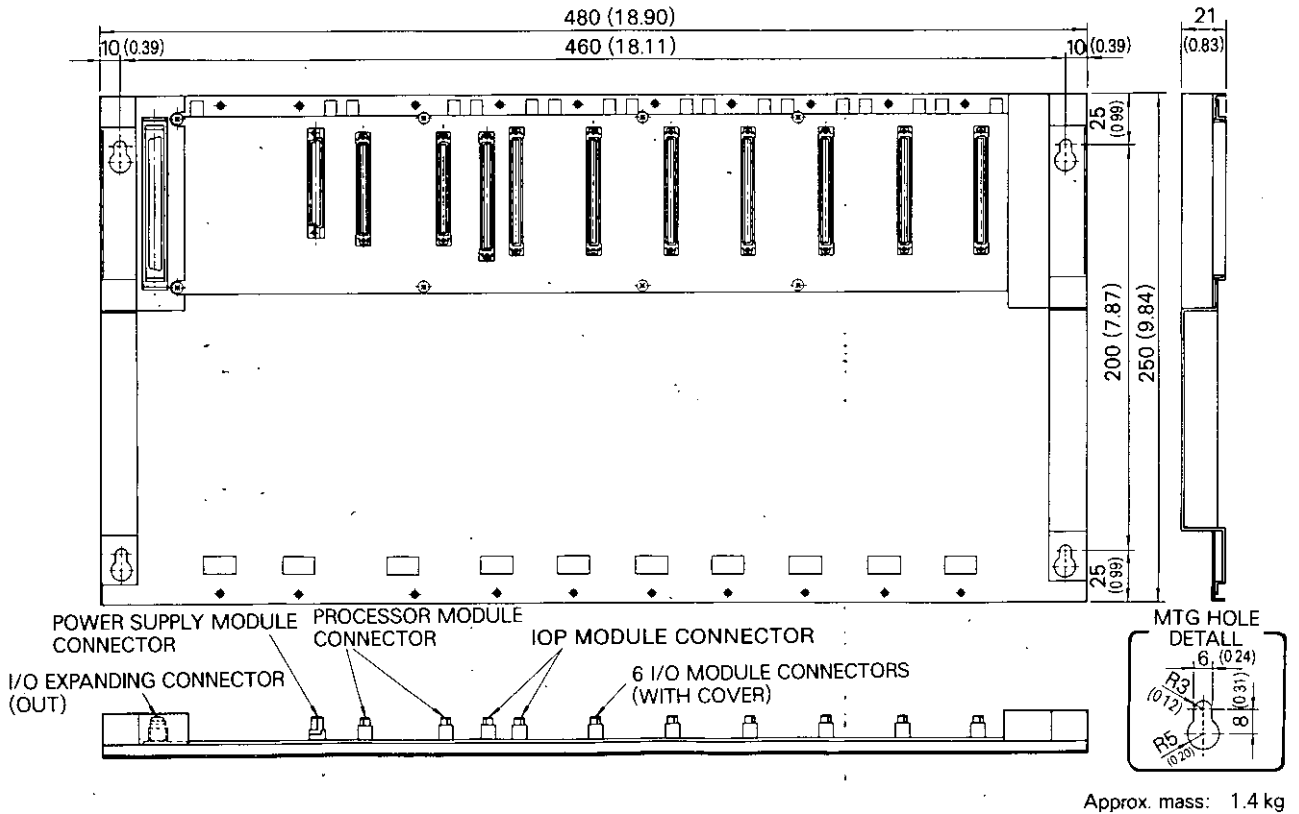
(8) Register Access Panel: RAP (Type DISCT-IF 69)



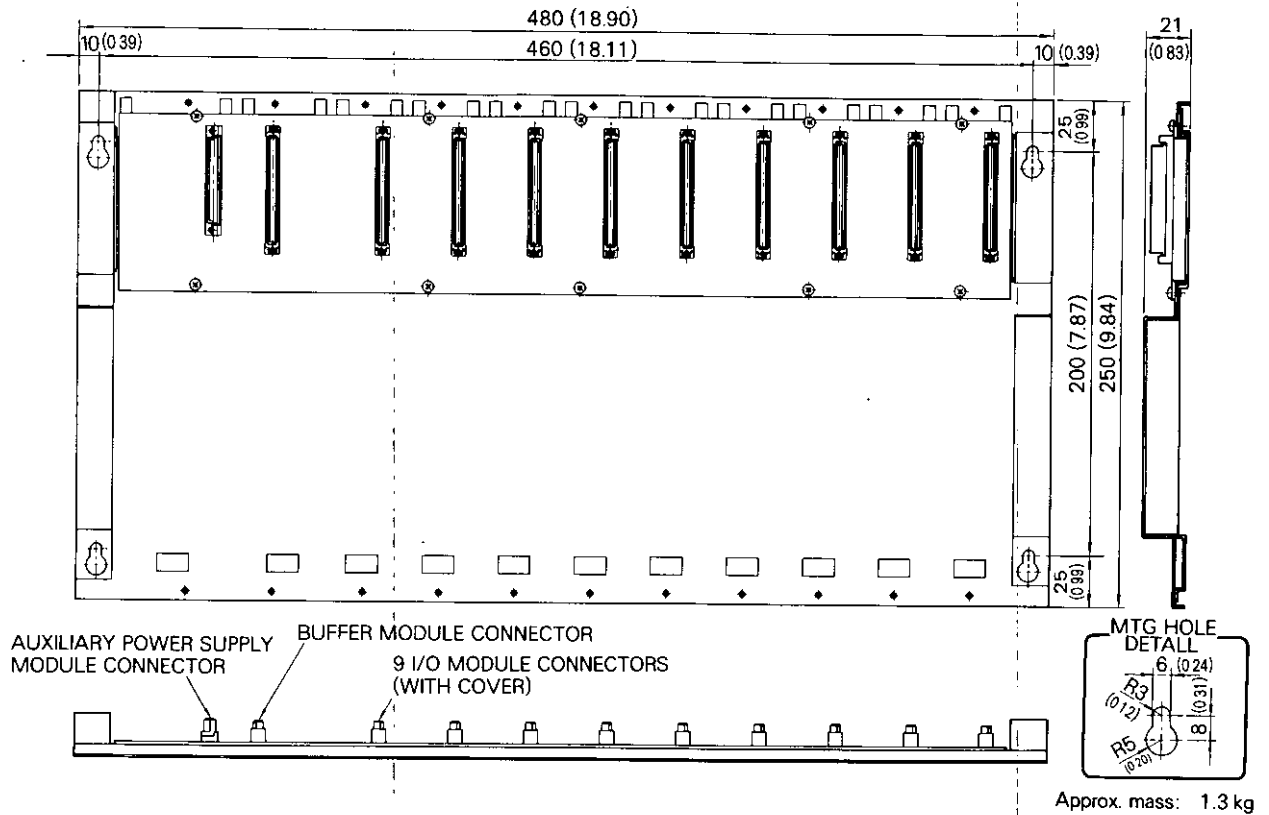
(9) I/O Buffer Module (Type JAMSC-B 2110A)



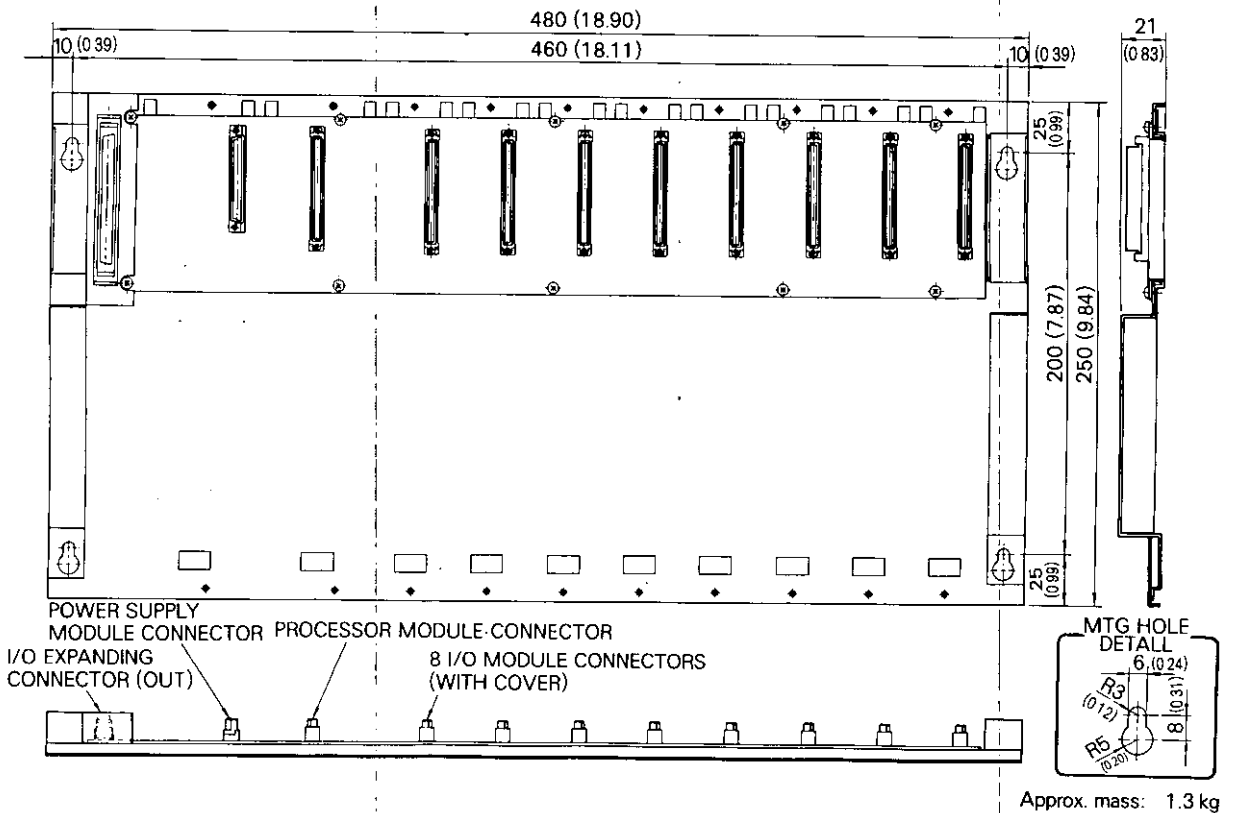
(10) MB 60 Mounting Base (Type JRMSI-MB 60)



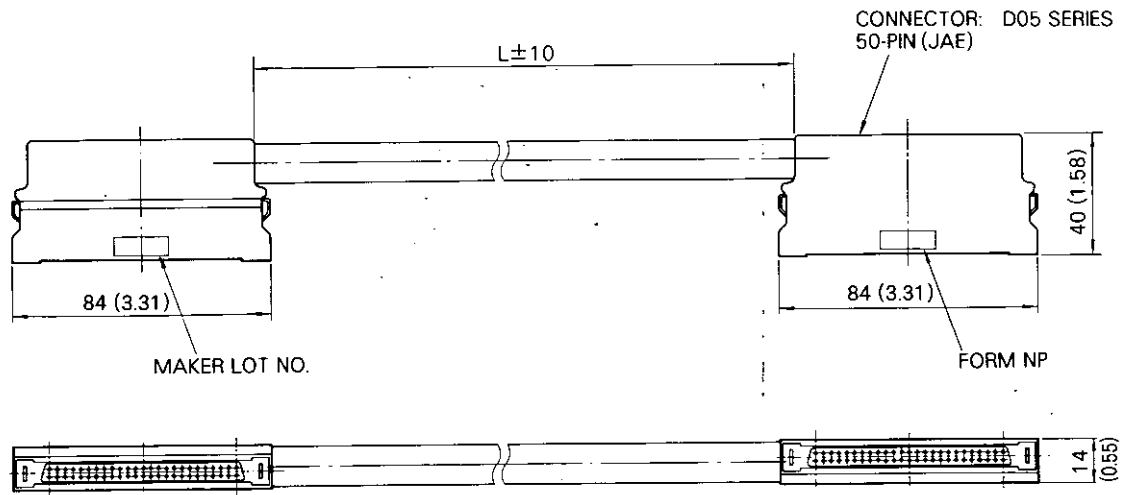
(11) MB 21 Mounting Base (Type JRMSI-MB 22)



(12) MB 70 Mounting Base (Type JRMSI-MB 70)

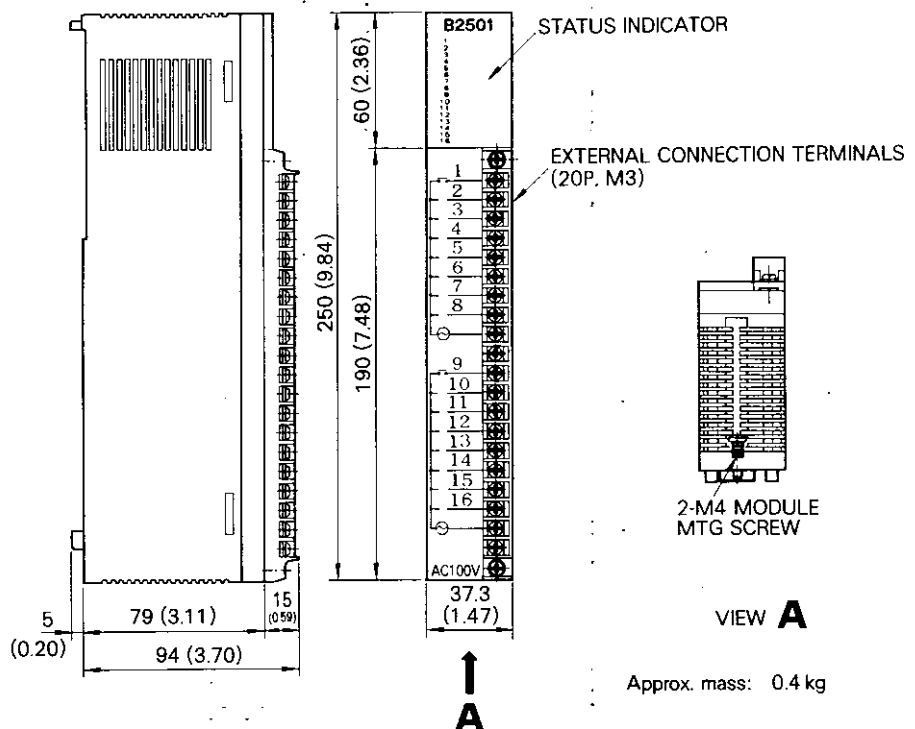


(13) I/O Cable (Type JZMSZ - W 20 - 1, -2)



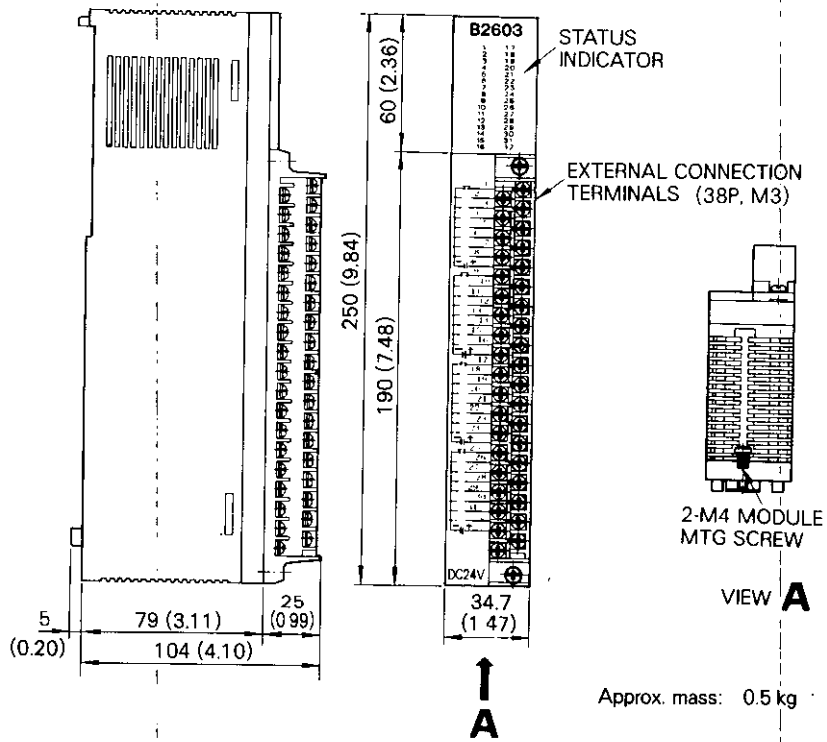
Type JZMSZ-	Length
W 20 - 1	0.5 (19.69)
W 20 - 2	1.5 (59.06)

(14) I/O Module (Type JAMSC-B 2501)

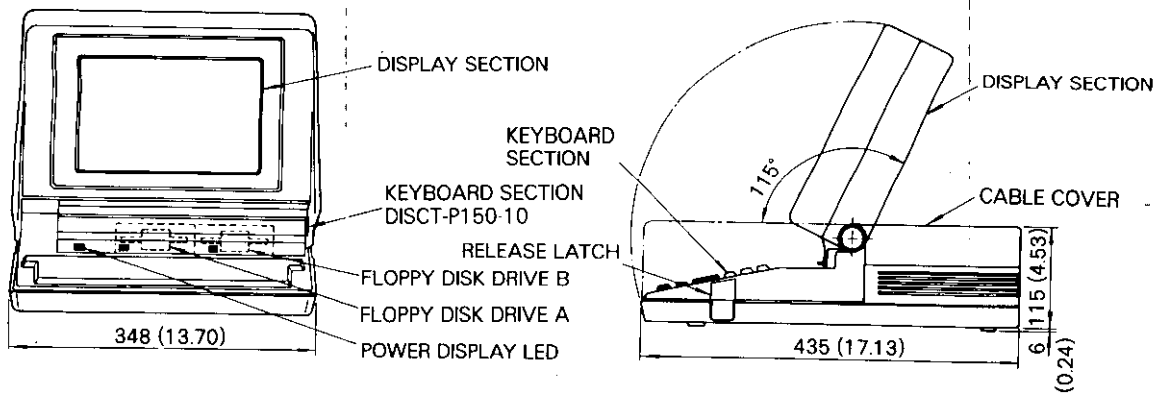


Approx. mass: 0.4 kg

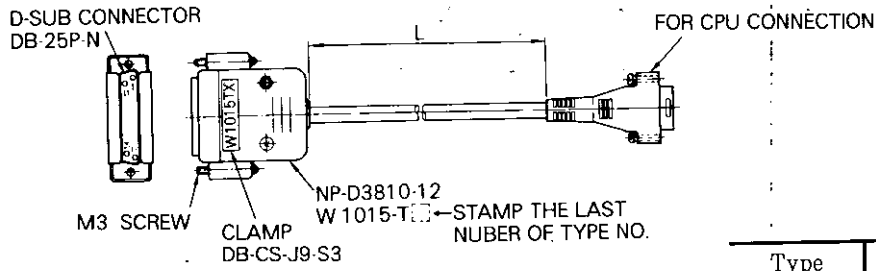
(15) I/O Module (Type JAMSC-B 2603)



(16) Programming Panel P 150 (Type DISCT-P 150)

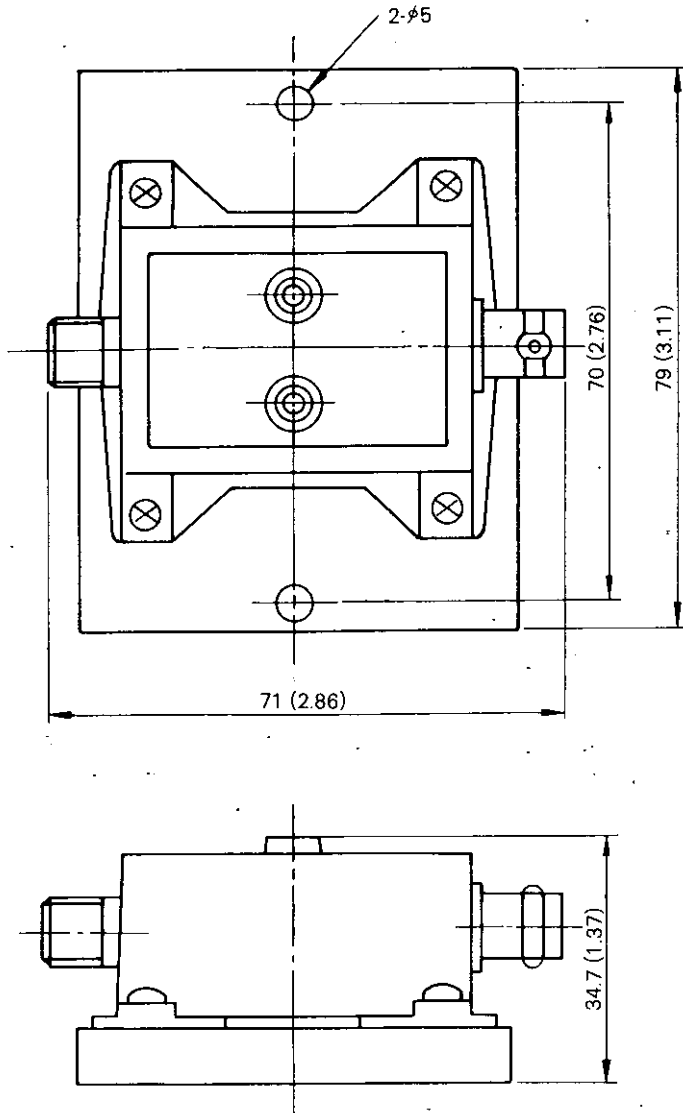


(17) Interface Cable (Type JZMSZ-W 1015-T□□)

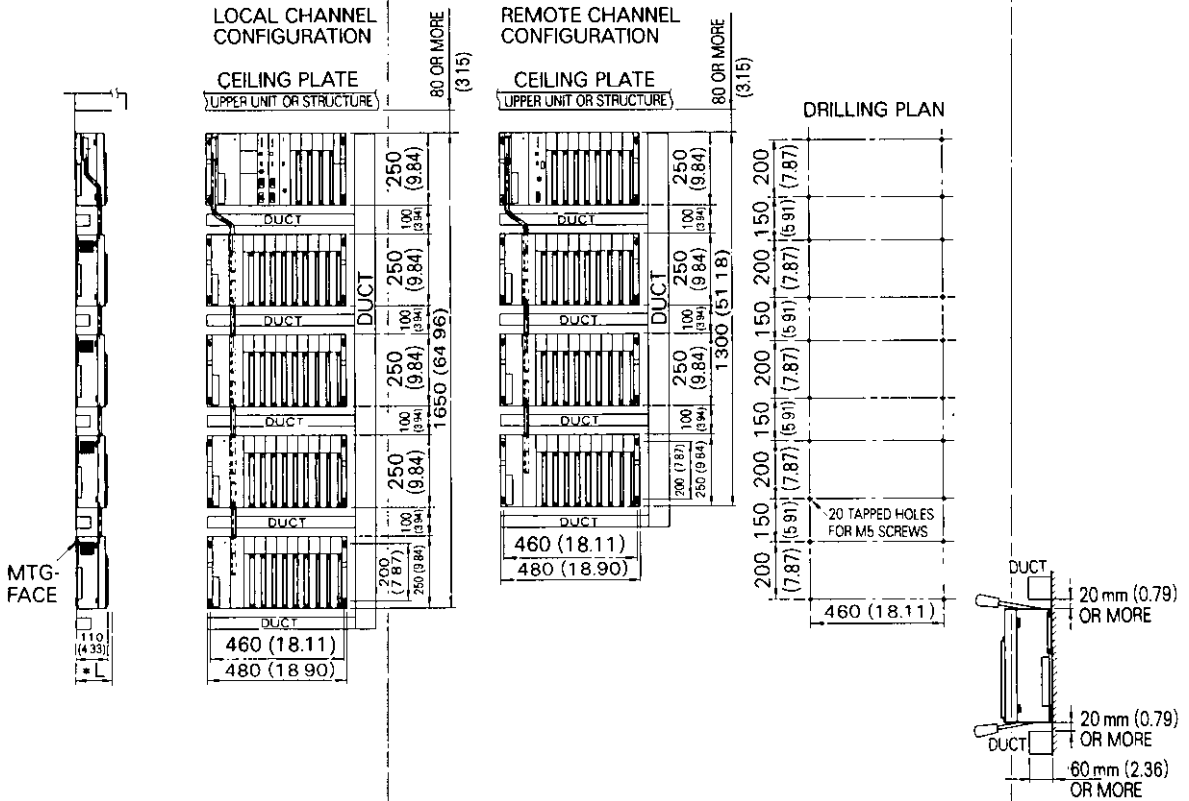


Type	Length	Approx. Mass
W 1015-T 1	2500 (98.42)	0.2 kg
W 1015-T 2	15000 (590.55)	1.0 kg

(18) Remote I/O Adapter (Type T-02698)



APPENDIX D Memocon-SC GL60H, GL70H LAYOUT AND DRILLING PLAN in mm (inch)



MOUNTING PRECAUTIONS

Observe the following when mounting the controller in a frame or other structure. The diagram above can be used as a reference.

1. Provide a spacing of more than 80mm (3.15inch) from the upper module unit or from the top part of the structure to ensure proper ventilation and for easy module replacement.
2. Apply a philips screwdriver (+) slightly diagonally when mounting or removing a module. Provide spaces at the top and the bottom of modules taking screwdriver and duct sizes into consideration.
3. The mounting side of the mounting base is plated to ensure conduction for better noise resistance. The mounting plate of the frame or of the other structure must also allow conduction with the mounting base.
4. Check the * L dimension (required maximum dimension) in the outline drawings of the modules if the module connectors are mounted on the panel surfaces.

MEMOCON-SC GL60H/GL70H DESCRIPTIVE INFORMATION

TOKYO OFFICE

New Pier Takeshiba South Tower, 1-16-1, Kaigan, Minatoku, Tokyo 105-0022 Japan
Phone 81-3-5402-4511 Fax 81-3-5402-4580

YASKAWA ELECTRIC AMERICA, INC.

Chicago-Corporate Headquarters

2942 MacArthur Blvd. Northbrook, IL 60062-2028, U.S.A.
Phone 1-847-291-2340 Fax 1-847-498-2430

Chicago-Technical Center

3160 MacArthur Blvd. Northbrook, IL 60062-1917, U.S.A.
Phone 1-847-291-0411 Fax 1-847-291-1018

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 LTDA.

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 49-6196-888-301

Motoman Robotics AB

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

Motoman Robotec GmbH

Kammerfeldstra e 1, 85391 Allershausen, Germany
Phone 49-8166-900 Fax 49-8166-9039

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

Kipa 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 65-282-3003 Fax 65-289-3003

YATEC ENGINEERING CORPORATION

Shen Hsiang Tang Sung Chiang Building 10F 146 Sung Chiang Road, Taipei, Taiwan
Phone 886-2-2563-0010 Fax 886-2-2567-4677

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

SHANGHAI OFFICE

27 Hui He Road Shanghai 200437 China
Phone 86-21-6553-6600 Fax 86-21-6531-4242

YASKAWA JASON (HK) COMPANY LIMITED

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

TAIPEI OFFICE

Shen Hsiang Tang Sung Chiang Building 10F 146 Sung Chiang Road, Taipei, Taiwan
Phone 886-2-2563-0010 Fax 886-2-2567-4677

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

27 Hui He Road Shanghai China 200437
Phone 86-21-6531-4242 Fax 86-21-6553-6060

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 ELECTRIC CORPORATION

YASKAWA

Specifications are subject to change without notice
for ongoing product modifications and improvements.

MANUAL NO. SIE-C815-17.1B

© Printed in Japan November 1998 91-11
98-5①
591-58